Wildlife Censuses Using Deep Learning in Aerial-Thermal Images

Guillem Martínez Sánchez

Resum– La creació de cens d'animals salvatges és essencial per la conservació de la fauna i ecosistemes. Per realitzar-ho àgilment, la tasca es realitza a partir d'imatges aèries i tèrmiques. En aquest TFG, es fa una revisió dels mètodes de visió per computador, models i datasets de detecció d'objectes utilitzats per la creació de censos de fauna. En aquest context, es proposa i avalua un sistema i s'estudia la utilització de YOLOv5 per la detecció d'animals en temps real. El sistema és avaluat amb un dataset d'imatges aèries i tèrmiques de referència, BIRDSAI. En el treball s'identifiquen problemes de desbalanceig i biaix de mostreig en el dataset. Com a solució, s'ha proposat fer-ne una redistribució estratificada. La proposta millora el mAP en un 33% respecte a una primera aproximació amb els subconjunts suggerits pels autors de BIRDSAI. A més, s'escenifiquen els reptes pel dataset en producció dels Agents Rurals en col·laboració amb el CVC.
Paraules clau– Cens d'Animals Salvatjes, Detecció d'Objectes, Aprenentatje Profund, YOLOv5

Abstract– The creation of wildlife censuses is essential for fauna and ecosystem conservation. To produce them agily, we need to use aerial-thermal images. In this TFG, a revision on computer vision methods, models and object detection datasets for wildlife censuses creation is performed. In this context, a system is proposed and evaluated, and the use of YOLOv5 for real-time wildlife detection is studied. This model is evaluated with a bench-marking wildlife dataset from an aerial-thermal perspective, BIRDSAI. In this work, unbalance and sampling bias is found in the data. As a solution, stratified sampling is proposed. The results show a 33% increase of the mAP with respect to a first approximation with the proposed subsets given by BIRDSAI's authors. Additionally, some insights are provided towards the dataset in production from the Agents Rurals in collaboration with the Computer Vision Center.

Keywords– Wildlife census, Object detection, Deep Learning, YOLOv5

1 Introduction

Wildlife conservation is crucial for several reasons: protecting ecological stability, biodiversity and endangered species or heritage and culture preservation. Governments [1] and several foundations [2, 3, 4] take part in wildlife conservation. As of 2020, Catalonia's authorities [5] catalogued 263 local animal species as endangered or vulnerable. There is not only a way of protecting fauna, but measures such as the creation of wildlife census [6] is proven to be an effective method. However, it requires resources and infrastructure everyone must be conscious of the actions that need to be taken toward fauna conservation. In Figure 1 we can find the causes of wildlife threats.

Wildlife census consists of the count of individuals of certain species in a determined area. It provides information for population monitoring, movement tracking, health and disease control and anti-poaching [7]. However, censusing wildlife is costly since, in many cases, it requires specific tools such as drones, visible and/or infra-red cameras and staff to prepare the information retrieval and to count and validate the results.

There is no rule of how frequent wildlife census should be [8]. However, to observe the population growth trend and for proper monitoring, they should be performed regularly [9].

There are many ways of performing censuses and retrieving fauna information. Species identification, for instance, relies on general characteristics such as animal body pattern, footprint, and sound. Manual techniques such as animal tracking, footprint recognition or drawing specific

• Contact e-mail: guillem.martinezs@uab.cat
• Specialization: Computation
• Tutored by: Daniel Ponsa Musarra (departament)
• Acadamic Year 2021/22

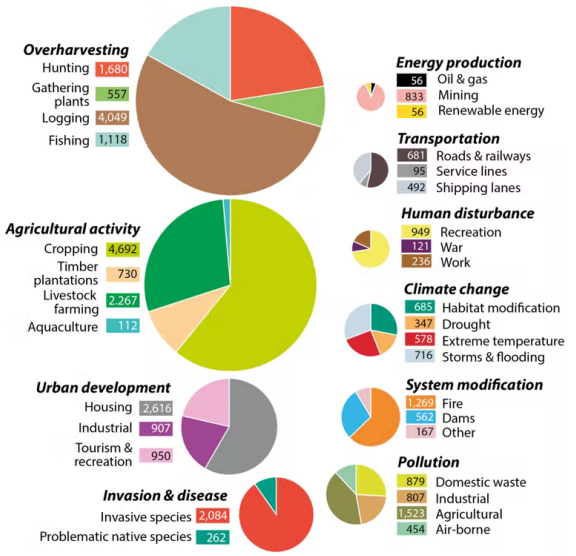


Fig. 1: Causes of wildlife threats. Human behaviour is the main cause to the threats to wildlife. [11].

animal characteristics were used in the past. However, they are outdated. Ground cameras have been commonly used before aerial imaging was made more effective for this task. The use of UAVs and computer vision techniques are currently used to aid the creation of wildlife censuses. With these techniques we can capture images from above, removing the occlusion of the environment and at the same time process, the information faster [10].

In this TFG, we collaborate with Catalonia's Agents Rurals to create a computer vision system from aerial-thermal imagery to aid the generation of wildlife censuses on Catalonia's territory. The Agents Rurals use quadcopters with thermal and visible spectra cameras to create wildlife census. They record from a-far mountain areas during the early morning to observe higher contrast in the thermal images between animals and their environment and manually count the species. We propose to develop a computer vision system aiming to reduce time and increase agility when creating wildlife censuses. However, this task has two main challenges:

- Large recording distance: Otherwise animals would run away if the drone were to be close enough.
- Forestry occlusion: animals are commonly observed partially, since they hid between trees, rocks and vegetation.

The result of joining those problems make the task of detecting animals from images difficult, since in most cases they are seen in 2-8 pixel areas.

Considering the previous information, we need to fulfill the following sub-objectives:

- Perform a survey and obtain infrared wildlife datasets.
- Develop an object detection algorithm for wildlife detection on infrared images.
- Evaluate, using objective and coherent metrics, the algorithm's performance with benchmark datasets and our own dataset.

1.1 Methodology

To achieve the project's goals, a research project methodology has been followed: First, a state-of-the-art survey was done to understand the technical context and possibilities. Then, a baseline with a simple object detection method for wildlife fauna from aerial-thermal images was implemented. After the baseline was settled, an evaluation was made to find the system's weaknesses. This was followed by iterations where different approaches and upgrades were considered to increase its performance.

During the development of this TFG, the Agents Rurals unit acquired thermal sequences to develop the system. Unfortunately, the amount of collected data was not enough for the methodological development of a system. For this reason, the proposed algorithm was done using a bench-marking dataset, which did not have the same characteristics as the acquired images by the Agents Rurals but allowed to perform a study of wildlife detection from aerial-thermal images.

The data used by the pipeline provided training, validation and testing subsets, where testing data was only used during the test phase and never used during training, to assure model generalization.

