Marshallplan Final Research Paper

A Comparison of Convolutional Neural Network Architectures for Waterfowl Species Detection and Classification

Author

Mohammad Sadoun

Spatial Information Management

Carinthia University of Applied Sciences

School of Engineering & IT

Department of Geoinformation & Environmental Technologies

Supervisors

FH-Prof. Mag. Dr. MSc. MAS Gernot Paulus, School of Spatial Information Management, Carinthia University of Applied Sciences

Christopher D. Lippitt, Ph.D. Department of Geography and Environmental Studies & Center for the Advancement of Spatial Informatics Research and Education, University of New Mexico

FH-PROF. Dr.-ING. Karl-Heinrich Anders, School of Spatial Information Management, Carinthia University of Applied Sciences

Villach, 08/09/2020

Abstract

Wildlife surveying is an important task that improves understanding how species live and distribute and therefore, improving methods to better understand and observe wildlife are in need. Ground and manned aircraft-based surveying are traditional methods that are performed to achieve such goals. However, these methods have disadvantages regarding the time consumption, potential risks on survivors, and isolated area reachability.

Uninhabited aerial system (UAS) brought advantages regarding risk and special coverage scale but with more lab time required to manually analyse imagery. Thus, we need to mitigate human intervention while maintaining satisfactory results by using machine methods. Different machine-based methods such as spectral-based analysis, supervised multi spectral classification and template matching are used to automate the process but are limited of their abilities to capture targeted species in environments where surroundings can be confused with target objects. In this project, we compare the performance of different architectures of Convolutional Neural Networks (CNN's) to propose an alternative method that automates the process of waterfowl species detection and classification.

Our dataset consists of 13 images each of size 5472x3648 and in these images LabelBox was used by 13 experts from United States Fish and Wildlife Service (USFWS) to label waterfowls. The waterfowl dataset includes three species (duck, goose, and crane) and eight sub-species (American wigeon, Canadian goose, gadwall, mallard, northern pintail, sand-hill crane and "Other" [mostly duck]). Thus, we test the ability of CNN's to detect the targeted objects on three levels (waterfowl, species, and sub-species).

We investigate the pre-processing steps that are necessary to be implemented on our dataset such as image cropping, redundant label removal, and label format standardization. We implemented three CNN architectures (YOLO, Retinanet and Faster R-CNN). CNNs recorded an average of 79.47% accuracy in the task of waterfowl detection. As for species classification, the CNN's recorded averages of 71.3%, 54.6 and 66.6 for duck, goose, and crane, respectively. Also, we found major performance degradation on the sub-species level to less than 30%. We discovered that results of CNN do not have a common denominator because it can detect non-waterfowl objects which have no reference in the ground truth. Faster R-CNN was found to detect much more non-waterfowl objects than YOLO and Retinanet.

Finally, we also analyzed the effect rough surroundings such as shadows and plants were CNN's were more likely to produce false negative prediction. Also, CNNs' ability to detect decreases as the waterfowl population density in the image increases.

Acknowledgement

In this regard, I would like to thank everyone who helped and supported us during the past two semesters. All thanks and appreciation to my supervisors, FH-Prof. Mag. Dr. MSc. MAS Gernot Paulus, Christopher D. Lippitt, Ph.D. and FH-PROF. Dr.-ING. Karl-Heinrich Anders for guiding me through this project by simplifying and explaining the concepts. I express my sincerest gratitude to the Marshallplan Foundation for allowing me to do this project in the University of New Mexico (UNM). Many thanks the staff and colleagues in UNM especially Nathan Holscher, Linda Melville, Phillip Gill, Jesse Sprague, Jillian Rutherford, and Rowan Converse for their outstanding support during the time of the project. Special thanks to Mr. Thomas Schneider and Ms. Alexandra Maurer for guiding me through the VISA application and the Marshallplan process. Finally, sincerest regards and fine wishes to my families and friends who had encouraged me through the journey of achieving this endeavour.

Table of Contents

Abstract 2

1. Introduction 5

   1.1 Motivation 5

   1.2 Problem Definition and Research Questions 5

   1.3 Methodology 6

   1.4 Expected results 7

   1.5 Structure of the Thesis 7

2. State of the Art and Literature Review 7

   2.1. Traditional surveying methods 7

   2.2. UAS Surveying 9

   2.3. Automated wildlife detection from UAS imagery 10

   2.4. Machine Learning 14

   2.5. Convolutional Neural Networks 17

   2.6. Related work in bird detection using Convolutional Neural Networks 19

3. Approach 20

   3.1. General project breakdown structure 20

   3.2. USFWF Dataset 21

   3.3. Choice of CNN Architectures 24

   3.4. Detailed specifications of chosen CNN's 25

   3.5. Workflow for the evaluation 29

4. Implementation 31

   4.1. Pre-processing 31

   4.2. Training 33

5. Results and error analysis 39

6. Discussion 46

7. Conclusions 49

8. Future work 49

References 51

1. Introduction

In this chapter, we discuss the motivation behind the CNN methodology and what advantages it has over traditional methods. We define our problem and research questions then We list the expected results.

1.1 Motivation

Waterfowl population recognition and classification have traditionally been undertaken by a combination of ground-based and manned aircraft surveys. Manned aircraft surveys indeed brought advantages when searching in large areas because of the large area scale at which it can cover (relative to ground-based) and the development of ultra-high-resolution and thermal cameras. However, manned aircraft surveys are expensive and can cause stress to wildlife (Wilson et al. 1991). UAS have been used to successfully survey a variety of bird species worldwide with much lower costs and risks. However, one factor hindering the adoption of UAS surveying is the additional human hours required in the lab to manually identify animals in the captured UAS imagery, compared with counts in the field (Linchant et al. 2015). Different automation techniques have been used in the process of waterfowl recognition with accuracy comparable to manual image counts such as spectral based analysis, including spectral thresholding (Laliberte and Ripple 2003), supervised classification (Grenzdörffer 2013), and template matching (Abd-Elrahman et al. 2005). However, these methods are limited in that they require animals to be highly spectrally separable from their environments, which hinders applications in heterogeneous environments in the study of species with cryptic colouration, or with image sets of varying brightness due to camera performance or weather conditions (Linchant et al. 2015, Chabot and Francis 2016). Thus, Machine Learning (ML) can be used to try to overcome the time consumption and spectral separation issues as the ML field has shown significant improvement with detection and classification tasks especially using CNN (Chen et al. 2012).

1.2 Problem Definition and Research Questions

The field of ML is the study of using computers to perform specific tasks without explicit instructions by learning from data. Several ML models can be applied to perform image classification such as support vector machine (SVM) and Key Nearest Neighbour (kNN). However, CNN's are chosen to be employed in this project because of their leading accuracy performances, non-linearity, and the ability to increase model complexity (adding convolutional layers for deeper feature extraction). A CNN is a class of ML that is applied for imagery segmentation and classification. CNN's have now been studied and matured to be utilized in many different types of automation processes in several application domains, for example, but not limited to crowd counting, object detection, face attributes recognition and geo-localization (Howard et al. 2017, Girshick et al. 2014, van Gemert et al. 2014) (Chen et al. 2012). These automation applications have major advantages such as cost reduction and time-saving that make the automated task more efficient. However, different architectures of CNN's are currently available to accomplish such goals.

This project aims to setup a comparative study of different CNN architectures for automated waterfowl detection, classification and counting by answering the following questions:

- What are the necessary preprocessing steps that should be done on the UAS waterfowl imagery for CNN to perform classification and counting?
- How does the prediction of CNN's change across the three different models (waterfowl, species, and sub-species)?
- How accurate can state of the art CNN's perform on UAS waterfowl imagery datasets?

1.3 Methodology

We want to utilize state of the art CNN architectures and evaluate how accurate they can detect and classify different waterfowl species on three levels namely (waterfowl count, species, and sub-species). The implementation of the selected CNN architectures will run on an already existing training set of labelled and high-resolution waterfowl images taken by UAS to enable comparison of various architecture's ability to recognize and classify different waterfowl species. The training set was collected using a crowdsourced image labelling service called LabelBox and consists of 13 images each of size 5472x3648 with total label count of 18469 labels. The survey mission took place in Bosque del Apache Wildlife Refuge in New Mexico using DJI Mavic drone in November 2018. The major goal of this empirical research project is to obtain the CNN architecture that performs best results defined as the number of correct classifications and counts (relative to the ground truth) and implements it with a special focus on the application domain of wildlife identification to produce an automated approach for identifying different waterfowl classes and in turn, feed accurate information about the counts and distribution to the specialists for more efficient decision making process.

Different CNN architectures (e.g., CifarNet, MobilNets, AlexNet, GoogLeNet, YOLO, etc.) (Howard et al. 2017, Zha et al. 2015) have accomplished significant performance improvements in many application domains such as object labelling and classification, event detection for safety systems, obstacle avoidance in autonomous driving and identity checking (Zha et al. 2015). Each architecture differs by certain aspects and features such as the number of intermediary layers that they have, number and size of the kernels used to be convolved with the main image and feature maps, error calculation methods, and activation functions. However, if we want to change the application but still use these CNNs, a process of training should be done. CNN should be fed with a training set that contains both the desired input and output (question and right answers). The CNN then performs error calculation for its predictions and as the CNN crawls over the training set, the error of prediction gets reduced. We want to compare different CNN architectures in terms of their performance in the recognition and classification of waterfowl species in support of the USFWS.

1.4 Expected results

- build a comprehensive evaluation study to compare and judge the performance of different CNN architectures based on the accuracy of classification and counting and being able to discover the necessary preprocessing steps for the waterfowl imagery dataset.

- To estimate the potential of using CNN's in waterfowl surveying.

- To build an implementation framework that other waterfowl datasets can utilize to reproduce prediction results

1.5 Structure of the Thesis

The thesis structure is as follows:

First, we present the literature review for traditional surveying methods, what improvements UAS brought into the field and where does the machine-based surveying stand. After that, the approach goes into the details of data acquisition, CNN's, and workflow of the implementation. Then, we present the project setup and implementation and we discuss the results. Finally, we present the conclusion of our findings and we talk about potential future work.

2. State of the Art and Literature Review

In this chapter, we review traditional surveying methods from literature where they differ and what are the advantages and disadvantages of each method. Then, we discuss surveying using UAS and what benefits can be achieved with it for both manual counts from imagery and machine-based from the same imagery. Finally, we review ML in image recognition and justify our choice of using CNN.

2.1. Traditional surveying methods

surveying of water birds emerged as an important method of tracking the changes that happen in the wildlife ecosystem and has been used to estimate populations.

Traditional surveying of waterbirds refers to ground and aerial surveying. Ground surveying is done by walking or on vehicle and manually count the number of target species whereas aerial surveying is done by flying an aircraft above the surveying area on an altitude where the surveyor can see and count target species.

One advantage of aerial surveying of water birds is the spatial scale at which it can be performed. Entire floodplain wetland systems can be surveyed which means that data can be collected at a scale similar to that at which a river system is managed. Aerial surveying of water birds can simultaneously collect data on a range of species, but the efficiency of areal methods has been investigated (Kingsford et al. 1999). However, more needs to be known about the accuracy and precision of multispecies surveys such as the limitations and the ability to distinguish between different kinds of species, to determine their usefulness compared with ground-based methods.

In (Kingsford et al. 1999), water birds living around Lake Altibouka in north-western New South Wales were surveyed using a Cessna 206 aircraft operated at a height of 30m, the lake covers 300 ha with a long axis of 3.3 km and a short axis of 0.8 km. It should be mentioned here that this lake was chosen because of its relatively small size and the absence of vegetation obstacles which makes it possible to effectively count water birds from the ground also thus, better comparison conditions.

The data was collected for both areal and ground surveys, into four classes (< 10 / 11 to 100 / 101 to 1000 / > 1000) species per survey. This was done to avoid the problem of large counts with high variance dominating counts. It should be noted that it took around 2.3 min to fly around the lake and between two and seven hours to do ground counts of the lake.

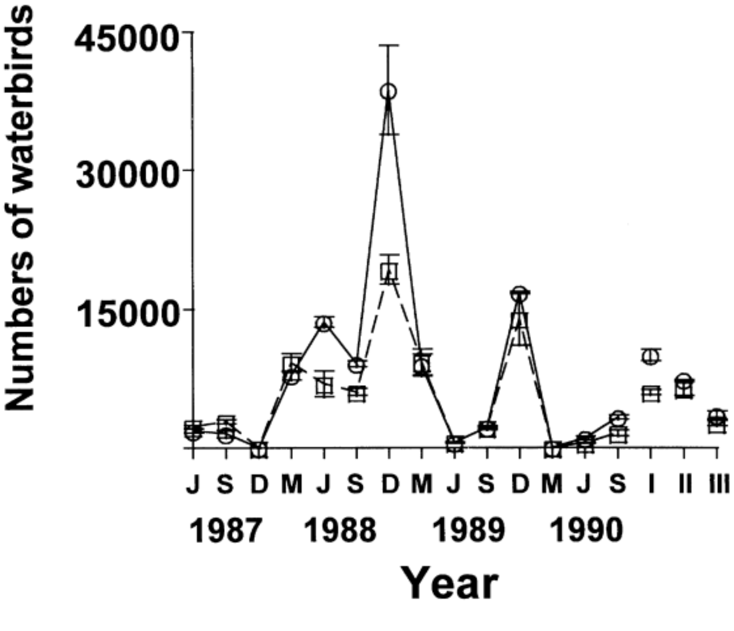


Figure1: Numbers of water birds counted on Lake Altibouka per field trip during 15 trips, (dotted are areal and continuous are ground-based), Adopted from “Aerial survey of water birds on wetlands as a measure of river and floodplain health” by (R. T. Kingsford. 1999) in Freshwater Biology (1999) 41, 425-438

It was clear after collecting the results that the number of species distinguished in the ground surveys was higher: 54 species of water birds could be differentiated during ground counts compared with 45 during aerial survey counts. As seen in figure1, ground counts were slightly more precise than aerial counts and aerial counts and ground counts for species which occurred in numbers of less than 10 were similarly imprecise. For accuracy, both methods recorder the same standard error (SE) except for the case where the abundance of birds was between 100 and 1000 where aerial count recorded SE of 0.02 and ground count recorder 0.2.

Aerial surveys can be used to collect data on waterbird population for up to 50 different species (Kingsford et al. 1999). Because the method is quick and inexpensive compared with ground counts, large areas may be surveyed, providing information at a landscape scale. More than one aerial survey of the same birds on a wetland allows estimation of precision. One of their most significant advantages is that the results of aerial surveys may be applied to the management of an entire river and its floodplain. Such information is more easily incorporated by river managers who tend to manage at the scale of the catchment. The more indices we have of river and floodplain health at the catchment scale, the more likely it is that results of studies by ecologists will be implemented by river managers.

However, it was shown by (Wilson et al. 1991) that wild birds can be disturbed as aircraft approach. Also, (Sasse 2003) showed that Ninety-one people died while participating in wildlife research and management activities between 1937 and 2000 and Aviation accidents, drowning, car, and truck accidents were the most common causes of death for aircraft surveyors.

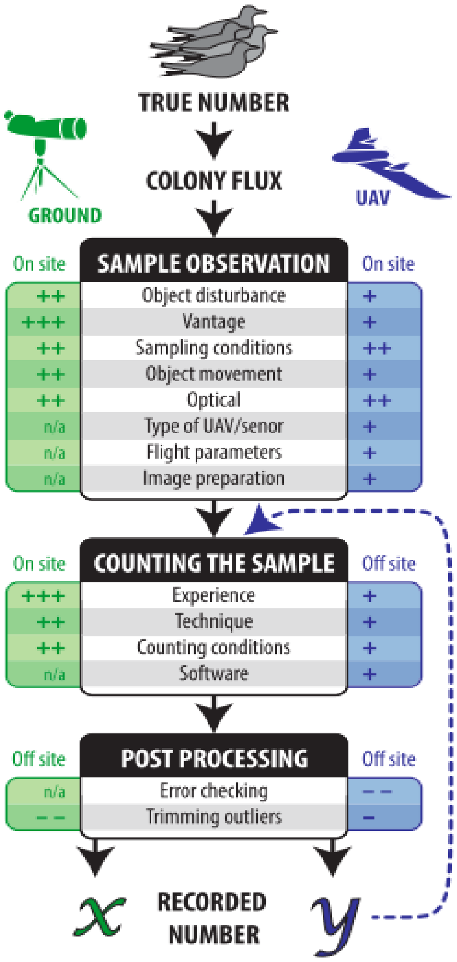
2.2. UAS Surveying

UAS equipped with high resolution multispectral sensors offer many of the advantages of manned-aircraft surveys at lower cost and lower risk in terms of operation. Bird population counts using UAS imagery have lower variance compared with traditional ground-based counts, and precision up to an order of an acceptable magnitude of ± (5 to 10) percent.

It was demonstrated that the precision (defined as the variance between replicated counts by different counters attempting to count the same sample) of population counts of waterbirds in both tropical and polar environments can be improved using UAV technology compared to ground counts where UAV-driven counts had significantly lower variance within colonies than ground counts for all species surveyed (Hodgson et al. 2016).

It was also found in (Hodgson et al. 2016) that UAV-driven counts are consistently similar to or significantly larger than ground counts because of the downward-facing perspective of UAV imagery that reduces the likelihood of missed counts due to topography and birds obscuring the counters' line of sight which states that the surroundings can greatly affect the accuracy of UAS count results. Additionally, still, imagery from UAVs presents the option of separating the count area into manageable subsets and completing counts in multiple sittings. However, the transition from traditional to new UAV-based monitoring methods requires careful consideration, particularly in terms of maintaining the relevance of historical data that has been collected at a substantial time and financial cost. Figure2 shows measurements sources of variance when estimating the number of subjects in a faunal aggregation using a traditional ground (green) or UAS (blue) counting technique ((+,-)minor, (++,--)moderate, (+++,---)major)

Moreover, it was found that no geese were observed flushing or leaving during drone surveys flown at 183 m altitude (Chabot and Bird 2012). The benefits of UAS for collection of data on surface-nesting birds are compelling, including perceived reductions in impact and greater spatial coverage and frequency compared to ground surveying. Therefore, UAS provide an alternative means of collecting important demographic and environmental data. For surface-nesting birds, UAS technology can provide a more accurate method of collecting population data because of its ability to take images of colonies, which can be counted carefully in the lab and compared through time, therefore reducing the uncertainty of estimates in traditional observer counts (Hodgson et al. 2016, van Gemert et al. 2014).



On site	SAMPLE OBSERVATION	On site
++	Object disturbance	+
+++	Vantage	+
++	Sampling conditions	++
++	Object movement	+
++	Optical	++
n/a	Type of UAV/senor	+
n/a	Flight parameters	+
n/a	Image preparation	+

On site	COUNTING THE SAMPLE	Off site
+++	Experience	+
++	Technique	+
++	Counting conditions	+
n/a	Software	+

Off site	POST PROCESSING	Off site
n/a	Error checking	– –
– –	Trimming outliers	–

Figure2: Measurements of sources of variance when estimating the number of subjects in a faunal aggregation using a traditional ground (green) or UAV (blue) counting technique. Adaptet from" Precision wildlife monitoring using unmanned aerial vehicles" by (Hodgson, J. C. et al 2016). In Precision wildlife monitoring using unmanned aerial vehicles. Scientific Reports, 6(1).

2.3. Automated wildlife detection from UAS imagery

Image processing techniques to extract elements from an image that match a target object can broadly be divided into two major classes namely area-based matching and feature-based matching. Feature matching algorithms make use of attributes such as colour, and texture. Features in one image are compared with potential corresponding features in the other image. A pair of features with similar attributes is accepted as a match. Area-based image matching methods use statistical similarity measures to compare the spectral composition of an image of a target object with the same size area in a moving window across another image. One such measure, normalized cross-correlation, is widely used for identifying control points and common features in overlapped imagery.

Although bird surveys conducted using UAS Imagery can be more accurate than the traditional methods (ground-based and manned aircraft), they can be more time consuming if images would be analysed manually (Chabot and Francis 2016).

In (Laliberte and Ripple 2003), spectral-based analysis (in this case, changes in brightness value per unit distance in any part of an image see Figure3) on black-and-white and coloured aerial images was applied with variety of resolutions containing different wildlife species so that the methods could be tested under various condition. The image analysis programs used were ERDAS Imagine and ImageToo.

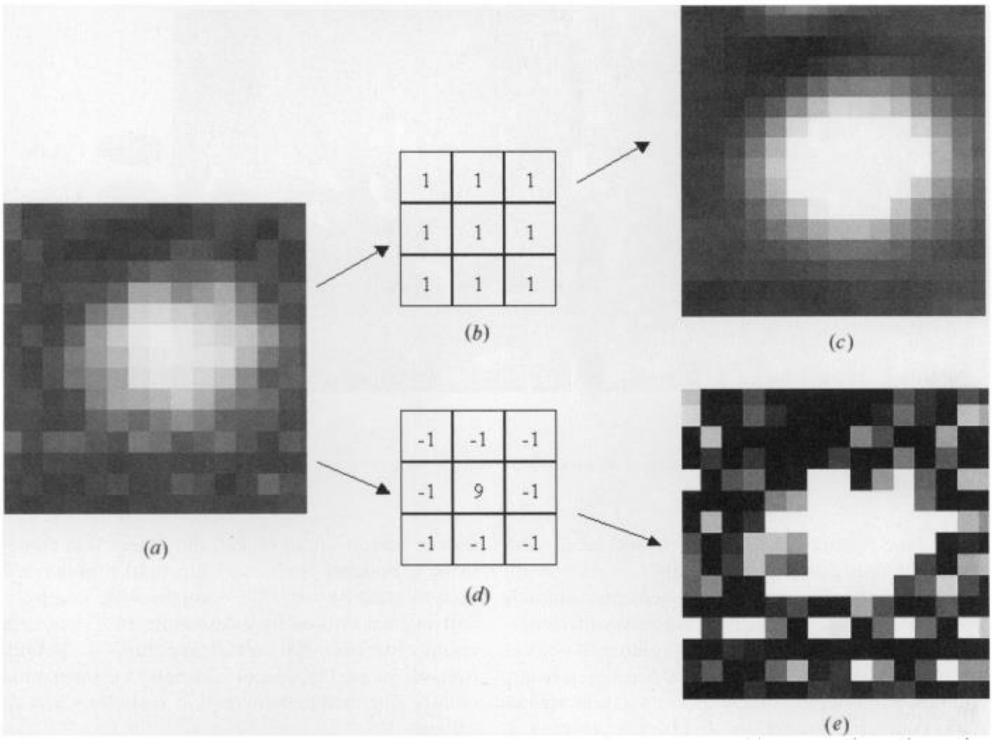


Figure3: Applying Lowpass filter (b) and High pass filter (d) to an image, Adopted from "Automated wildlife counts from remotely sensed imagery" by (Laliberte, A. S., and Ripple, W. J. 2003) In Wildlife Society Bulletin, 362–371.

Authors report that the results were promising even though the number of animals would be greater in conditions where objects are more spectrally separable (109 to 299 in the images used in the study). Figure4 shows the high correlation between manual from imagery and computer counts that was encouraging and demonstrated that this technique worked well.

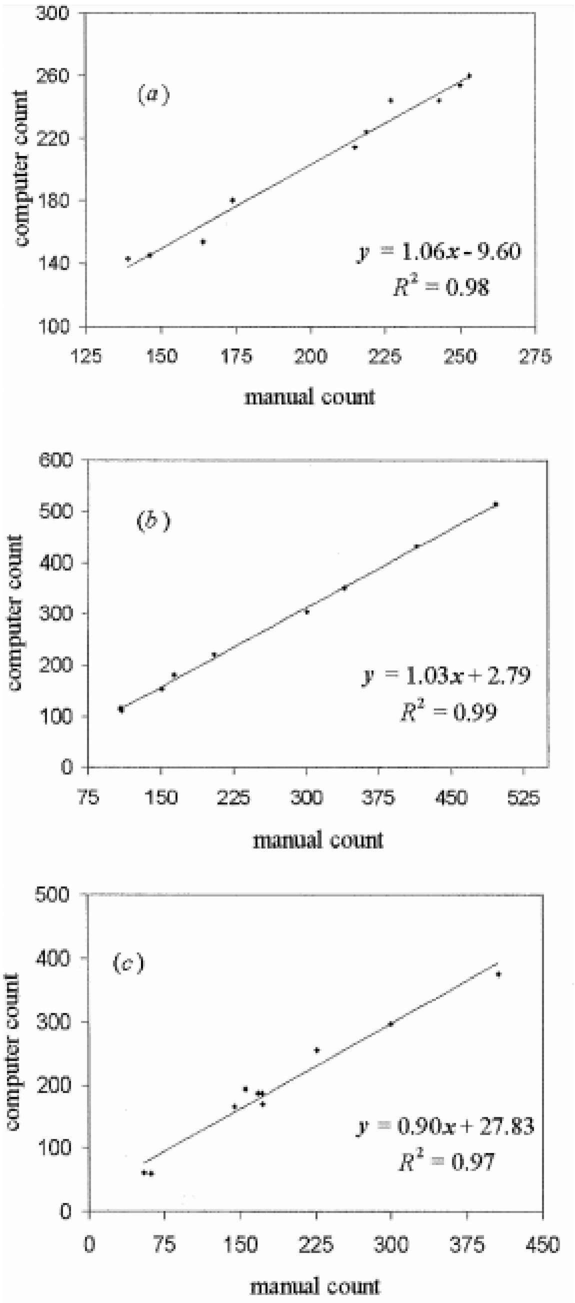


Figure4: Graphs of manual versus computer counts for 10 image subsets for 3 species. (a) snow geese, (b) Canada geese, (c) caribou all with the corresponding regression equations and the correlation. Adopted from Adapted from "Automated wildlife counts from remotely sensed imagery" by (Laliberte, A. S., and Ripple, W. J. 2003) In Wildlife Society Bulletin, 362–371.

Another spectral-based analysis introduced in (Grenzdorffer 2013) where ArcGIS 10 software was employed to perform supervised multi spectral classification with a total of 7 classes was used to perform supervised classification of gulls and an accuracy of 97.6% was verified. However, this methodology does not necessarily apply to other bird species, as the examined gulls provide very good contrast to its surroundings.

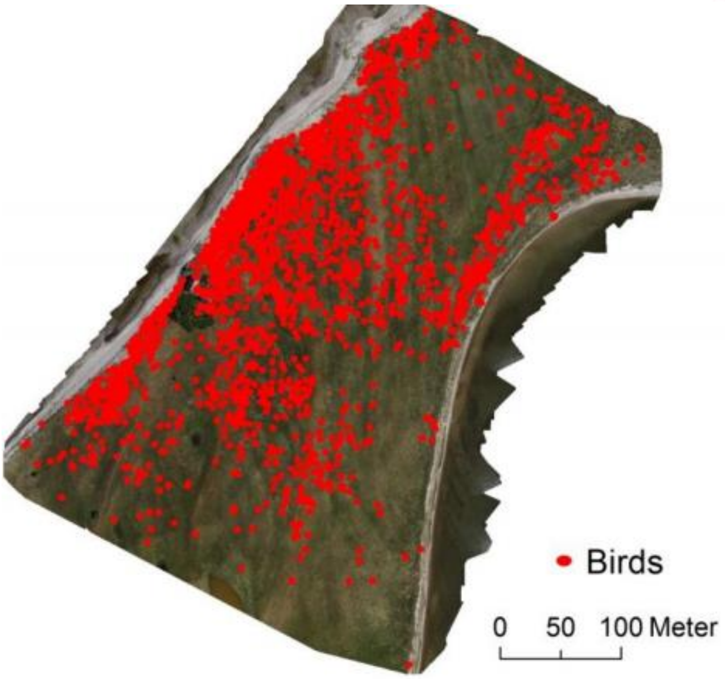


Figure5: Identified gulls objects (red dots) from UAS aerial survey of 25.5.2012 on the birds reserve island Langenwerder, Adopted from "UAS-based automatic bird count of a common gull colony" by (G. J. Grenzdörffer 2013) in International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 1, W2.

In the study of species with cryptic coloration, or with image sets of varying brightness due to camera performance or weather conditions (Linchant et al. 2015, Chabot and Francis 2016).

Moreover, a multi-stage pattern recognition algorithm (by means of template matching) was developed to identify individual birds using images captured by UAS (Abd-Elrahman et al. 2005). The developed pattern recognition algorithm for counting birds relies on a four-stage algorithm to enhance the overall obtained accuracy as follows: Normalize cross-correlation, Region grouping, Spectral Characteristics and Zero order shape moment. The algorithm performed with 94.02% to 96.42% accuracy.

However, the mentioned methods above are limited in that they require animals to be highly spectrally separable from their environments, which hinders applications in heterogenous environments, in the study of species with cryptic coloration, or with image sets of varying brightness due to camera performance or weather conditions

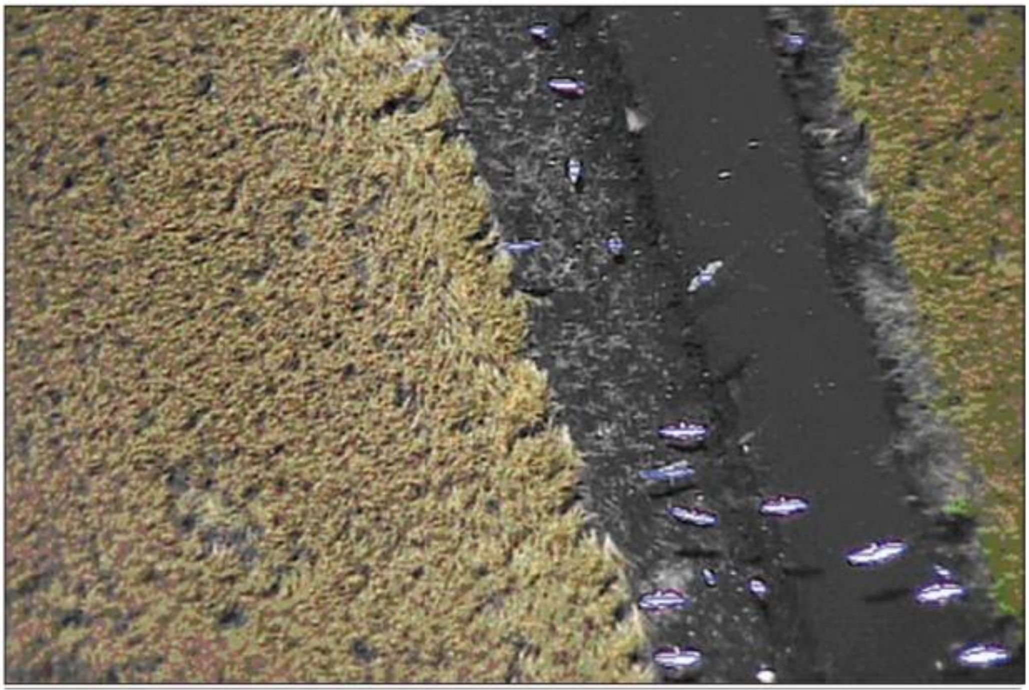


Figure6: Automatically identified individual birds are shown as blue polygons using template matching, Adopted from"Development of Pattern Recognition Algorithm for Automatic Bird Detection from Unmanned Aerial Vehicle Imagery" by (Abd-Elrahman, A., Pearlstine, L., and Percival, F. 2005) In Surveying and Land Information Science, 65(1), 37

2.4. Machine Learning

ML is a field that studies the building of computer systems that learn and improve by experience from data (Mitchell, T.M 2006). ML algorithms are mainly divided into two styles in terms of learning supervision.