When evaluating the Object Detection model's performance the Mean Average Precision (mAP) [12] is commonly used by Academia, hence, it was the performance metric for the developed systems.

At the beginning of the project, five stages were defined: Planning, Research, Experimentation, Analysis & Improvement and Presentation. Each stage was defined with multiple main tasks that can be found in the Gantt [13] diagram See Figures A.1, A.2 in the appendix. During the project development, short-term planning with Trello Board [14] was followed. A workflow toward the next tasks was defined in each stage.

The project was made in Python 3 and I used PyCharm IDE connected to a container allocated within Computer Vision Center computational resources. During the project development, PyTorch was the main deep learning framework. Other core Python libraries were NumPy, Pandas, Scikit-Image and OpenCV.

For the communication between me and my supervisor, we used Microsoft Teams [15]. During the development of this project, I also worked with Kanban [16], specifically Trello Board's framework [14]. Even in individual projects, it can be used to increase productivity [17, 18].

For use and further improvement of the system, the code and documentation can be found at GitHub [19].

2 State Of The Art

To achieve our goals and solve the problem, we need to evaluate previous answers to similar problems that apply the tools that we intend to use [10, 20, 21]. In this section, we cover Deep Learning (DL) for Object Detection (OD), Small Object Detection (Small-OD) and Thermal Infrared (TIR) imaging.

2.1 Deep Learning and Object Detection

Deep Learning (DL) is the stat-of-the-art in several computer vision tasks, including OD. Over the years, many DL OD models emerged increasing inference time and performance [22]. However, there is a speed-performance tradeoff [23] that divided the landscape into two main meta-architectures:

- Two-shot detectors (TSD): These perform region proposal and then classification of those regions and refinement of the location prediction. These have higher accuracy but less inference speed compared to Single-shot detectors.

- Single-shot detectors (SSD): These skips the region proposal stage and yields final localization and content prediction at once. These produce faster results but with less accuracy than Two-shot detectors.

Nowadays, we can find code repositories that group the main OD proposals, such as Detectron2: a library that propose a modular, extensive design that allows users to plug custom module implementations [24, 25]. It counts with an OD model zoo including Faster R-CNN, Mask R-CNN or RetinaNet and other TSD or SSD.

One of the earliest SSD architectures is the You Only Look Once (YOLO) family. Since the first version, YOLOv1 [26] was released in 2015, it has received incremental updates. In May 2020 the first version of YOLOv5 was available for public use [27]. All YOLOv5 models are pretrained with COCO [28] dataset. This framework is currently one of the most used in the industry because of its easy training and application for real-time OD.

In practice, real-time systems usually require tracking after producing object detection results. In this topic, implementations as YOLOv5 DeepSort [29] propose a two-stage-tracker that generates predictions with a YOLOv5 architecture and applies tracking with the Deep Sort algorithm [30]

In the context of wildlife surveillance, we can find some works that use OD models to increase human performance. Delplanque et al. [31] proposed the training and evaluation of three OD models to detect six different animal species of African mammals from aerial imagery, resulting in an 80% precision in a custom dataset.

In another study [32], Barbedo et al. provided an extensive survey on Convolutional Neural Network(CNN) models in the context of cattle monitoring, proving that deep learning and CNN can be used robustly used under non-ideal terrain circumstances.

In Tibetan Plateau, a modified Faster R-CNN for kiang detection [33] was used. The research concluded that the adopted tactics can be applied to either a semiautomatic survey to accelerate manual verification by 25 times or an automatic survey with an F1 score of approximately 90%. Hence, this work proved that UAS imagery and deep learning can generate automatic/semiautomatic, high-performance and efficient wild animal surveys and census creation.

Hong et al. [34] used 5 State-of-the-art OD models to detect birds. Their results showed Average Precision values ranging from 85% to 95% and an inference speed-accuracy trade-off between Faster-RCNN and YOLOv3. Overall, showing that deep learning, UAV imagery and OD can be used for bird detection.

However, the previous case studies only tackle imagery in the visible spectra. To better approach our problem we have to perform a survey on TIR imagery and Small-OD since these are the type of images to be processed in this work.

2.2 Thermal Infrared Imaging and Small Object Detection

Thermal imaging is the process of converting infrared (IR) radiation (heat) into visible images that depict the spatial distribution of temperature differences in a scene viewed by a thermal camera [35]. It is used in different contexts due to its capacity to differentiate the objects of interest from the background.

In wildlife, detection is especially interesting since animals camouflage with their environment. Hence, TIR imaging is commonly used in this subject.

In [36], Oishi et al. propose an application of the moving wildlife algorithm (DWA) to drastically improve the detection performance by 48% compared to human performance.

Another example of this is the work proposed by Lee et al. [37]. They proposed a Sobel filter method that provides a 0.804 precision and 0.699 recall on a custom dataset.

In his thesis, Marais et al. [38] propose a system including a region proposal algorithm and a deep learning algorithm. The results showed 98% accuracy and human workload reduction of 97%

Another problem that we need to face is the small size of the objects of interest. To address this, several works on this topic had been published. An example of such is this pre-print by Benjumea et al. [39], they propose structural modifications to [27] to increase by 6.9% the detection of small objects and only add 3ms to inference time. Their approach included modifications replacing the backbone for a ResNet50 [40] and modifying the neck from a PANet [41] to a biFPN [42].

Another example is the work proposed by Singh et al. [43] presented a "foveal"object detection framework with the idea of skimming over the images and spotting interesting regions that would be further processed, like human vision. They applied Scale Normalized Image Pyramids (SNIP) that enable attention to objects of different sizes and an efficient spatial sub-sampling scheme called Scale Normalized Image Pyramid with Efficient Resampling (SNIPER). AutoFocus is the resulting algorithm for joining

Table 1: Computer Vision Center computing resources availability. Not all CPU and RAM resources were used during the project development.

CPU	4x Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz
GPU	2x GeForce RTX 3090
RAM	187 GB
OS	Ubuntu 18.04.5 LTS

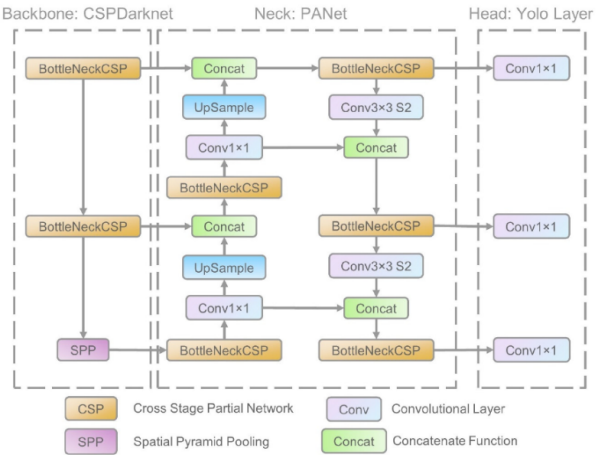


Fig. 2: YOLOv5 consists of three parts:(1) Backbone: CSPDarknet, (2) Neck: PANet, and (3) Head: Yolo Layer. Figured borrowed from [47]

this contributions and increases 2.5-5x inference time.