The first is Supervised Learning, where the machine is given an input data that has known labels. The machine then goes through a learning process and continuously make predictions about in input data until it achieves a targeted level of accuracy. Examples od such algorithms are linear regression and neural networks (NN).

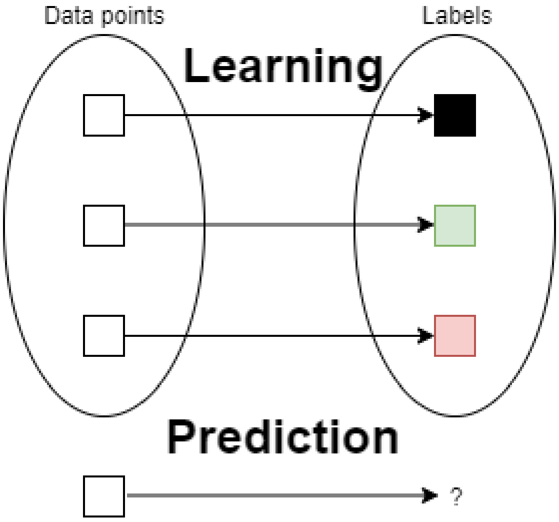


Figure7: A simplified diagram for supervised learning

The second style is Unsupervised Learning. In this style the input data are not labelled, and the machine rely on a mathematical operation to cluster the data an obtain general rules/similarities between data points. Examples of such algorithms are Apriori algorithm, K-Means and SVM.

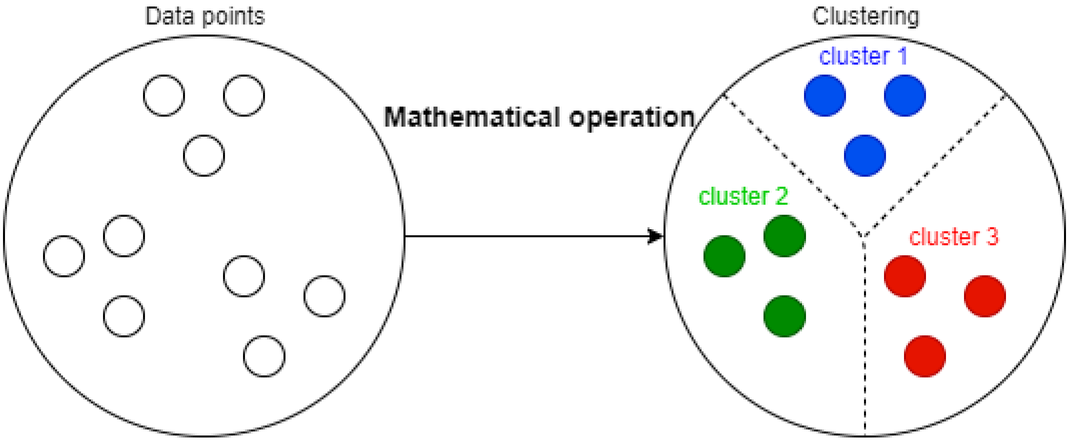


Figure8: A simplified diagram of unsupervised learning

One of the most famous concepts of supervised learning is deep learning (DL). DL is a sub-category of ML that mimics how the human brain works. It uses what is called Artificial Neural Networks (ANN) that consists of many layers that contain neurons (or nodes) all connected to form a web structure.

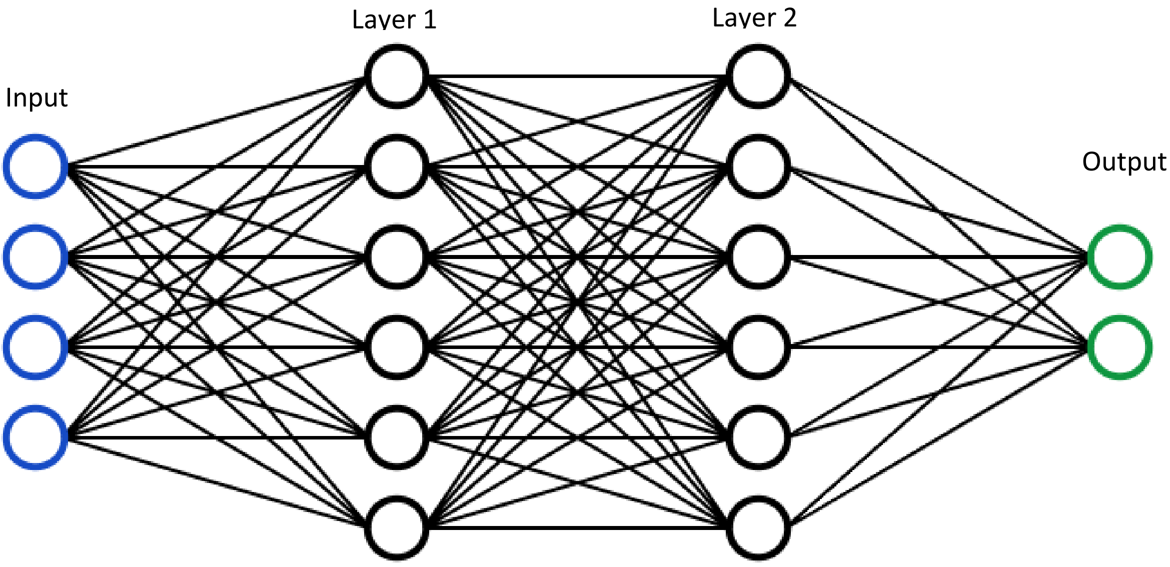


Figure9: A simplified diagram for a neural network

Each node transforms data by multiplying every value that inters the node with "bias/weight", which is a node-specific value of the node. Then the node sums all entered values. Then the node normalizes the output value by using an activation function. This process repeats until the nodes' weights are adjusted and the network achieves the target accuracy. The main task of the activation function is to provide nonlinearity to the process, which increases the networks' ability to capture complex patterns.

One of the forms of data that can be fed to an ML model are images. Several tasks can be achieved by employing ML algorithms in images such as object localization and classification, area segmentation, face recognition and action recognition. CNN, SVM and kNN are examples of ML algorithms that can be applied to perform object classification. In kNN, the algorithm relies on computing the distance between features (e.g. Euclidean distance) associated with target objects, then groups images that are close to each other as seen in figure 10.

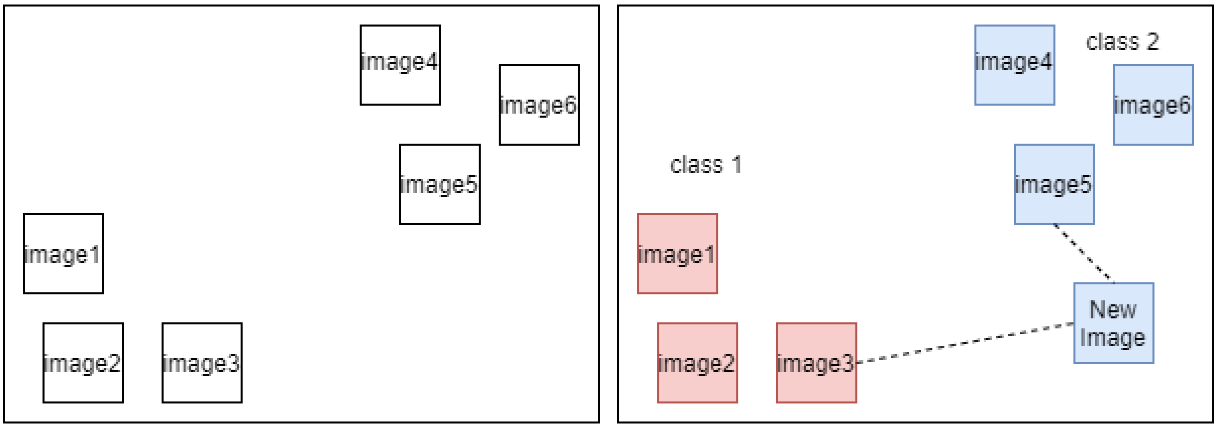


Figure10: simplified diagram of kNN function

The parameter k refers to number of closest neighbors considered for class assignment.

One of the disadvantages of k-NN is becomes limited with large data. This is due to huge cost of computation for distances between new data points and large training set. Moreover, the major reason why k-NN is not used in this project, is that k-NN can have a hard time separating high-dimensional data such as images, especially when we want to distinguish between images that contain birds that look fairly similar such as sub-species of duck.

SVM works in a different manner to k-NN, the algorithm assumes that there exists a hyperplane that separate the data such that each distinct group od data points can be clustered together.

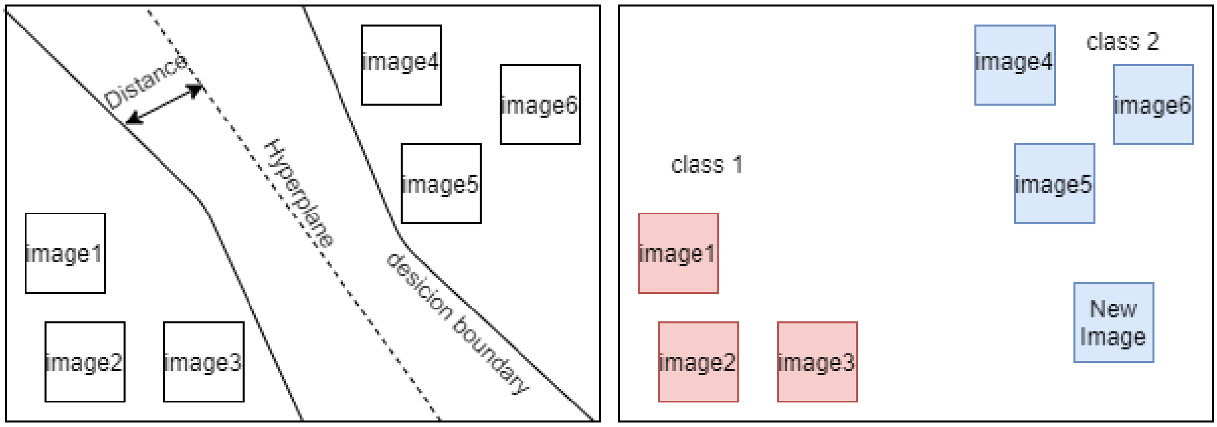


Figure11: simplified diagram of SVM function

Sometimes the data points cannot be separated by a hyperplane due to the nature of data distribution such as circular data. SVM use kernels to re-shape the data such that a new hyperplane exists that can separate the data points.

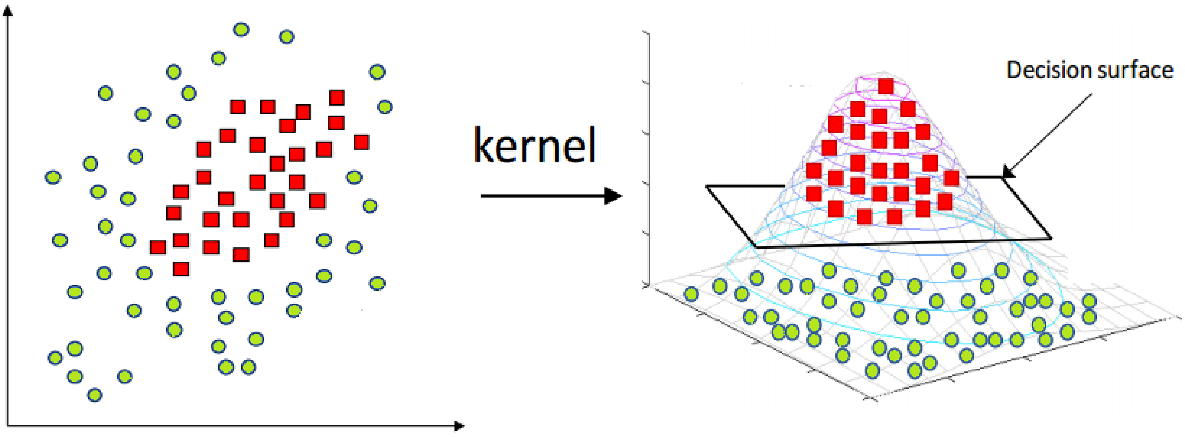


Figure12: kernel operation by Grace Zhang. November 2018. In "What is the kernel trick? Why is it important?". https://medium.com/@zxr.nju/what-is-the-kernel-trick-why-is-it-important-98a98db0961d

The main disadvantage SVM suffers from is that it assumes that data can be linearly separated. Even though, choosing the proper kernel function is not easy especially with high dimensional data such as images.

2.5. Convolutional Neural Networks

CNNs are just like regular neural networks which may be visualized as a group of neurons organized as in a cyclic graph. The main difference from a neural network is that a hidden layer neuron is only connected to a subset of neurons in the previous layer. Figure13 shows the basic architecture of a CNN.

CNNs are a widely used deep learning framework which was inspired by the visual cortex of animals. Initially it had been widely used for object recognition tasks but now it is being examined in other domains as well like object tracking, text detection and recognition, action recognition, scene labelling and many more (Aloysius N. and Geetha M. 2017).

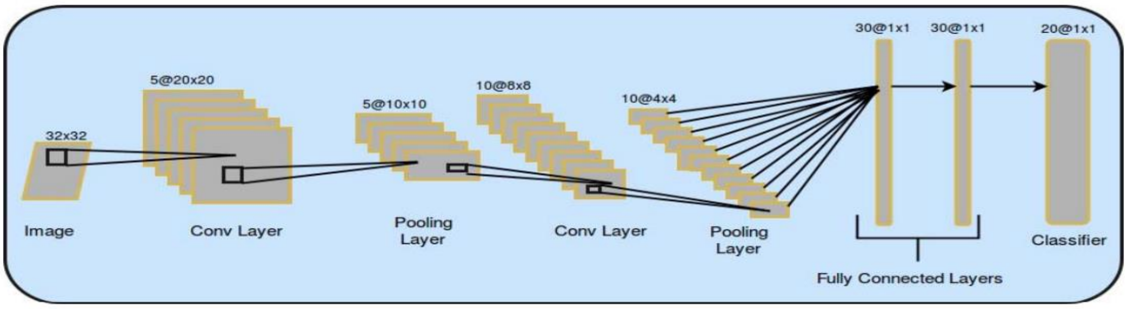
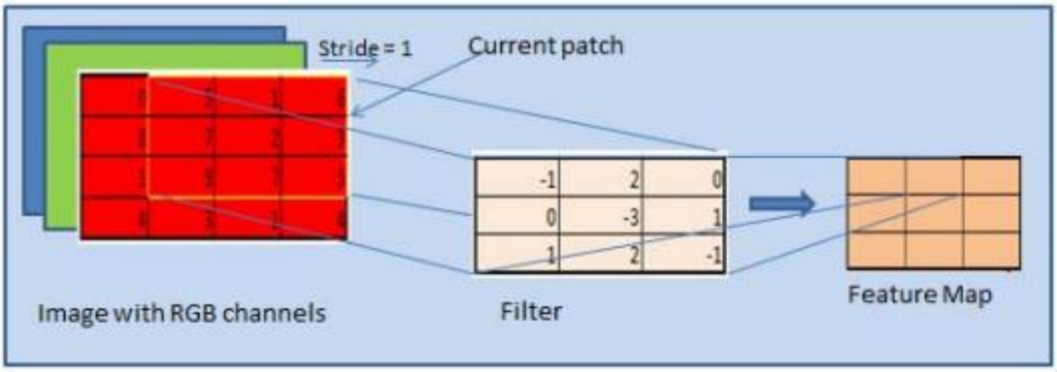


Figure13: Basic CNN architecture, Adopted from “A Review on Deep Convolutional Neural Networks” by (Aloysius, N. and Geetha, M., 2017) In International Conference on Communication and Signal Processing (ICCSP) (pp. 0588-0592). IEEE.

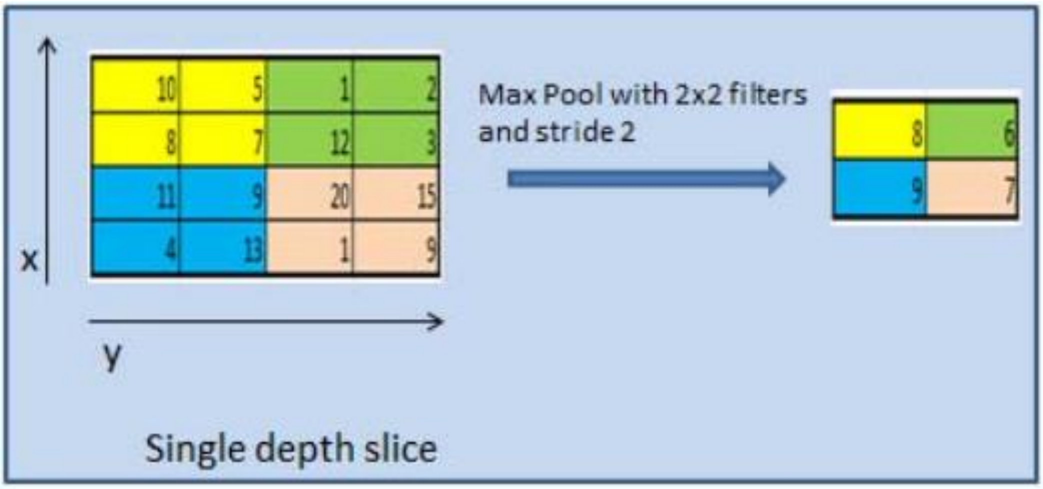
Convolutional Layer: This layer forms the basic unit of a CNN where most of the computation is involved. It is a set of feature maps with neurons arranged in it. The parameters of the layer are a set of filters (or kernels). These filters are convolved with the input image and the extracted feature maps from each stage. The parameters that control the size of the output volume are the depth (number of filters at a layer), stride (filter step) and padding (to control the size of output after convolution).



-1	2	0
0	-3	1
1	2	-1

Figure14: Convolution operation for R layer in an RGB image Adopted from "A Review on Deep Convolutional Neural Networks" by (Aloysius, N. and Geetha, M., 2017) In International Conference on Communication and Signal Processing (ICCSP) (pp. 0588-0592). IEEE.

Pooling Layer: pooling layers and the latter functions to reduce the spatial dimension of the activation maps (without loss of information as much as possible) and the number of parameters in the net and thus reducing the overall computational complexity. This controls the problem of overfitting. Some of the common pooling operations are max pooling, average pooling, stochastic pooling.



10	5	1	2
8	7	12	3
11	9	20	15
4	13	1	9

8	6
9	7

Figure15: Max-pooling operation Adopted from "A Review on Deep Convolutional Neural Networks" by (Aloysius, N. and Geetha, M., 2017) In International Conference on Communication and Signal Processing (ICCSP) (pp. 0588-0592). IEEE.

Fully Connected Layer: Neurons in this layer are fully connected to all neurons in the previous layer, as in a regular Neural Network. The neurons are one dimensional so there cannot be a conv layer after a fully connected layer.

Loss Layer: The last fully connected layer serves as the loss layer that computes the loss or error which is a penalty for discrepancy between desired and actual output. For predicting a single class out of K mutually exclusive classes Softmax loss is usually used. It maps the predictions to non-negative values and normalized to get probability distribution over classes.

Activation Functions: Activation functions are non-linarites that take on a pixel in the input, feature maps or fully connected neurons and do a mathematical operation on them. Many such functions exist such as Sigmoid, Tanh, ReLU and Leaky ReLU.

Many frameworks are available for deep learning, of which Google's **TensorFlow** is the latest and fast growing. With this, a single API can be used to distribute load between multiple nodes (CPUs or GPUs). This library is publicly available since November 2015. **Keras** is second fast-growing deep learning framework. This open source library written in Python can run on top of **TensorFlow** or **Theano**. **Theano** is an open source Python library for numerical computations and simplifies the process of writing deep learning models. Another framework is **Caffe**, developed by the Berkeley Vision and Learning Center (BVLC). It has many worked examples of deep learning, written in Python. Giving more importance to GPUs is the framework called Torch, having an underlying C/CUDA implementation. MATLAB's matconvnet and Torch's torch are also widely used frameworks for deep learning.

2.6. Related work in bird detection using Convolutional Neural Networks

In a recent study done in Korea (Hong, S.J. et. al. 2019), five different CNN architectures where employed for the purpose of bird detection namely (Faster R-CNN, R-FCN, SSD, Retinanet and YOLO) and evaluated by comparing their speed and accuracy. The accuracy of the detection was measured using the intersection of union IOU, defined as the ratio of intersection between the predicted box and the ground truth box. A threshold of 0.3 and 0.5 of IOU to determine the acceptability of the detection and the CNNs' performance was measured for both thresholds. The training data was 25,864 UAS images including 137,486 birds.

Table1: Evaluation results in (Hong, S.J. et. al. 2019) for CNN architectures

Architecture	Feature extractor	Interface Time (ms/photograph)	IOU:0.3	IOU:0.5
Faster R-CNN	Resnet 101	95	95.44	80.63
	Inception v.2	82	94.04	79.35
R-FCN	Resnet 101	87	94.86	79.83
Retinanet	Resnet 50	75	91.49	73.66
	Mobilenet v.1	57	85.01	66.01
SSD	Mobilenet v.2	23	85.9	54.87
YOLO v3	Darknet-53	41	91.8	58.53
YOLO v2	Darknet-19	34	90.99	56.8
	Tiny YOLO	21	88.23	54.22

It is shown that Faster R-CNN and R-FCN performed relatively the same in accuracy and speed. It is also noticed that Retinanet was slightly better than YOLO in accuracy when having IOU:0.3 but YOLO was almost twice as fast. However, if we looked at the IOU:0.5 accuracy, Retinanet performed much better than YOLO. SSD averaged the least IOU:0.3 and was better than YOLO in IOU:0.5 with Mobilenet v.1 feature extractor.

It should be mentioned here that the study evaluated the performance of different CNNs with how accurate they can detect a "bird" object without further classification. In our case, we have two additional levels of evaluation for species (duck, goose, and crane) and sub-species (American wigeon, Canadian goose, gadwall, mallard, northern pintail, sand-hill crane and "Other"). However, the results obtained indicate that the CNN architectures are suitable as a bird detection technique with UAS imagery with average precision of 85.01% to 94.44% (IOU:0.3).

The bird size in the images used in the study was calculated to be 40x40 pixels in 6480x4320 pixels aerial photographs which implies that the bird objects can rarely be detected without preprocessing because the CNN resizes the large input image to a much smaller scale e.g. (416x416 in YOLO) which makes the target area in the image (bird area) very hard to recognize due to vanishing characteristics of the object. Therefore 233 sub-images were obtained for each aerial photograph so that bird object can be detected easier by CNNs with clearer localization.

In our project, the dataset consists of 13 images each of size 5472x3648 pixels and JSON file containing 18469 labels for 8 waterfowl classes performed by 13 experts. The average bird dimensions were calculated to be 52x54 pixels and therefore, the used CNN's in our project can benefit from applying similar cropping to avoid blank predictions (close to zero confidence).

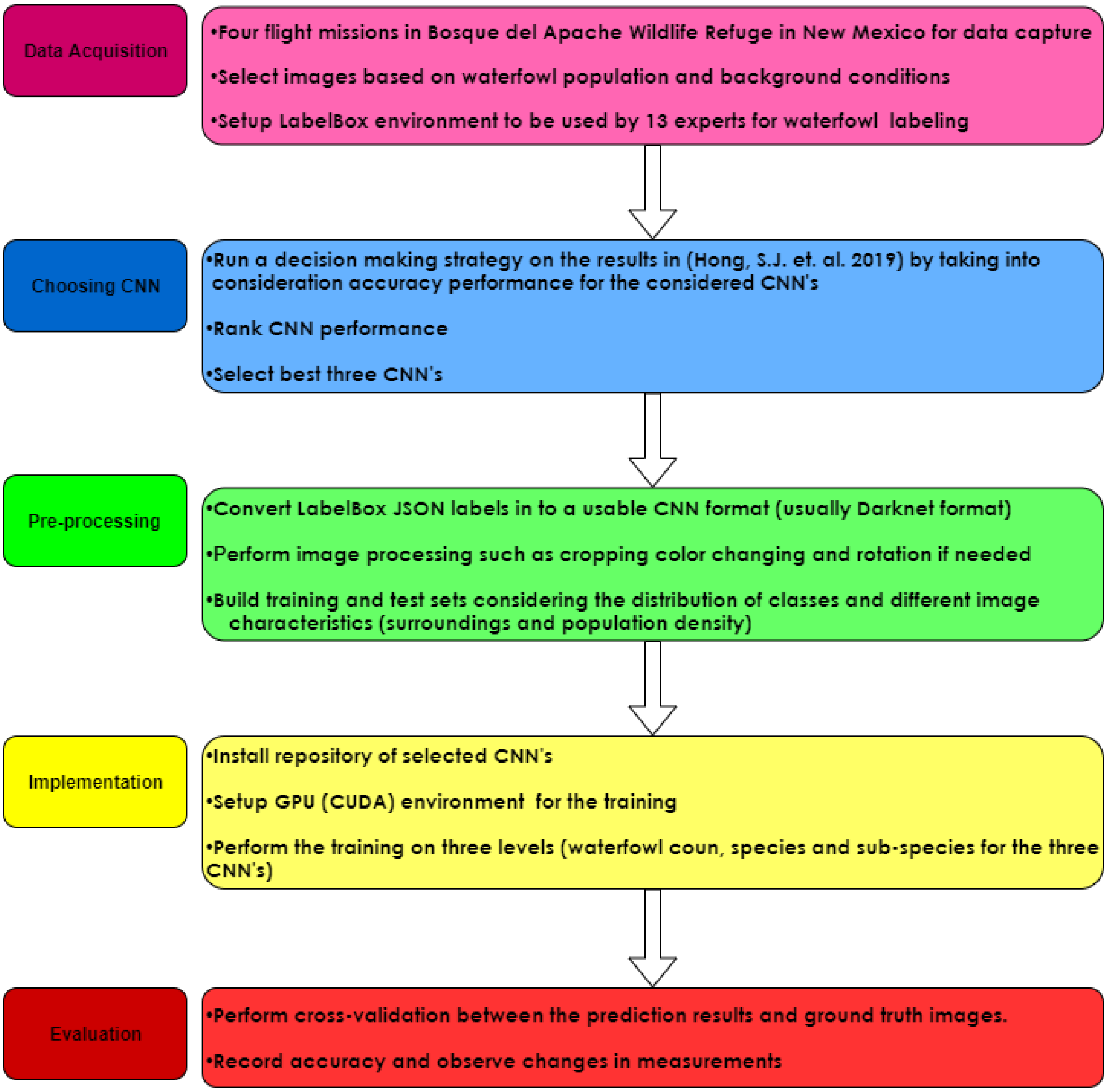
3. Approach

In this chapter, we describe the necessary steps for implementation. We begin with a big picture overview on the building block of this project from data acquisition to implementation. Then we justify our choice of CNN architectures with detailed description. Finally, we present the work and requirement breakdown structure.

3.1. General project breakdown structure

Our project goes through five main phases. The first phase is the data acquisition and consists of flight missions done on the studied area to collect imagery and LabelBox labelling by experts. The second phase is the justification of chosen CNN architectures where we apply a decision-making process to choose the best 3 CNN based on the results presented in (Hong, S.J. et. al. 2019). Third is pre-processing phase where we discover the necessary data manipulation and reformation so that our dataset is ready to be fed to the CNN. Implementation phase is the fourth stage where we begin the training and fine-tuning process to get acceptable accuracies and also setup the software (installing python environment and downloading necessary libraries) and hardware requirements (running the implementation on GPU). Finally, we do the evaluation phase where we compare the predictions with ground truth data.

It should be mentioned here that the data acquisition phase was done by USFWS and our part was to utilize their effort in implementation. Nevertheless, this phase is included because it represents a substantial asset in the study and gives a better general understanding in the application domain. Figure 16 represents the phases breakdown.



Data Acquisition

- Four flight missions in Bosque del Apache Wildlife Refuge in New Mexico for data capture
- Select images based on waterfowl population and background conditions
- Setup LabelBox environment to be used by 13 experts for waterfowl labeling

Choosing CNN

- Run a decision making strategy on the results in (Hong, S.J. et. al. 2019) by taking into consideration accuracy performance for the considered CNN's
- Rank CNN performance
- Select best three CNN's

Pre-processing

- Convert LabelBox JSON labels in to a usable CNN format (usually Darknet format)
- Perform image processing such as cropping color changing and rotation if needed
- Build training and test sets considering the distribution of classes and different image characteristics (surroundings and population density)

Implementation

- Install repository of selected CNN's
- Setup GPU (CUDA) environment for the training
- Perform the training on three levels (waterfowl coun, species and sub-species for the three CNN's)

Evaluation

- Perform cross-validation between the prediction results and ground truth images.
- Record accuracy and observe changes in measurements

Figure:16 General Workflow structure

3.2. USFWF Dataset

The surveyed area is Bosque del Apache Wildlife Refuge in New Mexico, two main location were surveyed (Maxwell Lake and Bosque del Apache). The flight mission took place between December 2017 and November 2018. Table 2 represents a summary of missions.