To address the thesis goal, I will evaluate YOLOv5 [27] performance for thermal images, since all the beforementioned works employ RGB images. To do so, I use BIRDSAI's dataset [44], a savanna wildlife dataset from an aereal-thermal perspective.

3 Materials and Methods

In this section, we describe the data, preprocessing and used models used during the development of this project.

3.1 YOLOv5

YoloV5 [27] is an open-source architecture family created by Ultralytics for real-time OD purposes. It is considered a state-of-the-art method because of its inference speed without drastic performance loss, computational resources demand and simplicity to train and deploy [45]. YOLOv5 is originally built in PyTorch [46] ML open-source framework, which is also known for its high abstraction level, easy learning and development. It is considered one of the top DL tools.

YOLOv5 architecture (see Figure 2) consist of three parts: The input goes through a Cross Stage Partial Network (CSPNet) [48] which perform feature extraction, and then the data is fed to Path Aggregation Network (PANet) [41] for feature fusion. Lastly, YOLO Layer outputs detection results (class, score, location, size). The combination of a CSPNet, which reduces computation and enhances performance compared to other CNNs, jointly with a Feature Pyramid Network (FPN)[49] such as PANet, that decreases computation cost and enriches multi-scale OD makes YOLOv5 a very good fit to the project needs.

The architecture used in this project is a YOLOv5m that can be found at [27].

Fig. 3: In the image we can see two four real-world images from the BIRDSAI dataset [44].

3.2 Datasets

As already mentioned, at the time of developing this TFG there was no annotated data from the acquisition services captured by Agents Rurals. Due to that, a search was done to identify datasets available, where to check the performance of YOLOv5 on thermal data. There are no aerial-thermal datasets that perfectly match the project challenges. However, there exist two potential datasets: The Benchmarking IR Dataset for Surveillance with Aerial Intelligence (BIRDSAI) [44] and, NOAA Arctic Seals 2019 (Artic Seals Dataset) [50]. Both of them contain labelled bounding boxes of thermal animals from an aerial perspective. In practice, the thermal images from Arctic Seals Dataset contain many sequences with corrupted thermal information, which made this dataset unusable. Consequently, we only use the BIRDSAI dataset, even if the animals are bigger than what we can expect in the animal species that are observable in the sequences collected by Agents Rurals.

BIRDSAI [44] is a TIR video dataset containing nighttime images of animals and humans in Southern Africa. It is composed of 48 real aerial videos and 124 AirSim's [51] synthetic aerial videos, for a total of 62k and 100k images, respectively.

The dataset contains a total of 9 classes, including an unknown class, human, elephant and lion in real and synthetic data, giraffe and dog occur only in real data and crocodile, hippo, zebra, and rhino occur only in synthetic data.

In the released paper, the authors state that is interesting the use of simulated data for this topic, since, is costly and requires a lot of work to generate real-world data and annotations. They created the simulated data with the AirSim-W platform [52] and a savanna scenario also defined in [52]. They did the simulation from a UAV perspective.

A comparison between real-world and synthetic data can be found in Figures 3, 5

Table 2: BIRDSAI Dataset. Baseline-Real, experiment. Metrics per class.

Class	P	R	mAP@0.5	mAP@0.5:0.95
all	0.396	0.167	0.161	0.076
elephant	0.847	0.758	0.817	0.393
human	0.205	0.021	0.039	0.014
lion	0.002	0.014	0.000	0.000
giraffe	0.193	0.058	0.061	0.033
dog	1.000	0.000	0.000	0.000
unknown	0.128	0.153	0.046	0.017

Table 3: BIRDSAI Dataset. Baseline-Total experiment. Metrics per class.

Class	P	R	mAP@0.5	mAP@0.5:0.95
all	0.399	0.153	0.156	0.059
elephant	0.848	0.674	0.744	0.283
human	0.399	0.070	0.104	0.035
lion	0.021	0.043	0.007	0.003
giraffe	0.054	0.060	0.062	0.026
dog	1.000	0.000	0.000	0.000
unknown	0.069	0.066	0.020	0.008

BIRDSAI also include tracking information for each of the animals and humans in the videos. Nevertheless, we do not use it in this project since our focus is to validate the performance of the YOLOv5 detector on single thermal images.

4 Experiments and Results

In this section, the different experiments carried out in my work using BIRDSAI's dataset 3.2 are presented, including their purpose, design factors and details, jointly with the results in an organised way.

YOLOv5m is the common model for all experiments. Additionally, the default hyper-parameters are used, as mentioned in the best practices and tips for training YOLOv5 [53].

All the experiments tune the image-size parameter between 640 and 1280. this parameter re-scales the images, making small object detection easier.

Also, training and validation are divided with an 80-20 ratio, and, none of the subsets shares the same frames of a sequence.

4.1 Training with real sequences

The objective of this experiment was to define a first baseline with the dataset, as they do in the original BIRDSAI's paper [44]. Hence, this experiment uses the default training and test set proposed by the authors of the dataset.

This experiment was performed at first with Google Colaboratory's free computational resources. Hence, it was trained with only the real-world data since, the simulated data could not fit the space requirements.

However, in this iteration mistakes were made. While splitting the data between training and validation, instead of splitting the sequences, the frames were divided between the two subsets, therefore, the validation data was practically the same as the training data since they were sharing contiguous frames. The results produced by this first iteration had overfitting.

In the second iteration, this mistake was solved by dividing the sequences into training and validation and not the frames within the sequences. Additionally, this iteration was performed with Computer Vision Center resources that are stated at Table 1.

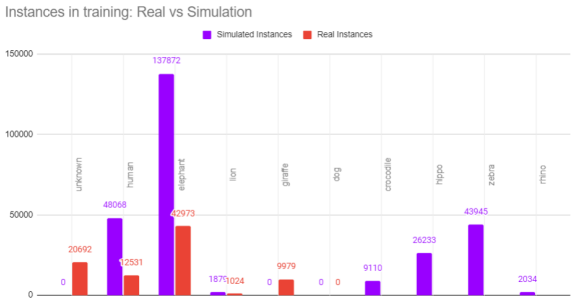


Fig. 4: This graphic compares the instances distribution of the joined real-world and simulation training set. We can also see from this perspective that the simulated data does not help to balance the real-world instances.