Table2: Summary of USFWS surveying mission

Location	Date	Time	Number of Images	Flight AGL (m)	Sensor	Platform	GSD (cm/px)
Maxwell Lake	12/15/2017	12:00	1963	50	senseFly S.O.D.A.	senseFly eBee	1.1
Bosque del Apache	11/06/2018	12:00	974	40	Hasselblad L1D-20c	DJI Mavic	0.94
Bosque del Apache	11/07/2018	9:30	1278	40	Hasselblad L1D-20c	DJI Mavic	0.94
Bosque del Apache	11/13/2018	12:00	608	40	Hasselblad L1D-20c	DJI Mavic	0.94
Bosque del Apache	11/27/2018	11:30	264	40	Hasselblad L1D-20c	DJI Mavic	0.94

Only 13 images were selected from this set based on the following requirements: images must contain substantial number of birds (>10) and contain a variety of background conditions. Figure 17 shows the studied area of Bosque del Apache

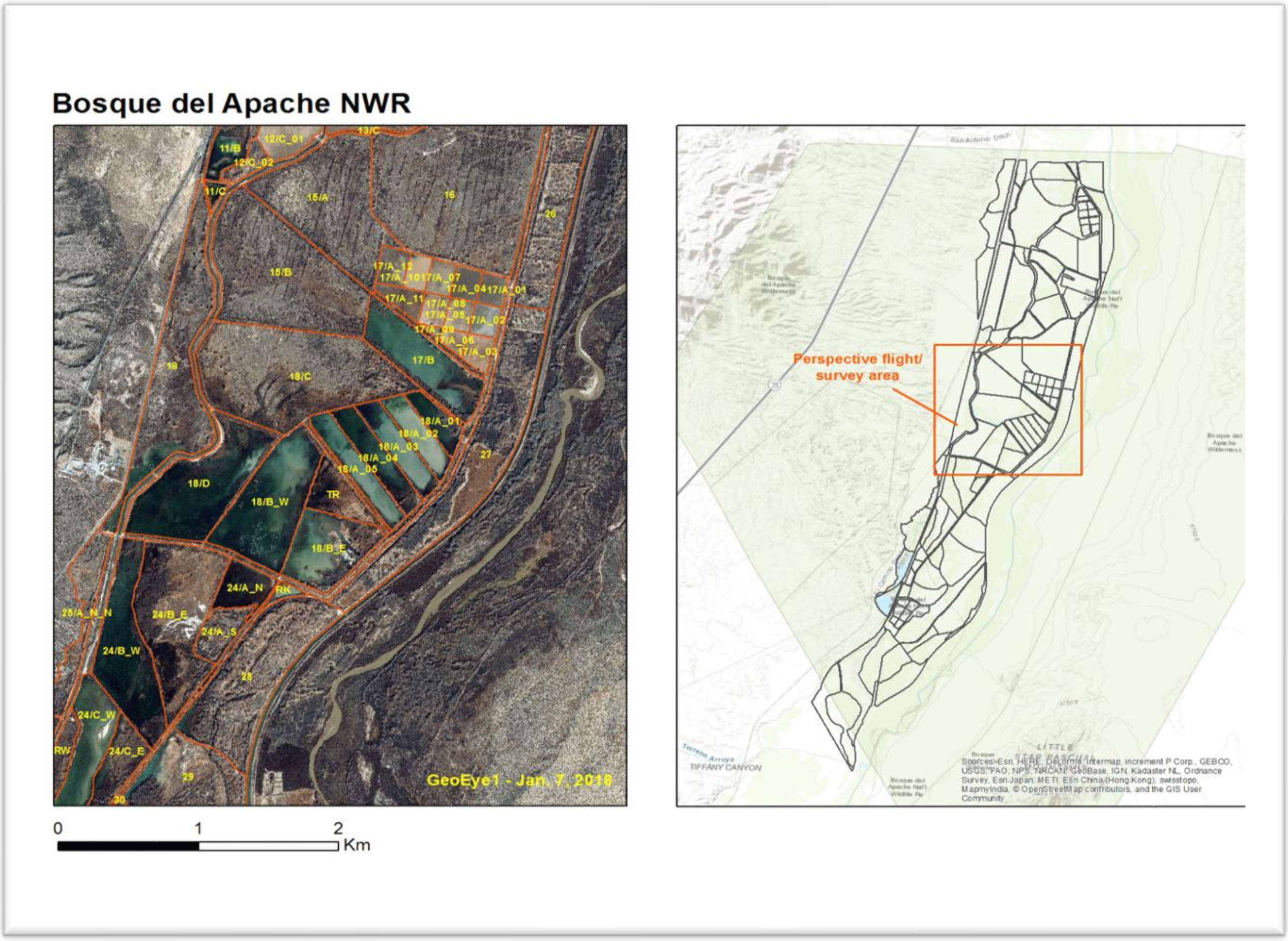


Figure17: Bosque del Apache Wildlife Refuge

Wildlife Refuge in New Mexico. The images were then used in a LabelBox environment to be labeled by 13 experts

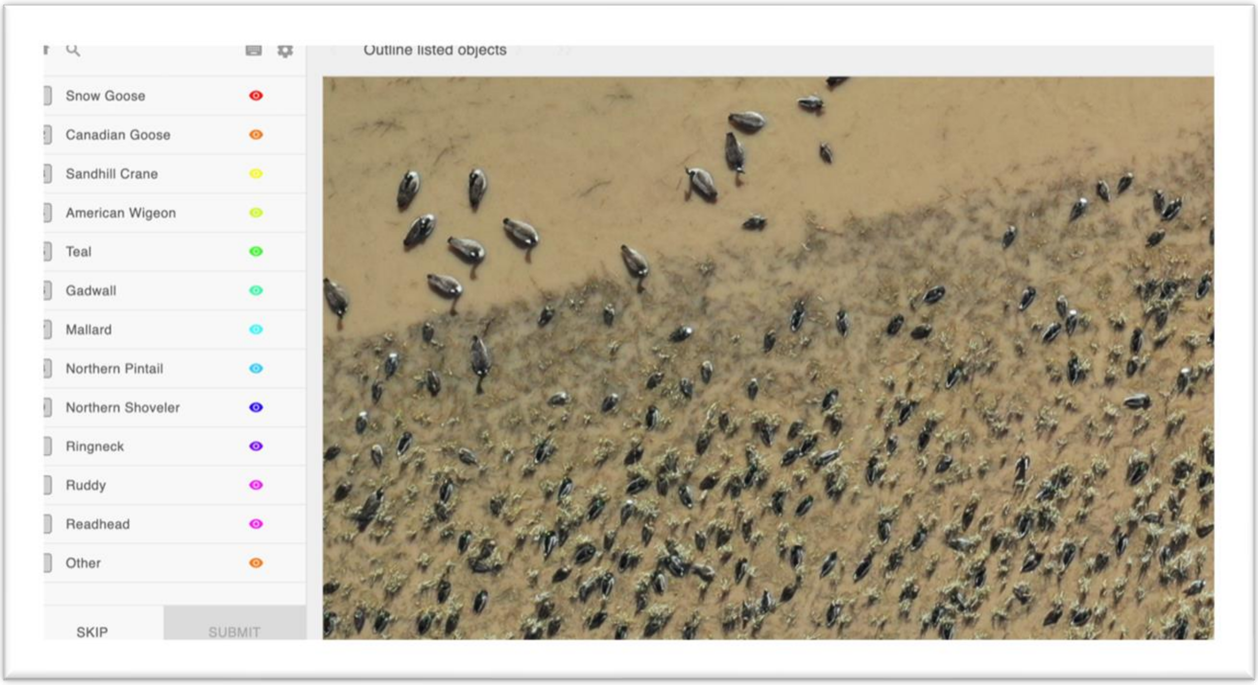
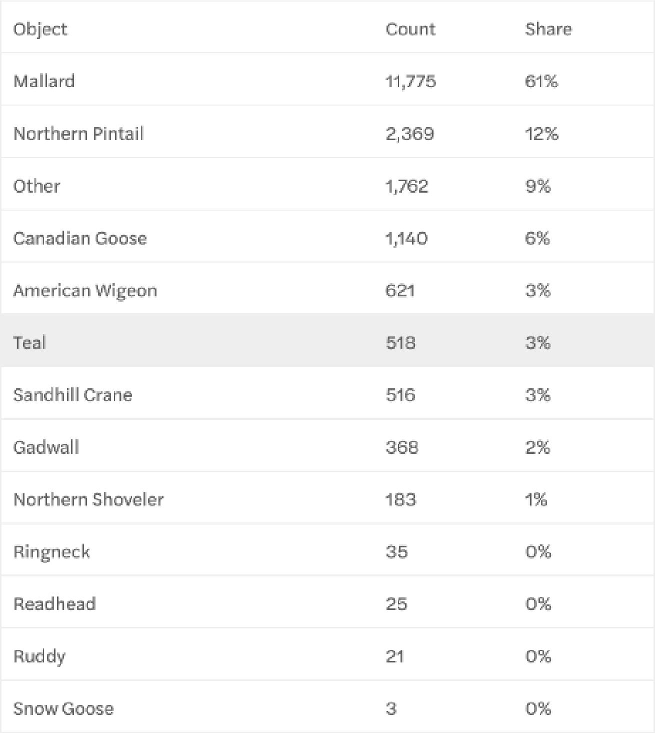


Figure18: LabelBox interface

We can see that the number of sub-species classes in 12 how ever classes with less than 200 label are not representative and have a very slight chance of being detected (compared to other larger classes) and will behave as a source of confusion to the CNN. The eliminated classes are (Northern Shovler, Ringneck, Ruddy and Redhead).



Object	Count	Share
Mallard	11,775	61%
Northern Pintail	2,369	12%
Other	1,762	9%
Canadian Goose	1,140	6%
American Wigeon	621	3%
Teal	518	3%
Sandhill Crane	516	3%
Gadwall	368	2%
Northern Shoveler	183	1%
Ringneck	35	0%
Readhead	25	0%
Ruddy	21	0%
Snow Goose	3	0%

Figure19: Sub-species class count

3.3. Choice of CNN Architectures

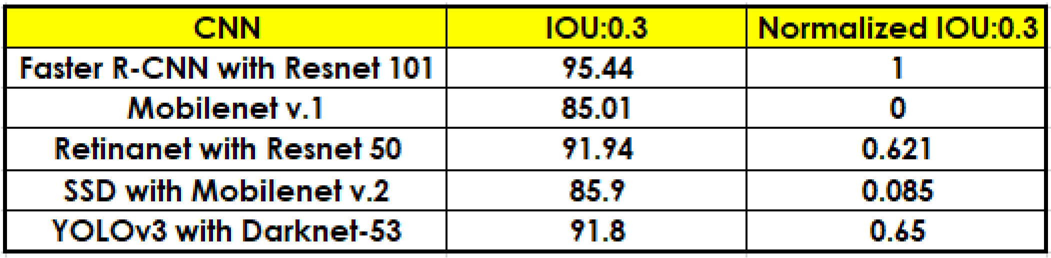
We will run a Weighted Linear Combination (WLC) operation in order to rank the used CNNs and take the best three based on accuracy defined as IOU:0.3 score. It should be noted here that not all CNN's in Table1 would be taken into consideration. R-FCN with Resnet 101 will not be taken as this model performed very similarly to Faster R-CNNs with slightly less accuracy but better time. Another reason to suppress R-FCN is that it contains the same feature extractor of Faster R-CNN which is (Resnet 101) as the core model for detection and thus not much to compare would be available if we chose both in our implementation. Also, we will only take the IOU:0.3 measurement accuracy as it was stated in the study that even with IOU:0.3 all bird objects were detected. Also, YOLOv2 was suppressed because the latest version of YOLOv3 was developed to obtain better accuracy which is the prime focus in this study.

We first normalize the speed and accuracy for each option using the below equation.

$$z_i = x_i - \min(x)/\max(x) - \min(x)$$

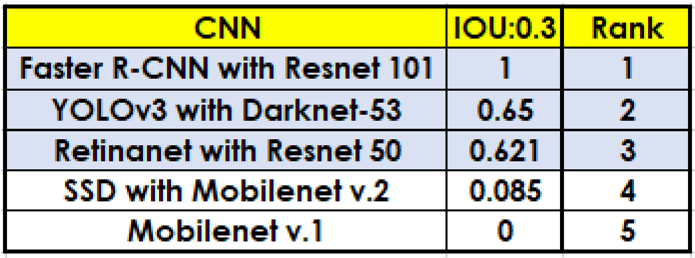
To obtain values between 0 and 1 for each field.

Table3: Normalized Speed and accuracy measurements



CNN	IOU:0.3	Normalized IOU:0.3
Faster R-CNN with Resnet 101	95.44	1
Mobilenet v.1	85.01	0
Retinanet with Resnet 50	91.94	0.621
SSD with Mobilenet v.2	85.9	0.085
YOLOv3 with Darknet-53	91.8	0.65

Table4: WLC operation scores and ranking



CNN	IOU:0.3	Rank
Faster R-CNN with Resnet 101	1	1
YOLOv3 with Darknet-53	0.65	2
Retinanet with Resnet 50	0.621	3
SSD with Mobilenet v.2	0.085	4
Mobilenet v.1	0	5

3.4. Detailed specifications of chosen CNN's

You Only Look Once(YOLO): YOLO is a CNN designed by collaboration between University of Washington, Allen Institute of AI and Facebook AI Research and built for wide range of computer vision tasks such as but not limited to activity recognition, face detection, face recognition and video object co-segmentation.

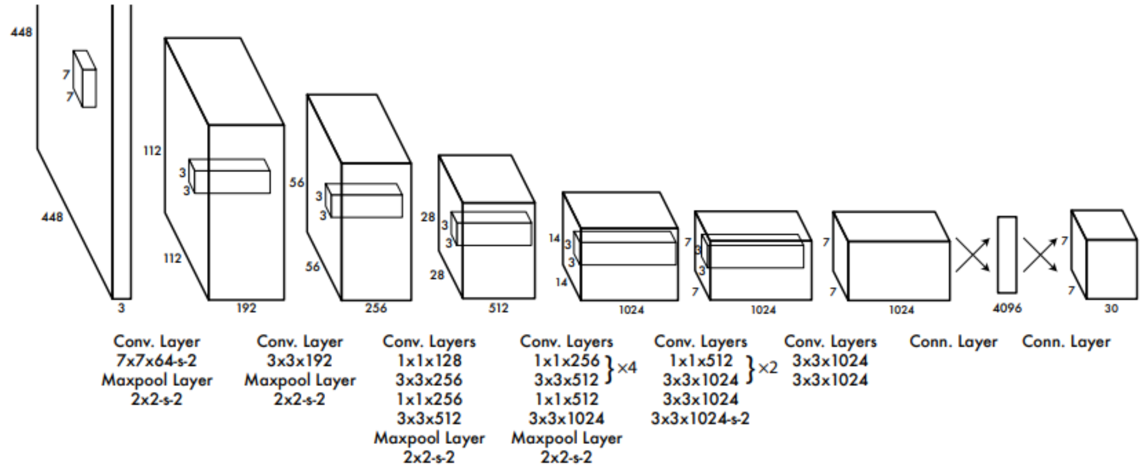


Figure 20: YOLO Architecture, Adopted from "Unified, real-time object detection" by (Redmon, J. et al 2016) Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

YOLO utilizes a single CNN for object detection and the architecture takes the whole image and split it up on a SxS grid, pass it through a neural network to create bounding boxes and class predictions to determine the final detection output.

To calculate the bounding boxes, YOLO implements two key post-processing steps: IOU (Intersect of Union) and NMS (Non-maximum suppression).

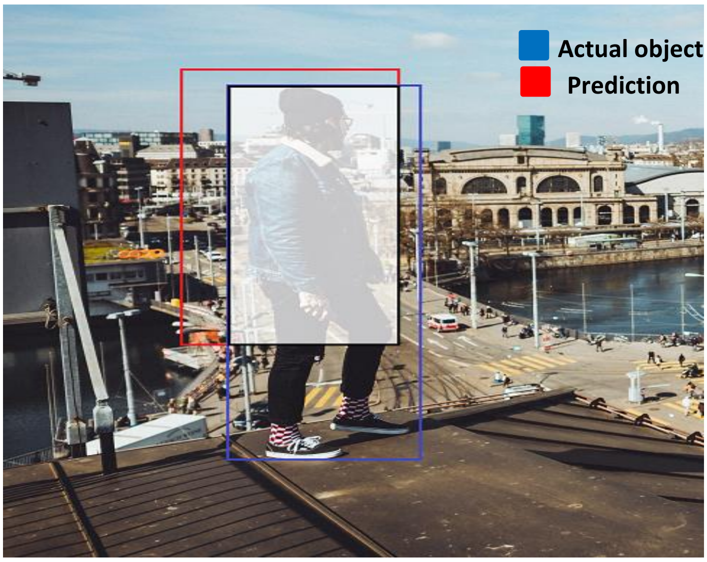


Figure21: IOU illustration

In Figure21, The Red box is what the computer thinks is the person, while the blue is the actual bounding box of the object. The overlap of the two boxes gives us our IOU.

NMS ensures we identify the optimal cell among all candidates where the desired object belongs. Rather than determining that there are multiple cases of the object in the image, NMS chooses the highest probability of the boxes that are determining the same object (Rothe, R et. al 2014). Figure 22 shows a demonstration of NMS.

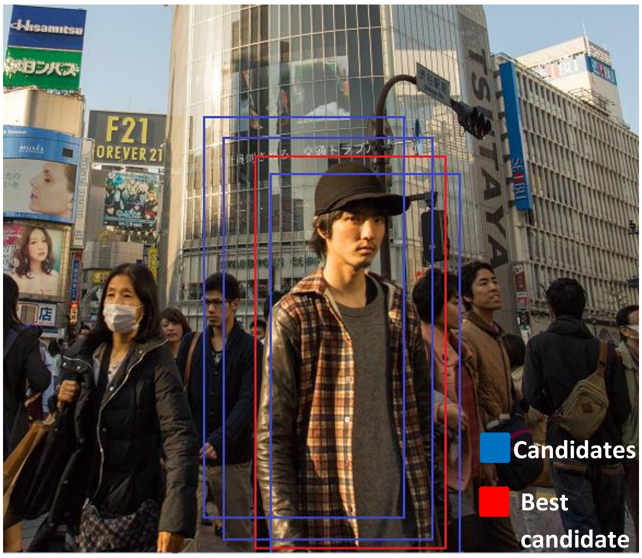


Figure22: NMS illustration

However, one drawback of YOLO is the inability to detect multiple objects that are either too close or too small (Redmon, J. et al 2016).

Faster R-CNN:

Faster R-CNN is a CNN designed by Microsoft Research to propose possible regions of interest and to classify and adjust them.

R-CNN is a type of detecting algorithms which detect objects by proposing region in the input image that may contain an object. These regions which are also called anchors in the image are proposed by an external region proposal method which is selective search. A convolutional layer is then applied to the proposed regions for different purposes. The problem with R-CNNs is that the time consumption during the testing process that is due to the large number of regions proposed.

Fast R-CNNs is an improved version of R-CNN that applies the convolution operation to the input image first then it performs region proposal on the feature maps

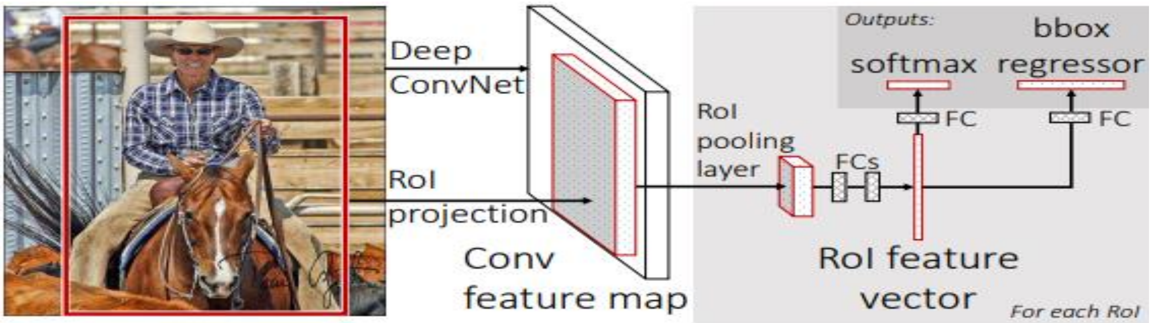


Figure23: Fast R-CNN architecture, Adapted from "Fast R-CNN" by (Girshick, R. 2015) In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

extracted by the convolution process which is less in size as convoluting the input image with the CNN kernels results smaller images of feature hence less proposed regions.

Faster R-CNN is an improvement over Fast R-CNN. The input image of Faster R-CNN is first resized to 600 pixels on the shorter side and 1000 pixel on the shorter side then a sliding window of the size 40x60 is applied to the reduced image which result on a maximum of 2400 possible window locations.

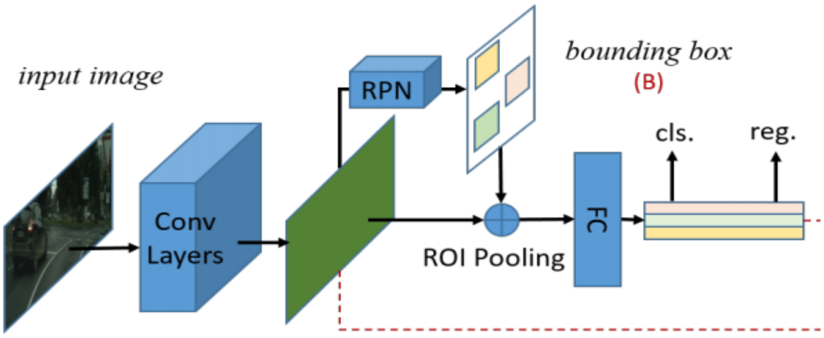


Figure24: Faster R-CNN architecture, Adapted from "Domain Adaptive Faster R-CNN for Object Detection in the Wild" by (Chen, Y. et al 2018) In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3339-3348).

The anchors are proposed or using Region Proposal Network (RPN) shown in Figure24. Each produced region has what is called an abjectness score measured using IOU.

the model ignores boundary windows and applies NMS to further reduce the number of proposed regions

this operation results 512 window locations out of the 2400. The 512 regions are then split in half as positive and positive regions based on the IOU of each window.

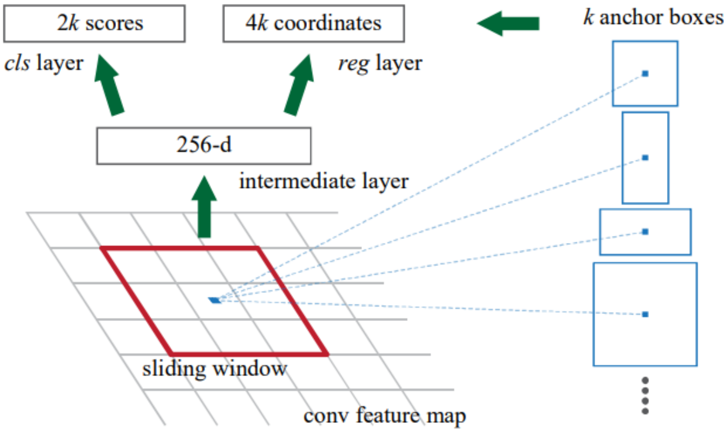


Figure25: Region Proposal Network (RPN), Adopted from "Faster r-cnn: Towards real-time object detection with region proposal networks." By (Ren, S. et al 2015) In Advances in neural information processing systems (pp. 91-99).

Retinanet:

Retinanet is a single-stage detection architecture, built by Facebook AI Research (FAIR) to outperform the state-of-the art one-stage detectors (YOLO and SSD) and achieve accuracy results close to two-stage detectors (Faster-RCNN, Fast R-CNN).

Retinanet overcome the problem of accuracy in one-stage detectors by densely covering all possible locations that could contain objects in an input Image.

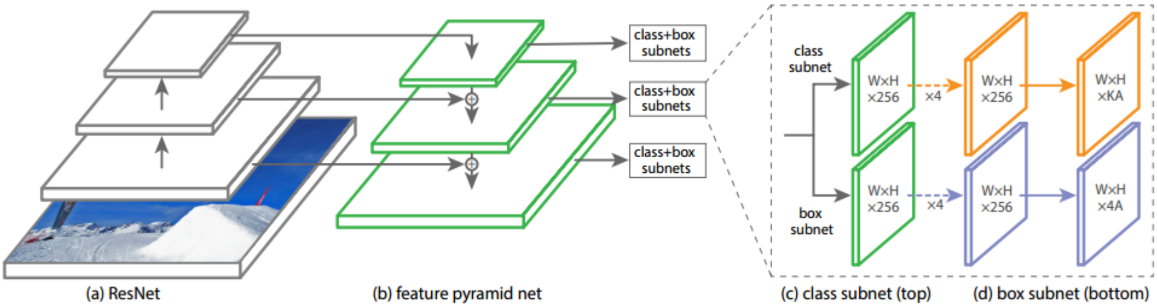


Figure26: Retinanet Architecture, Adopted from "Focal Loss for Dense Object Detection" by (Lin, T.Y. et al 2017) In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988

In practice, Retinanet processes ~100k locations which is significantly larger number than some one-stage detectors such as YOLOv2(~1k) and SSD(8~26k).

The Primary problem of having that much possible locations in an image is that most of these locations would be from the background of the Image and therefore uninformative and only few locations from the foreground of the image. This problem is called Class Imbalance and Figure27 shows an illustration of the problem.

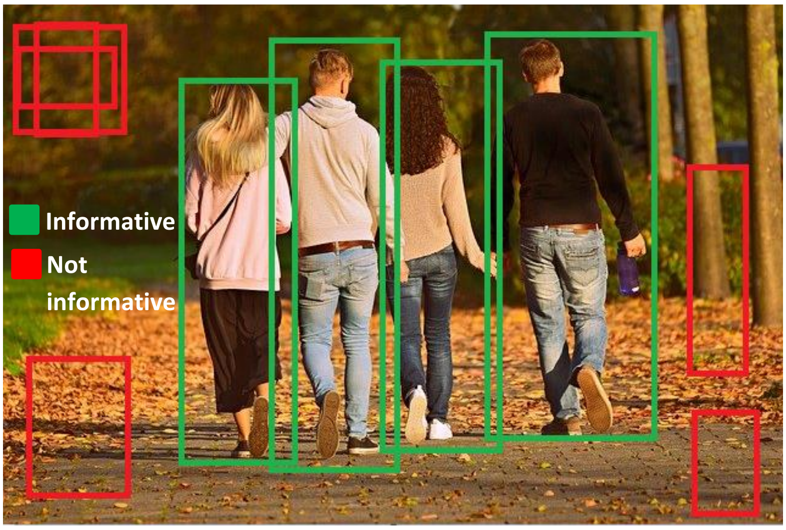


Figure27: illustration of class imbalance

Retinanet solves the problem of class imbalance by introducing a Focal Loss function instead of cross entropy loss function which is commonly used in one-stage detectors. The importance of Focal Loss is that it reduces the features learned from background locations (not informative) while maintaining the features learned from the foreground locations (informative).

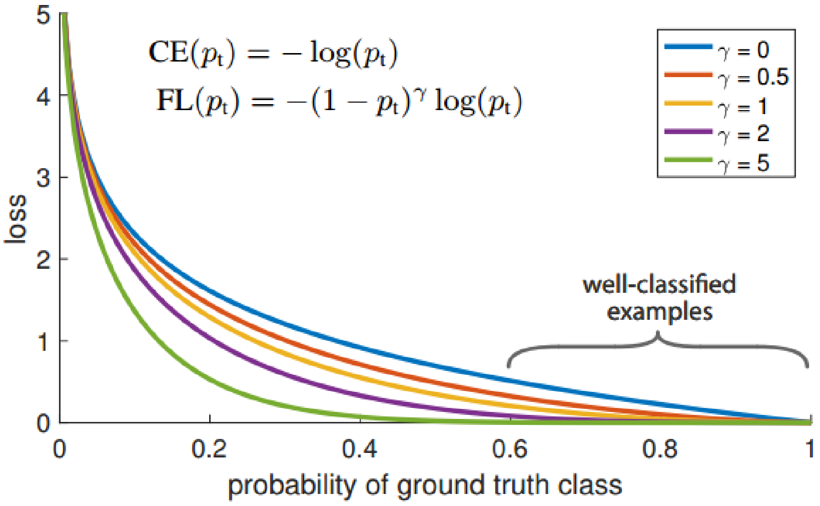


Figure28: Cross Entropy and Focal loss functions, Adopted from "Focal Loss for Dense Object Detection" by (Lin, T.Y. et al 2017) In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988

3.5. Workflow for the evaluation

Before proceeding with implementation, we must make sure that dataset is prepossessed, and the software and hardware setups to supports the implementation. The first part of the setup is preprocessing the dataset and this is done in the following steps:

- Convert LabelBox JSON Annotations to each CNN acceptable format.
- Split each image to a group on sub-images for the object to be easily identified.
- Adjust the labels so each label corresponds to an object in the sub-images.
- Construct a training set (images not fed to the CNN in the training process)

We should also link the software with the hardware available. For example, CUDA Toolkit with compatible Keras and TensorFlow versions. Moreover, minimum requirements of software and hardware should be met such as Windows 64bit with Python v3.5 or higher to support the implementation of the chosen CNNs. Figures 29 and 30 show the requirements and work structures, respectively.

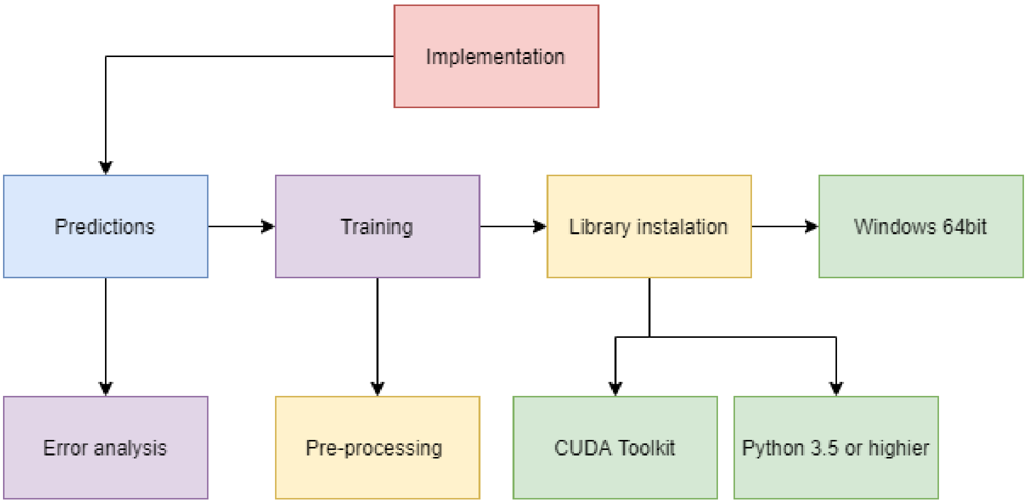


Figure 29: Requirements Breakdown

The data should be segmented into 3 parts namely: training, testing, and evaluation. It should be mentioned here that the training data should be larger than testing and evaluation, giving the CNNs the chance as much as possible from all the possible variations in the dataset. The accuracy measurement should be divided into two major parts: accuracy of detection and accuracy of classification. The accuracy of detection will be defined as how close the number of successful predictions to the actual number of birds in the test set is. The classification accuracy is how close the number of successful class predictions to the number of actual class occurrences in the test set. Nevertheless, manual evaluation on the results (looking at each image result) is necessary to understand why an error has occurred in the detections.

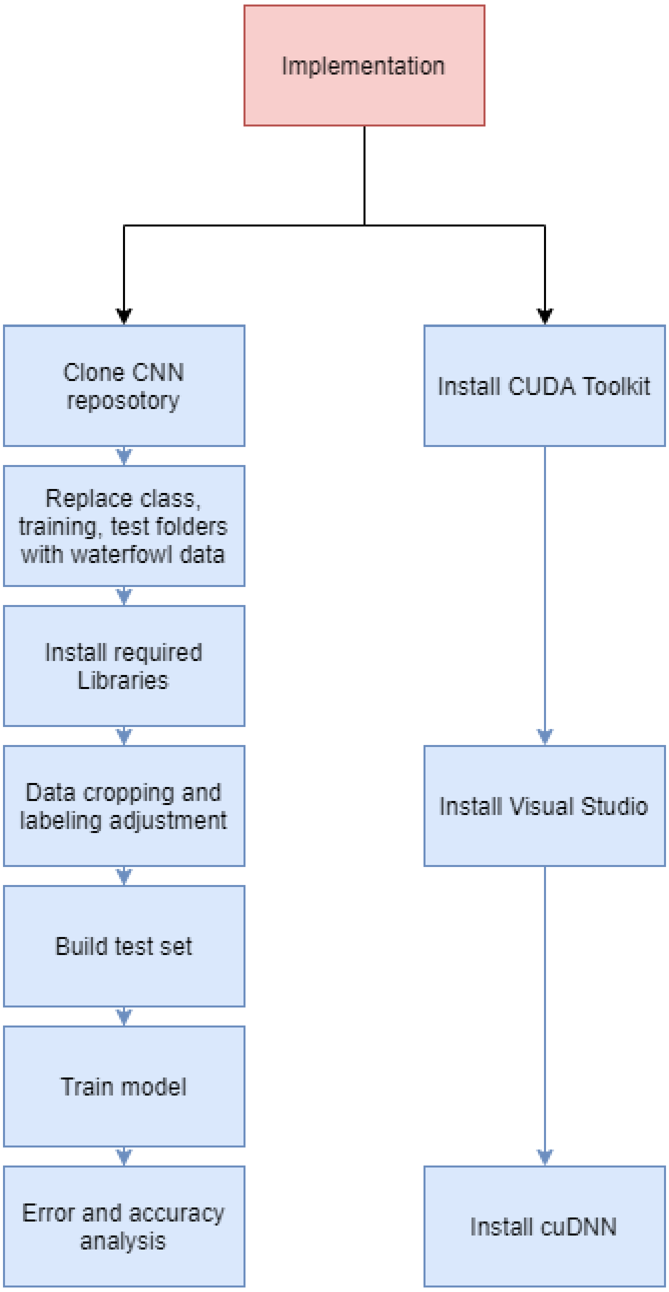

Implementation
Clone CNN reposotory
Replace class, training, test folders with waterfowl data
Install required Libraries
Data cropping and labeling adjustment
Build test set
Train model
Error and accuracy analysis
Install CUDA Toolkit
Install Visual Studio
Install cuDNN

Figure 30: Work breakdown

4. Implementation

In this chapter, we present the two steps of implementation. We begin with pre-processing phase and discuss the necessary adjustments and modifications that should be done to the dataset. The second phase is the training and includes a description of the used software and hardware setup along with sample image results and an explanation of how to numerically present the performance of each CNN.