In Table 2 we can see the results for the second iteration of this experiment. We can observe that the model was effective (high precision, recall and mAP) with the class 'elephant' but noneffective for the remaining classes. Hence, we can suspect that either there are not enough instances of the other classes. This is understandable since, most of the data is simulated and used in training in the original paper.

This, naturally leads to the following second experiment, training with real and synthetic sequences.

4.2 Training with real and synthetic sequences

In this experiment, the objective is, still, to define a performance baseline for YOLOv5. And, as stated before, we add the simulated data to the training stage, so we can increase the number of instances per class and train the model with more data.

In table 3 we can see the results for this experiment. After adding the simulated data, there is no increase in the performance of the model. By observing the simulated data in 4, four classes that do not appear in the test set are added to the training set. Also, the 'lion' and 'dog' classes have very few representations in the training data. B. Therefore, the simulated data does not solve the unbalance problem but made it worse.

Additionally, the lack of domain transfer should be considered, since simulated data is different from real-world data.

To have a dataset without the unbalance problem a subset dataset is created with the most representative classes.

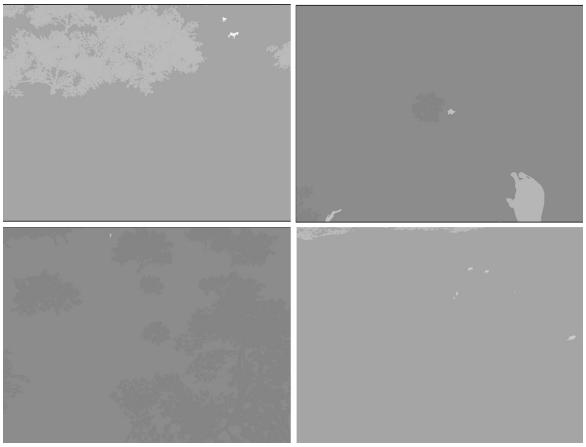


Fig. 5: In the image we can see two four synthetic images from the BIRDSAI dataset [44]. We can see a huge disparity between real-world data from Figure 3

4.3 Human-Elephant-Giraffe experiment

To understand whether the data or the model is the problem, the experiment is simplified. The idea behind it is to only use the three most representative classes: 'human', 'elephant' and 'giraffe' classes. The experiment name is abbreviated to 'HEG'. It is not necessary to move the other labels to unknown, nor use the unknown class, since the sequences are divided by classes and, in the targeted sequences, only HEG classes appear.

Table 4: BIRDSAI Dataset. HEG experiment. Metrics per class.

Class	P	R	mAP@0.5	mAP@0.5:0.95
all	0.480	0.254	0.296	0.136
elephant	0.850	0.623	0.716	0.348
human	0.546	0.067	0.111	0.032
giraffe	0.045	0.073	0.061	0.029

In table 4 we can see the results of this experiment. After simplifying the problem by only using three classes we can see that results do not improve as expected. The performance metrics of 'human' and 'giraffe are very low. The distribution of instances between training and test can be seen in Figure 6.

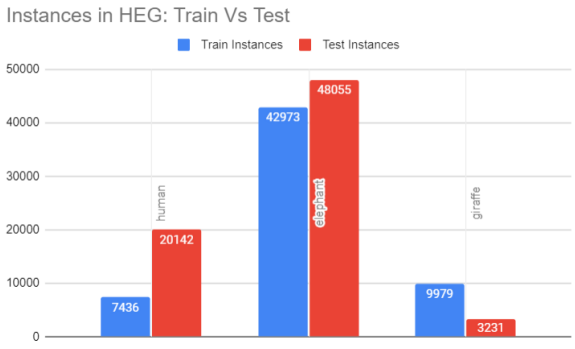


Fig. 6: This graphic compares the instances distribution of the proposed HEG training and test.

After this experiment, and taking an analytical look at the original testing set, a different angle is tried. The hypothesis behind it is that the test subset could contain sequences that are very different from the training subset, difficulting the model task to generalize.

4.4 Split & Rearrange

This experiment consists in, by only using the real data, modify the split between training and test. The test set can include sampling bias since all the instances are extracted from fifteen short video sequences that can be different from the (also few and short) thirty-two training video sequences. In Figure 7 can be seen the distribution between training and test instances.

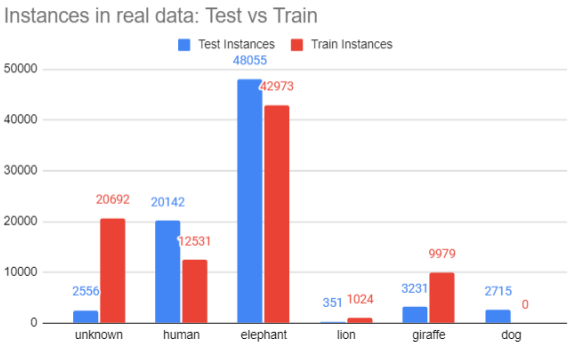


Fig. 7: This graphic compares the instances distribution of the original training and test. Overall is an unbalanced dataset, also, there are no instances of dogs in the training set, but there are in the test set.

To do so, all the real-world sequences are divided into two and training and test subsets contain one-half of the sequence each. This way we assure stratified sampling for the test set. A visualization of the resulting distribution can be seen in Figure 8. This experiment provides a fairer distribution and assures that the sequences between training and test are similar.

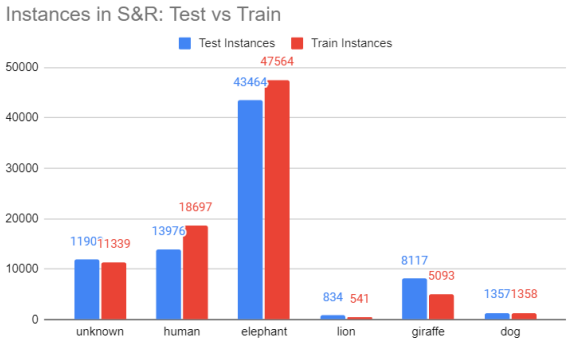


Fig. 8: This graphic compares the instances distribution of the proposed training and test. It is a fairer distribution between training and test, nonetheless, the dataset is unbalanced still.

In table 5 we can see the results of this experiment. As expected, we can observe that the metrics considerably increase compared to the other experiments, proving that the dataset contains sampling bias.

Table 5: BIRDSAI Dataset. Split & Rearrange experiment. Metrics per class.

Class	P	R	mAP@0.5	mAP@0.5:0.95
all	0.756	0.455	0.484	0.211
elephant	0.878	0.85	0.908	0.507
human	0.648	0.445	0.483	0.019
lion	0.754	0.176	0.158	0.057
giraffe	0.74	0.575	0.638	0.242
dog	0.753	0.020	0.033	0.018
unknown	0.764	0.668	0.683	0.251

4.5 HEG: Split & Rearrange

To see how the HEG experiment would perform with the stratified sampling method, the HEG experiment is repeated and applied this tactic.