4.1. Pre-processing

In this section, we describe the data preprocessing steps along with the necessary software and hardware setup for the implementation.

First, we should convert LabelBox JSON annotation to Darknet annotation format (YOLO format). The LabelBox annotations are represented as:

<IMAGE_NAME>","Label":{"CLASS_NAME_1":[{"geometry":[{"x":X0,"y":Y0},{"x":X1,"y":Y1}……..":{"CLASS_NAME_2"":[{"geometry":[{"x":X0,"y":Y0},{"x":X1,"y":Y1}……….]}.

While YOLOv3 annotation format is as follows:

<IMAGE_PATH> Xmin,Ymin,Xmax,Ymax,<CLASS_ID> [each label in a single line]

Therefore, a python script to perform the conversion war written. The code goes to each row in the JSON file and extracts ***<IMAGE_NAME>*** and append it with the calculated X,Y (min and max)coordinates for each object ***<CLASS_NAME>.***

The conversion process result was 18469 labels for 8 classes. However, as the dataset was labelled by 13 experts: redundant labels for each object should be removed in the following manner:

- If two or more labels have an IOU of more than 50%
  - Keep the label with the smallest area

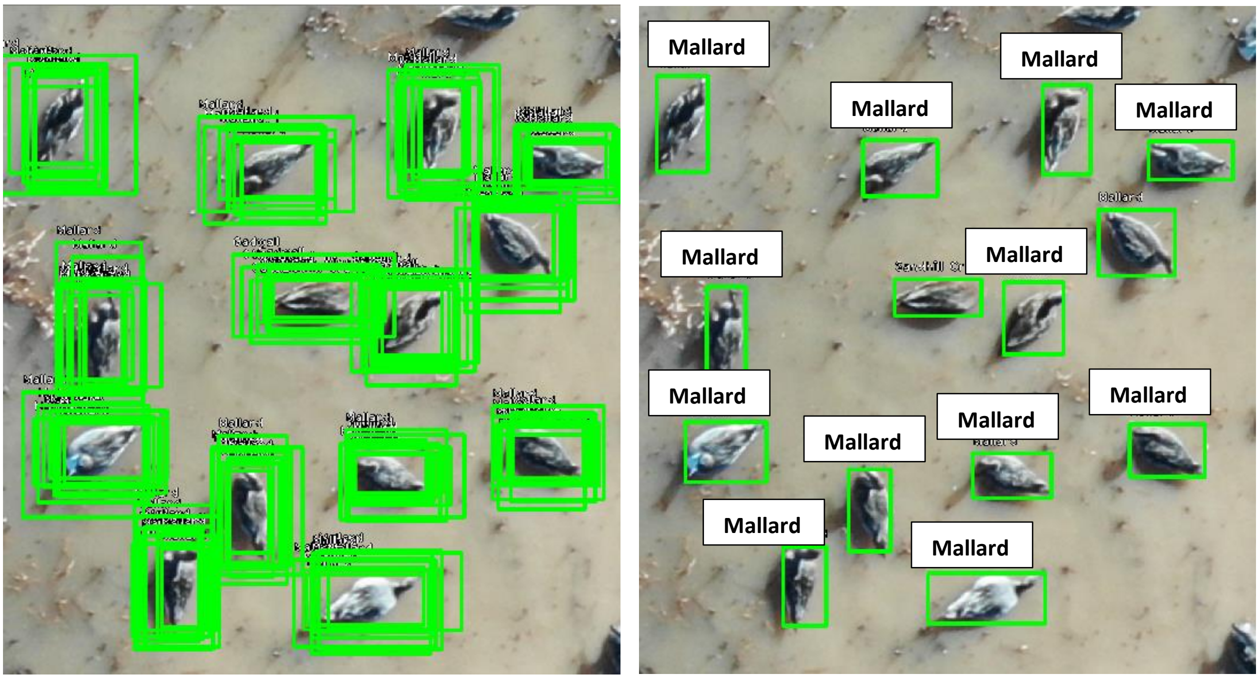


Figure31: Removing multiple labels from the training set

we chose to keep the smallest-area label because it contains less background noise relative to larger labels bounding the same object. It should be mentioned here that such a change is only valid under the assumption that all labels cover the body of the desired object as shown in the figure 31

The result of removing multiple labels resulted 2908 labels with the following distribution:

Table 5: Class distribution in the training set

8 Class	Count	3 Class	Count	1 Class	Count
American_Wigeon	162	Duck	2559	Waterfowl	2908
Teal	263				
Gadwall	174				
Mallard	1000				
Northen_Pintail	600				
Other	360				
Sandhil_Crane	118	Crane	118		
Canadian_Goose	231	Goose	231		

Each image is 5472x3648 pixels and an average label size is 52x54 which takes .014% of the total image area. This small percentage hinders YOLOv3 ability to detect desired objects. Thus, if we crop the image to multiple sub-images to enlarge the ratio of area taken by the label in the image.

Each image will be cropped to 56 sub-images (7 rows 8 columns) of size 684x521 pixels and therefore the average percentage of area covered by a single label will be 0.78%. Figure 32 shows how a bird is enlarged by the cropping process.

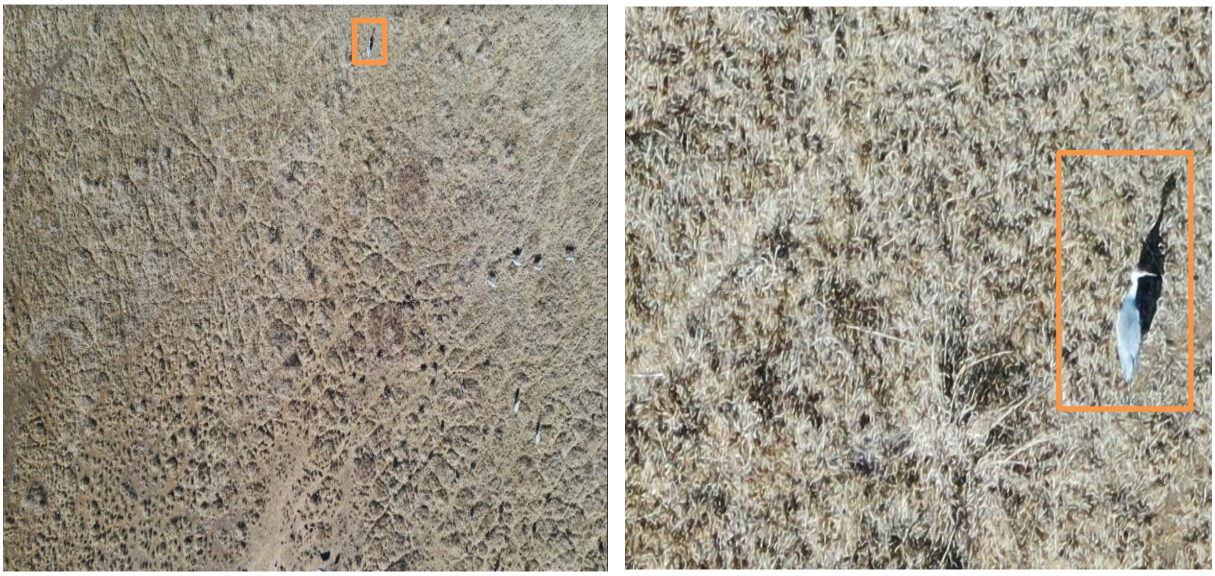


Figure32: Cropping results

After that, all labels should be adjusted to refer to each sub-image. The tile at which a label belongs can be found by the results of integer division between the original X,Y coordinates and the sub-image size. The new coordinates of the label will be the reminder of division operation as shown in the example below.

<IMAGE_1> 4536,2581,4589,2602 → <IMAGE_1_06_04> 432,497,485,518

It should be mentioned here that, performing image cropping brings an advantage of reducing the background noise introduced to the CNN since we only feed the CNN on sub-images that contain labels. In our dataset a total of 728 sub-images were generated but only 357 had labels.

Now that we have our training set ready, we can begin to build the test set by taking the following points into consideration:

- Test images should contain all classes
- Should not be fed to the CNN in the training process
- A good practice is to augment the data (rotation, zoom in)
- Choose different backgrounds, bird sizes
- Different population density

30 images were chosen with 177 waterfowls. Table 6 shows the class distribution in the test set.

Table6: Class distribution in the test set

8 Class	Count	3 Class	Count	1 Class	Count
American_Wigeon	8	Duck	143	Waterfowl	177
Teal	7				
Gadwall	6				
Mallard	95				
Northen_Pintail	9				
Other	18				
Sandhil_Crane	5	Crane	5		
Canadian_Goose	29	Goose	29		

4.2. Training

The training processes was performed on Windows64bit equipped with NVIDIA GeForce GTX 1080 with 8GB dedicated RAM.

Three training were performed with the flowing levels

- Sub-species (American wigeon, Canadian goose, gadwall, mallard, northern pintail, sand-hill crane and "Other")
- species (Duck Goose and Crane)
- Waterfowl (count)

The idea of running three trainings with on three levels is to evaluate how results change by changing the complexity of features that must captured by the CNNs

YOLOv3 Training:

We used https://github.com/AntonMu/TrainYourOwnYOLO repository. The optimal number of epochs to be performed was decided using the elbow method where CNN loss drops by less than 5% as shown in figure 33

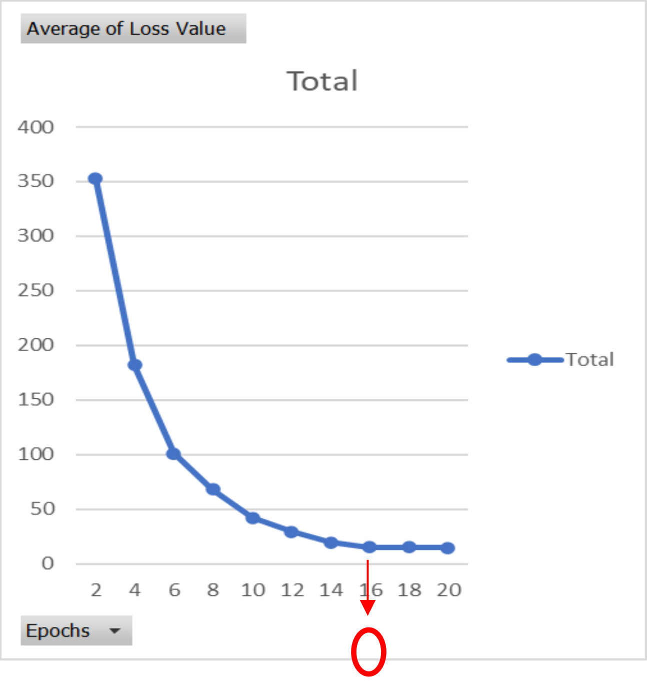


Figure33: Loss value vs number of epochs

The number of epochs changes with each level as 16, 12 and 11 for the sub-species, species, and waterfowl respectively, with an average of 200 seconds per epoch. This decrement in number of epochs is due to the decrement of data complexity or number of class-specific patterns needed to be learned.

The output of YOLOv3, Retinanet and Faster R-CNN consists of two parts. The first part is labeled images and the second is an excel sheet with all labels with a degree of confidence for each label. The confidence value refers to how sure the CNN is about the generated prediction.

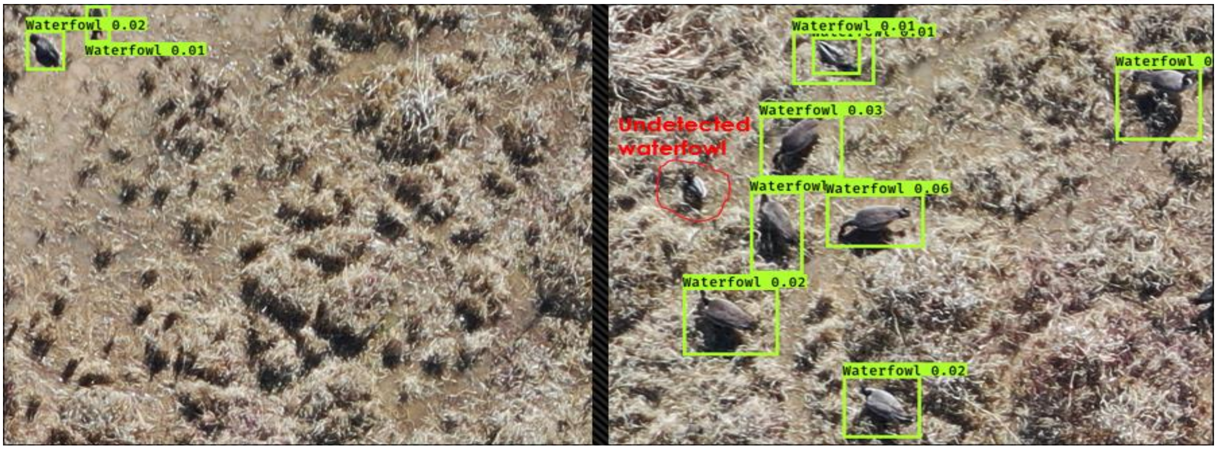


Figure34: sample of YOLOv3 results for waterfowl level

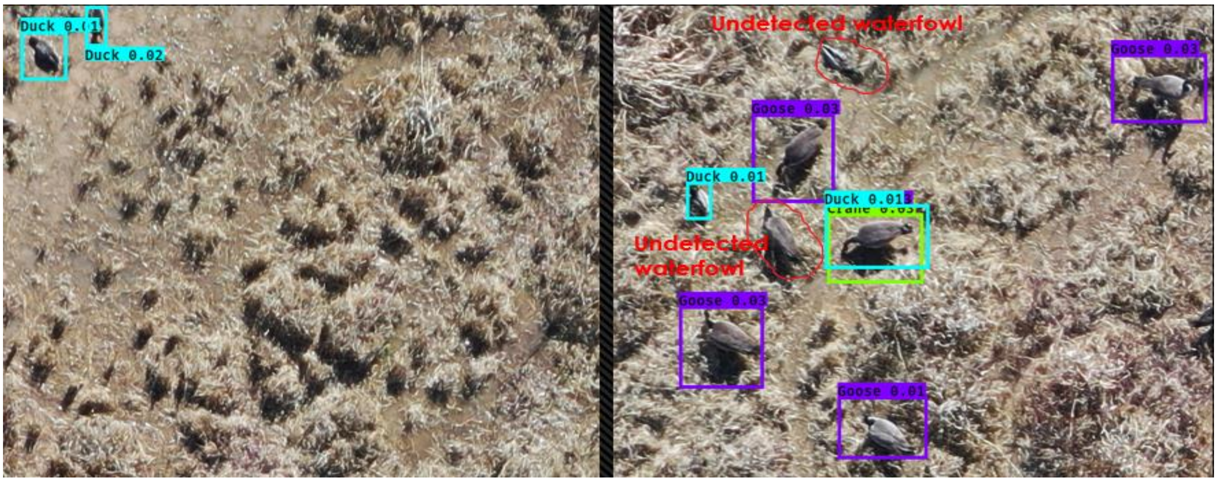


Figure35: sample of YOLOv3 results for species level

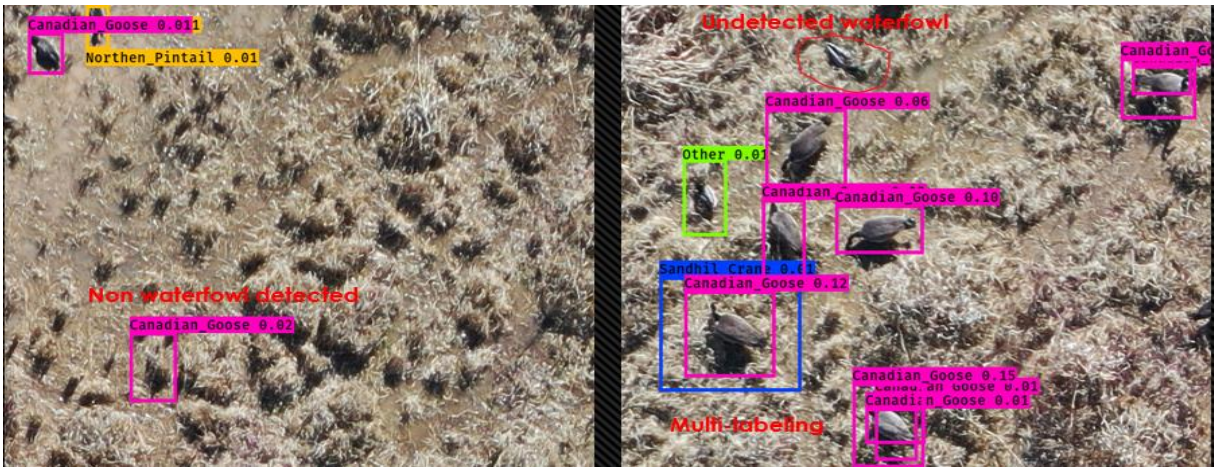


Figure36: sample of YOLOv3 results for sub-species level

Multi-Labelling problem:

We can see in figures 34, 35 and 36 that there is a problem of labelling an object more than once. There are two scenarios for this event.