Table 6: BIRDSAI Dataset. HEG: Split & Rearrange experiment. Metrics per class.

Class	P	R	mAP@0.5	mAP@0.5:0.95
all	0.797	0.639	0.707	0.329
elephant	0.892	0.834	0.899	0.509
human	0.748	0.450	0.539	0.219
giraffe	0.753	0.632	0.683	0.258

In table 6 we can see the results for this experiment. The mAP increased by 0.4 compared to the HEG experiment with the original training-test split.

5 Discussion

In this section, knowledge is extracted from the results and new questions are formulated towards future work.

Table 7: BIRDSAI Dataset. Class average results over the experiments described in the section 4. Real and Sim are the original real-world and simulated data. S&R stands by Split & Rearrange; this subset is extracted from the real-world data.

Experiment	Data	P	R	mAP@0.5
Baseline-Real	Real	0.396	0.167	0.161
Baseline-Total	Real+Sim	0.399	0.153	0.156
S&R	Real-S&R	**0.756**	**0.455**	**0.484**
HEG	Real	0.480	0.254	0.296
S&R-HEG	Real-S&R	**0.789**	**0.639**	**0.707**

In Table 7, the general results for all the experiments are shown. The S&R experiments are the ones with the best performance. The first S&R experiment shows that stratified sampling solves the sampling bias in the dataset since it improves the precision, recall and mAP by 0.357, 0.302 and 0.323 points respectively.

In the S&R experiment where we only use the most representative classes (HEG), the performance also increases with respect to the HEG baseline experiment in precision, recall and mAP by 0.309, 0.396 and 0.411 respectively. This result shows that when properly representing the instances we can get better results.

The different experiments perform within the TFG allowed the identification of a set of problems in the bench-marking dataset used:

The main problem is that most classes are underrepresented and lack a proper number of instances. ¡PROVE¿.

This post by Ultralytics [53], empathizes that for proper training of YOLOv5, a dataset should contain more than 1.5k images per class and more than 10k instances per class, which is something that BIRDSAI's struggles with.

The second one is that the simulated data does not help to balance out the underrepresented instances. The simulation data add classes that are not seen in the test set and mainly increases the number of instances of the already well-represented classes such as 'elephant. See Figure 4.

Moreover, using simulation data requires domain adaptation, see Figure 5, which is a technique that is not native to YOLOv5 and was not applied in this work.

The third problem is that the original test set contains sequences that are not representative of the training data. Some of the previously stated issues include sampling bias, and simulation data not representative of real data (classes that are not seen in real-world data or the simulated data being way different from real-world data). This can be proven by the S&R experiments, where stratified sampling is applied.

6 Conclusions and Future Work

In this work, a system for wildlife census from aerial-thermal images was implemented. For this, a survey on computer vision and deep learning methods used for real-time object detection and small object detection was performed. Moreover, datasets for the creation of wildlife censuses using TIR images from an aerial perspective were reviewed.

YOLOv5 was the chosen method for this task since it showed better performance for RGB images.

While applying this algorithm to BIRDSAI's dataset, the results identified low performance.

A series of experiments were performed to analyze the data. The experimentation concluded that the dataset had class unbalance and sampling bias. A stratified redistribution was proposed to fix these problems. The changes increased the mAP by 33% when using the entire dataset and 41% when exclusively using the HEG classes.

For future work, I would propose to implement oversampling of the underrepresented classes, since YOLOv5 does not count with this technique nor weighted losses by default.

Also, other experiments could be done with different models other than YOLOv5, such as Fast-RCNN, Mask-RCNN or other approaches that include domain adaptation or multi-scale features.

Additionally, some other points can be written down from this TFG:

First, there are few low-quality datasets for wildlife object detection from an aerial-thermal perspective. Hence, in this field, more datasets are needed.

In the second place, before committing to BIRDSAI's dataset, a more exhaustive data exploration should have been done. Knowing that this was a very challenging dataset would have been more obvious.

Foreseeing the upcoming dataset from Agent Rurals, it is necessary to comment that it will be an even less trivial dataset since the object occlusion will be way higher than a savanna dataset such as BIRDSAI. Moreover, the size of the objects will be a lot smaller.

REFERÈNCIES

[1] Generalitat de Catalunya, "Qui són els agents rurals?," *Departament d'Interior*, 2022.

[2] Animal Welfare Institute, "Wildlife," https://awionline.org/content/wildlife (Accessed: Feb 2022).

[3] Micaela Samodelov, "How censuses support wildlife conservation," https://www.awf.org/blog/how-censuses-support-wildlife-conservation (Accessed: February 2022).

[4] David Serramitjana, "Qui som?," https://fundaciofauna.wixsite.com/fundaciofauna (Accessed: March 2022).

[5] Generalitat de Catalunya, "Nota de premsa, territori," https://territori.gencat.cat/ca/inici/nota-premsa/?id=387762 (Accessed: February 2022).

[6] Mariana Nagy-Reis, Melanie Dickie, Péter Sólymos, Sophie L. Gilbert, Craig A. DeMars, Robert Serrouya, and Stan Boutin, "'wildlift': An open-source tool to guide decisions for wildlife conservation," *Frontiers in Ecology and Evolution*, vol. 8, 2020.

[7] CB Banga, B Besbes, B Balvay, L Chazo, OM Jamaa, A Rozstalnyy, G Rovere, A Toto, KR Trivedi, et al., "Current situation of animal identification and recording systems in developing countries and countries with economies in transition," *Farm animal breeding, identification, production recording and management Proceedings of the 37th ICAR Biennial Session*, pp. 53–59, 2010.

[8] A Marm Kilpatrick, Joseph R Hoyt, R Andrew King, Heather M Kaarakka, Jennifer A Redell, J Paul White, and Kate E Langwig, "Impact of censusing and research on wildlife populations," *Conservation Science and Practice*, vol. 2, no. 11, pp. e264, 2020.

[9] World Wildlife Fund, ," https://wwfeu.awsassets.panda.org/ downloads/counting_wildlife_mozambiqe_english.pdf (Accessed: March 2022).

[10] Tinao Petso, Rodrigo S. Jamisola, and Dimane Mpoeleng, "Review on methods used for wildlife species and individual identification - european journal of wildlife research," *SpringerLink*, Dec 2021.

[11] Sean L. Maxwell, Richard A. Fuller, Thomas M. Brooks, and James E. M. Watson, "Biodiversity: The ravages of guns, nets and bulldozers," *Nature News*, Aug 2016.

[12] Jonathan Hui, "Map (mean average precision) for object detection," https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection (Accessed: March 2022).

[13] "What is a gantt diagram," https://www.gantt.com/ (Accessed: April 2022).

[14] Trello Board, "What is trello?," https://help.trello.com/article/708-what-is-trello (Accessed: March 2022).

[15] Microsoft Teams, "Microsoft teams," https://www.microsoft.com/es-es/microsoft-teams/group-chat-software (Accessed: April 2022).

[16] Atlassian, "Kanban - a brief introduction," https://www.atlassian.com/agile/kanban (Accessed: March 2022).

[17] Bryan Collins, "How to use kanban to become insanely productive: A short guide," https://www.forbes.com/sites/bryancollinseurope/2018/07/19/how-to-use-kanban-to-become-insanely-productive-a-short-guide/?sh=22fde1a43c16 (Accessed: March 2022).

[18] Jessica Everitt, "The complete guide to personal kanban," https://www.wrike.com/blog/complete-guide-personal-kanban/ (Accessed: March 2022).

[19] "Where the world builds software," https://github.com/ (Accessed: April 2022).

[20] Audrey Verma, René van der Wal, and Anke Fischer, "Imagining wildlife: New technologies and animal censuses, maps and museums," *Geoforum*, vol. 75, pp. 75–86, 2016.

[21] Colin J. Torney, David J. Lloyd-Jones, Mark Chevallier, David C. Moyer, Honori T. Maliti, Machoke Mwita, Edward M. Kohi, and Grant C. Hopcraft, "A comparison of deep learning and citizen science techniques for counting wildlife in aerial survey images," *Methods in Ecology and Evolution*, vol. 10, no. 6, pp. 779–787, 2019.

[22] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee, "A survey of modern deep learning based object detection models," *arXiv.org*, May 2021.

[23] Gal Hyams and Dan Malowany, "The battle of speed vs. accuracy: Single-shot vs two-shot detection meta-architecture," Apr 2020.

[24] Facebook Meta, "Detectron2: A pytorch-based modular object detection library," 2022.

[25] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You only look once: Unified, real-time object detection," *arXiv*, 2015.

[27] Glenn Jocher, "Yolov5," https://github.com/ultralytics/yolov5 (Accessed: April 2022).

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár, "Microsoft coco: Common objects in context," *arXiv*, 2014.

[29] Mikel Broström, "Real-time multi-object tracker using yolov5 and deep sort," https://github.com/mikel-brostrom/Yolov5_DeepSort_Pytorch (Accessed: April 2022).

[30] Nicolai Wojke, Alex Bewley, and Dietrich Paulus, "Simple online and realtime tracking with a deep association metric," *CoRR*, vol. abs/1703.07402, 2017.

[31] Alexandre Delplanque, Samuel Foucher, Philippe Lejeune, Julie Linchant, and Jérôme Théau, "Multispecies detection and identification of african mammals in aerial imagery using convolutional neural networks," *Remote Sensing in Ecology and Conservation*, vol. n/a, no. n/a, Aug 2021.

[32] Jayme Garcia Arnal Barbedo, Luciano Vieira Koenigkan, Thiago Teixeira Santos, and Patrícia Menezes Santos, "A study on the detection of cattle in uav images using deep learning," *Sensors*, vol. 19, no. 24, 2019.

[33] Jinbang Peng, Dongliang Wang, Xiaohan Liao, Quanqin Shao, Zhigang Sun, Huanyin Yue, and Huping Ye, "Wild animal survey using uas imagery and deep learning: modified faster r-cnn for kiang detection in tibetan plateau," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 169, pp. 364–376, 2020.

[34] Suk-Ju Hong, Yunhyeok Han, Sang-Yeon Kim, Ah-Yeong Lee, and Ghiseok Kim, "Application of deep-learning methods to bird detection using unmanned aerial vehicle imagery," *Sensors*, vol. 19, no. 7, 2019.

[35] Science Direct, "Thermal imaging," https://www.sciencedirect.com/ (Accessed: April 2022), 2022.

[36] Yu Oishi, Hiroyuki Oguma, Ayako Tamura, Ryosuke Nakamura, and Tsuneo Matsunaga, "Animal detection using thermal images and its required observation conditions," *Remote Sensing*, vol. 10, no. 7, 2018.

[37] Seunghyeon Lee, Youngkeun Song, and Sung-Ho Kil, "Feasibility analyses of real-time detection of wildlife using uav-derived thermal and rgb images," *Remote Sensing*, vol. 13, no. 11, 2021.

[38] Jacques Charles Marais, "Automated elephant detection and classification from aerial infrared and colour images using deep learning," M.S. thesis, Stellenbosch University, Mar 2018.

[39] Aduen Benjumea, Izzeddin Teeti, Fabio Cuzzolin, and Andrew Bradley, "Yolo-z: Improving small object detection in yolov5 for autonomous vehicles," 11 2021.

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," 2015.

[41] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia, "Path aggregation network for instance segmentation," *arXiv*, 2018.

[42] Mingxing Tan, Ruoming Pang, and Quoc V. Le, "Efficientdet: Scalable and efficient object detection," 2019.

[43] Bharat Singh, Mahyar Najibi, Abhishek Sharma, and Larry S. Davis, "Scale normalized image pyramids with autofocus for object detection," 2021.

[44] Elizabeth Bondi, Raghav Jain, Palash Aggrawal, Saket Anand, Robert Hannaford, Ashish Kapoor, Jim Piavis, Shital Shah, Lucas Joppa, Bistra Dilkina, and Milind Tambe, "Birdsai: A dataset for detection and tracking in aerial thermal infrared videos," in *WACV*, 2020.

[45] Priya Dwivedi, "Yolov5 compared to faster rcnn. who wins?," https://towardsdatascience.com/yolov5-compared-to-faster-rcnn-who-wins-a771cd6c9fb4 (Accessed: April 2022).

[46] Facebook AI, "Pytorch documentation¶," https://pytorch.org/docs/stable/index.html (Accessed: May 2022).

[47] Renjie Xu, Haifeng Lin, Kangjie Lu, Lin Cao, and Yunfei Liu, "A forest fire detection system based on ensemble learning," *Forests*, vol. 12, pp. 217, 02 2021.

[48] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh, "Cspnet: A new backbone that can enhance learning capability of cnn," *arXiv*, 2019.

[49] Jonathan Hui, "Understanding feature pyramid networks for object detection (fpn)," https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c (Accessed: March 2022).

[50] Erin Moreland, "A dataset for machine learning algorithm development," https://www.fisheries.noaa.gov/inport/item/63322 (Accessed: May 2022).

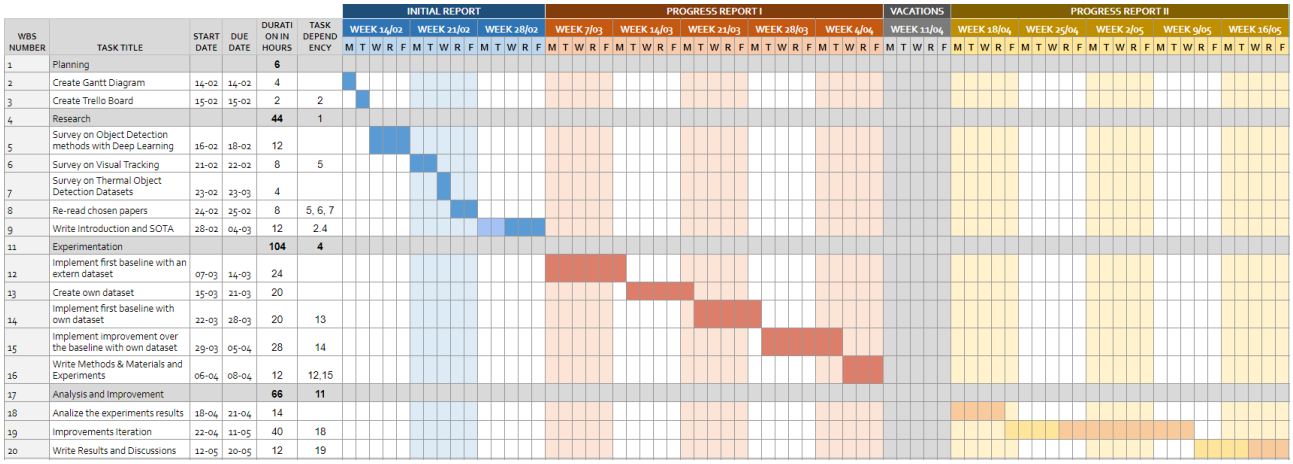
[51] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.

[52] Elizabeth Bondi, Debadeepta Dey, Ashish Kapoor, Jim Piavis, Shital Shah, Fei Fang, Bistra Dilkina, Robert Hannaford, Arvind Iyer, Lucas Joppa, and Milind Tambe, "Airsim-w: A simulation environment for wildlife conservation with uavs," in *In COMPASS '18: ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS), June 20–22, 2018,*, Menlo Park and San Jose, CA, USA. ACM, New York, NY, USA, 2018.

[53] documentation Ultralytics, "Tips for best training results," https://docs.ultralytics.com/tutorials/training-tips-best-results/ (Accessed: June 2022).

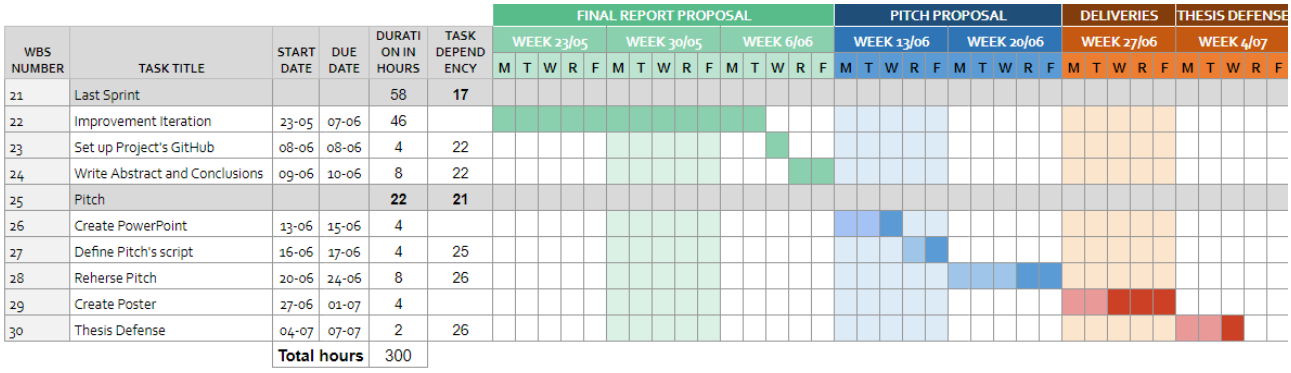
A PLANNING: GANTT DIAGRAM

The project executed the Gantt's scheduling defined at Figures A.1, A.2



WBS NUMBER	TASK TITLE	START DATE	DUE DATE	DURATION IN HOURS	TASK DEPENDENCY
1	Planning			**6**	
2	Create Gantt Diagram	14-02	14-02	4	
3	Create Trello Board	15-02	15-02	2	2
4	Research			**44**	1
5	Survey on Object Detection methods with Deep Learning	16-02	18-02	12	
6	Survey on Visual Tracking	21-02	22-02	8	5
7	Survey on Thermal Object Detection Datasets	23-02	23-03	4	
8	Re-read chosen papers	24-02	25-02	8	5, 6, 7
9	Write Introduction and SOTA	28-02	04-03	12	2.4
11	Experimentation			**104**	**4**
12	Implement first baseline with an extern dataset	07-03	14-03	24	
13	Create own dataset	15-03	21-03	20	
14	Implement first baseline with own dataset	22-03	28-03	20	13
15	Implement improvement over the baseline with own dataset	29-03	05-04	28	14
16	Write Methods & Materials and Experiments	06-04	08-04	12	12,15
17	Analysis and Improvement			**66**	**11**
18	Analize the experiments results	18-04	21-04	14	
19	Improvements Iteration	22-04	11-05	40	18
20	Write Results and Discussions	12-05	20-05	12	19

Fig. A.1: First part of the Gantt's Diagram. This section tackles from the beginning of the project to the Progress Report II



WBS NUMBER	TASK TITLE	START DATE	DUE DATE	DURATION IN HOURS	TASK DEPENDENCY
21	Last Sprint			58	**17**
22	Improvement Iteration	23-05	07-06	46	
23	Set up Project's GitHub	08-06	08-06	4	22
24	Write Abstract and Conclusions	09-06	10-06	8	22
25	Pitch			**22**	**21**
26	Create PowerPoint	13-06	15-06	4	
27	Define Pitch's script	16-06	17-06	4	25
28	Reherse Pitch	20-06	24-06	8	26
29	Create Poster	27-06	01-07	4	
30	Thesis Defense	04-07	07-07	2	26
		Total hours		300	

Fig. A.2: Second part of the Gantt's Diagram. This section tackles from the Final Report Proposal to the Thesis Defense

B BIRDSAI: INSTANCES PER CLASS

In this section, we can find BIRDSAI's dataset instances per class. The figures show different subsets used during the experiments:

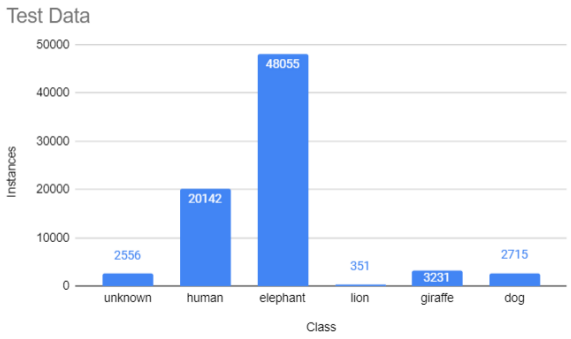


Fig. B.1: Test Data instances per class. This test set is composed of only real-world sequences and is the one proposed by the authors. However, it is unbalanced and contains very few instances of the classes unknown, lion, giraffe and dog.

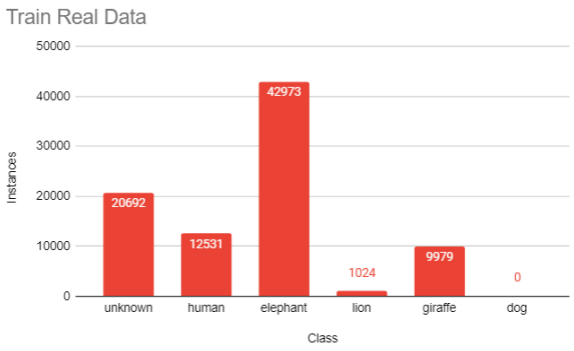


Fig. B.2: Real Training Data instances per class. This training set is composed of only real-world sequences and is the one proposed by the authors. However, it is unbalanced and contains very few instances of the class lion and none of the dog class.

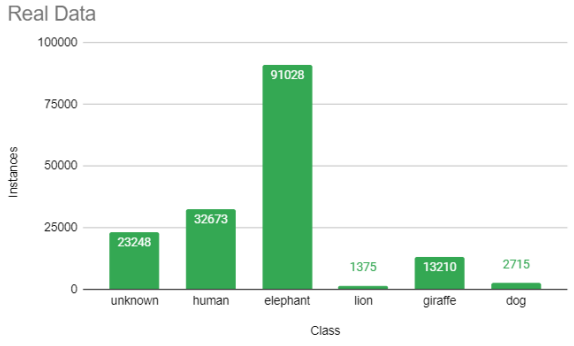


Fig. B.3: This graphic showcase the instances per class of the real-world data. As previously stated, it is class-unbalanced.

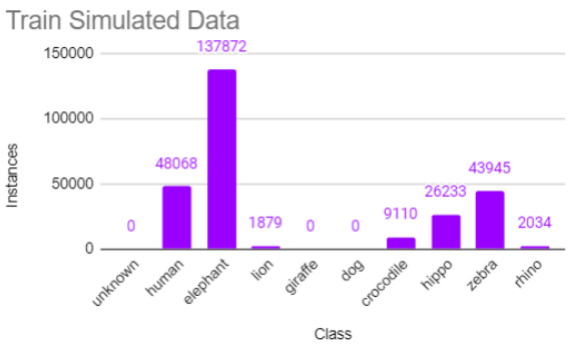


Fig. B.4: Simulated Training Data instances per class. This training set is composed of only simulated sequences. However, contains very few instances of the underrepresented classes in the real-world data. The simulation adds classes that are not seen in the real-world data.

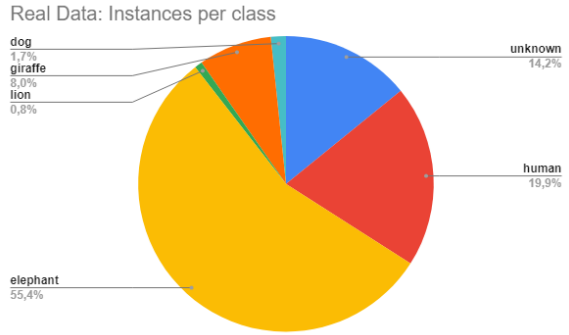


Fig. B.5: This graphic showcase the instances per class of the real data.

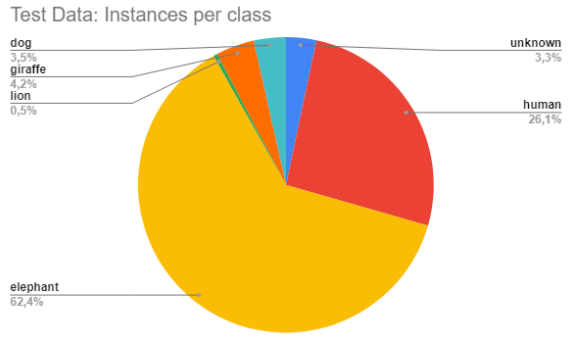


Fig. B.6: This graphic showcase the instances per class of the original test data.

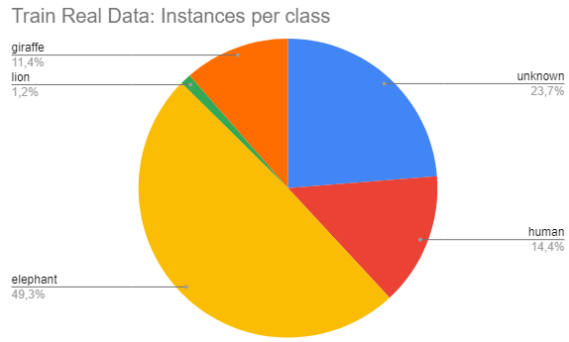


Fig. B.7: This graphic showcase the instances per class of the original train real data.

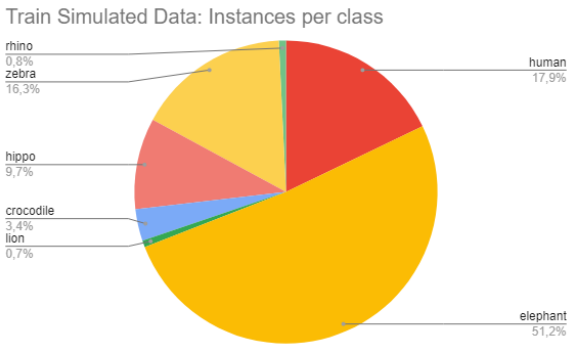


Fig. B.8: This graphic showcase the instances per class of the simulated data

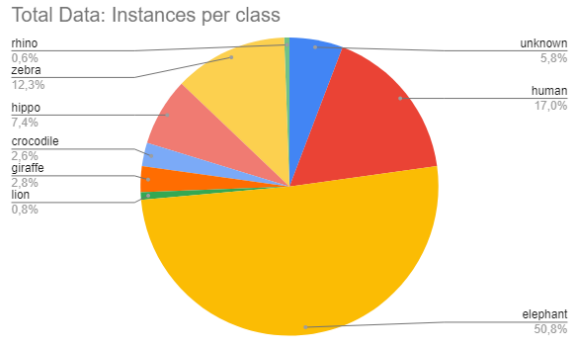


Fig. B.9: This graphic showcase the instances per class of the total data.