- Two or more labels with 100% IOU and different class assignment
    - Caused by the CNN not having enough training samples for to distinguish between each class, so the different class assignments happen to be above the display threshold which was 0.1% in YOLOv3 case.
- Two or more labels with +50% IOU and different or same class assignment
    - This problem is caused by the generated BBOXES having very close confidence values which happen to be above the display threshold
    - The close confidence values are also a result of not having enough training samples

The approach of solving the problem is to eliminate intersecting labels with IOU +50% and lower confidence.

Confusion matrix representation:

The most common method of representing prediction results is to build a confusion matrix. A cross validation process was implemented by taking the ground truth labels as a reference where we calculated the IOU between each label in the ground truth and prediction labels.

Then the ground truth label gets assigned the prediction label with the highest IOU.

An example of the final form of the confusion matrix is shown below in Table 7.

Table7: Example Confusion Matrix

Classes	Class 1	Class 2	Class 3	Undetected	Non-Waterfowl
Class 1	🟩	🟨	🟨	🟦	🟥
Class 2	🟨	🟩	🟨	🟦	🟥
Class 3	🟨	🟨	🟩	🟦	🟥

The diagonal values (green cells) represent successful detection and classification of the desired object. If there is a ground truth label that has maximum IOU of 0, then we know that this label was undetected (blue cells).

The process can be reverted by taking the predictions as a reference and if a prediction label has maximum IOU of 0 then we know that this label refers to non-waterfowl label but was wrongly labeled as one (red cells).

The yellow cells represent successful detection but incorrect classification of the desired object.

Retinanet Training:

The implementation of Retinanet was achieved using the famous https://github.com/fizyr/keras-retinanet repository. The number of epochs per level was 3, 2 and 2 for the sub-species, species, and waterfowl respectively, with an average of 1340 seconds per epoch. The increment of average computation time is expected as Retinanet generates much more areas in the image that potentially correspond to an object.

As mentioned in section 3.2, Retinanet uses a focal loss function to boost the confidence of "informative labels" while keeping "uninformative labels" at a fairly low degree of confidence. Thus, it is expected that Retinanet labels will have higher average confidence value relative to YOLOv3. Boosting the confidence does not necessarily enhance the accuracy. The primary goal of this function is to prevent unwanted detections from the results. Also, boosting the confidence can lead to detection of non-waterfowl object in images having shadows of plants and birds that look like might look like a waterfowl.

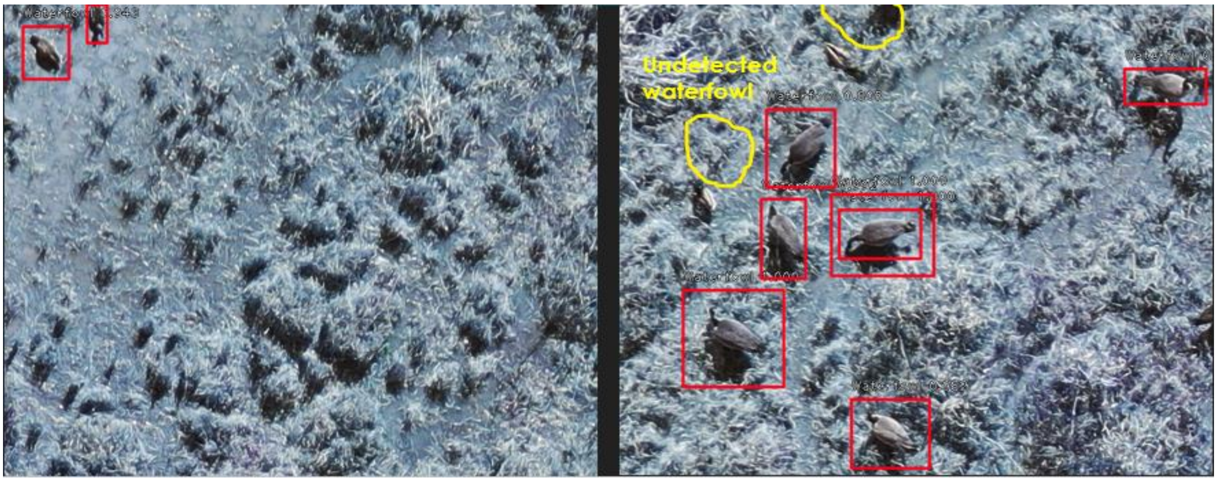


Figure 37: sample of Retinanet results for waterfowl level

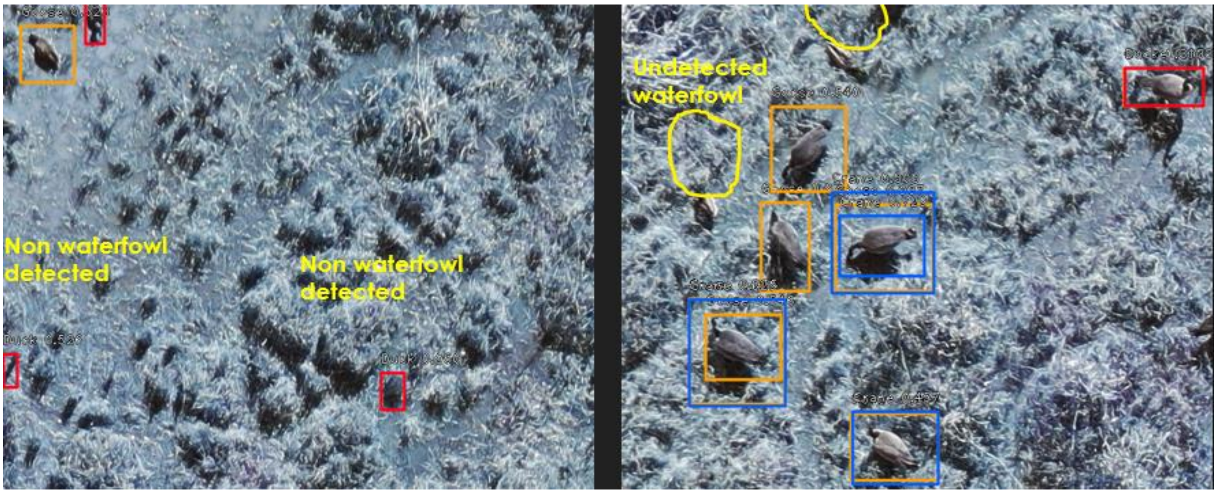


Figure 38: sample of Retinanet results for species level

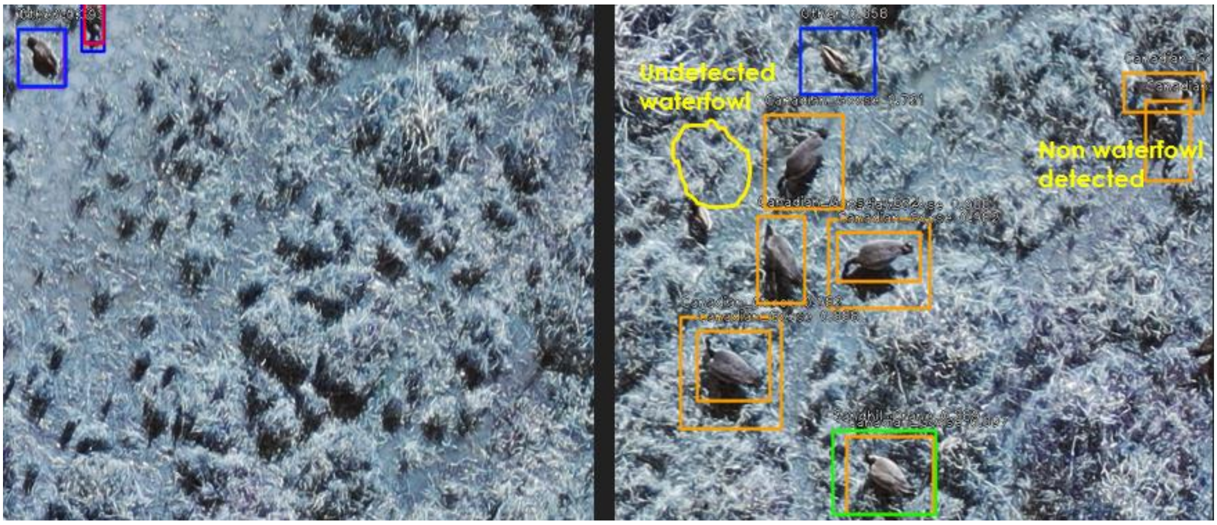


Figure 39: sample of Retinanet results for sub-species level

Faster R-CNN Training:

we used https://github.com/kbardool/keras-frcnn.git repository to implement Faster R-CNN. Faster R-CNN is a two-stage detector where the first stage performs binary classification of "object" and "no object" then second stage proceeds with further pattern recognition to classify "objects". The term "Faster" is a description of the evolution of this type of two-stage detectors relative to the "Fast R-CNN". However, single stage detectors such as YOLOv3 and Retinanet are faster than Faster R-CNN (Soviany and Ionescu 2018).

The number of epochs per level was 118, 92 and 87 for the sub-species, species, and waterfowl respectively, with an average of 2280 seconds per epoch. As the classification process is not performed with a single shot, it is expected that the first stage (binary classification) will be consistent in terms of the number of detected objects. However, further classification accuracy essentially depends on the amount of data fed per class.

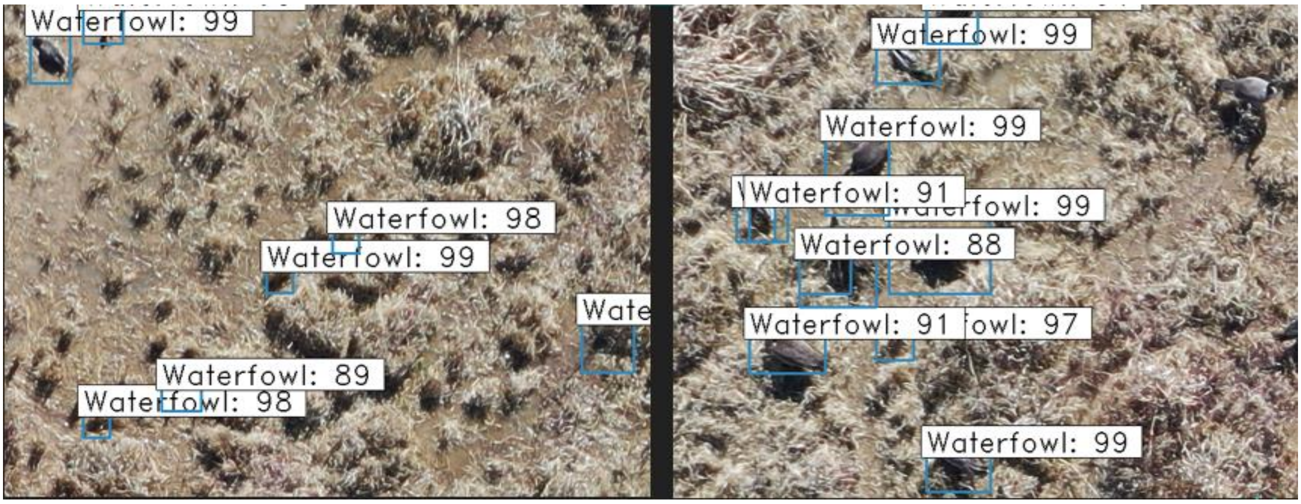


Figure 40: sample of Faster R-CNN results for waterfowl level

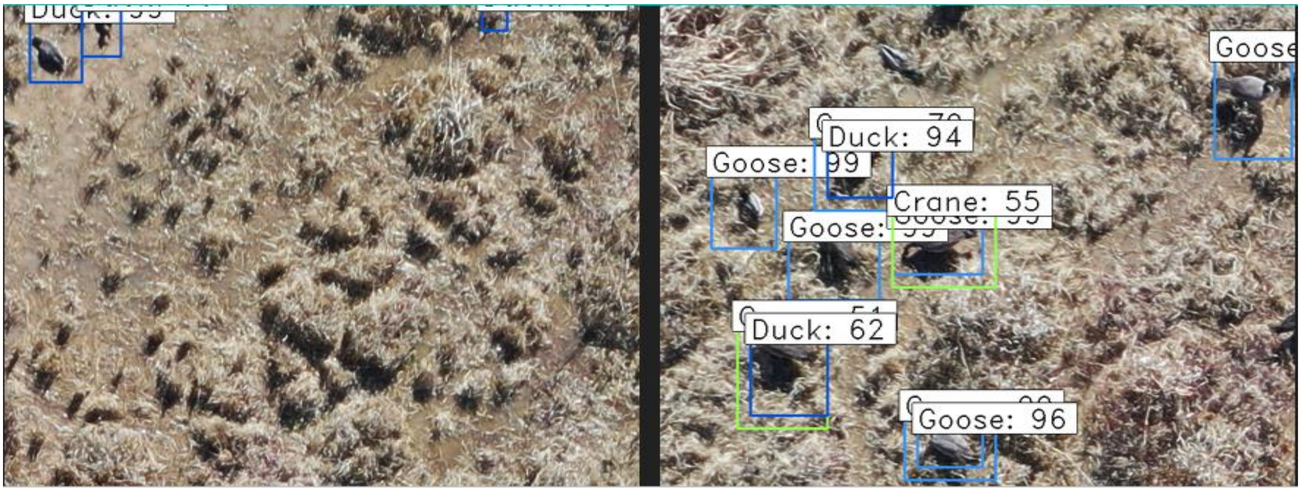


Figure 41: sample of Faster R-CNN results for species level

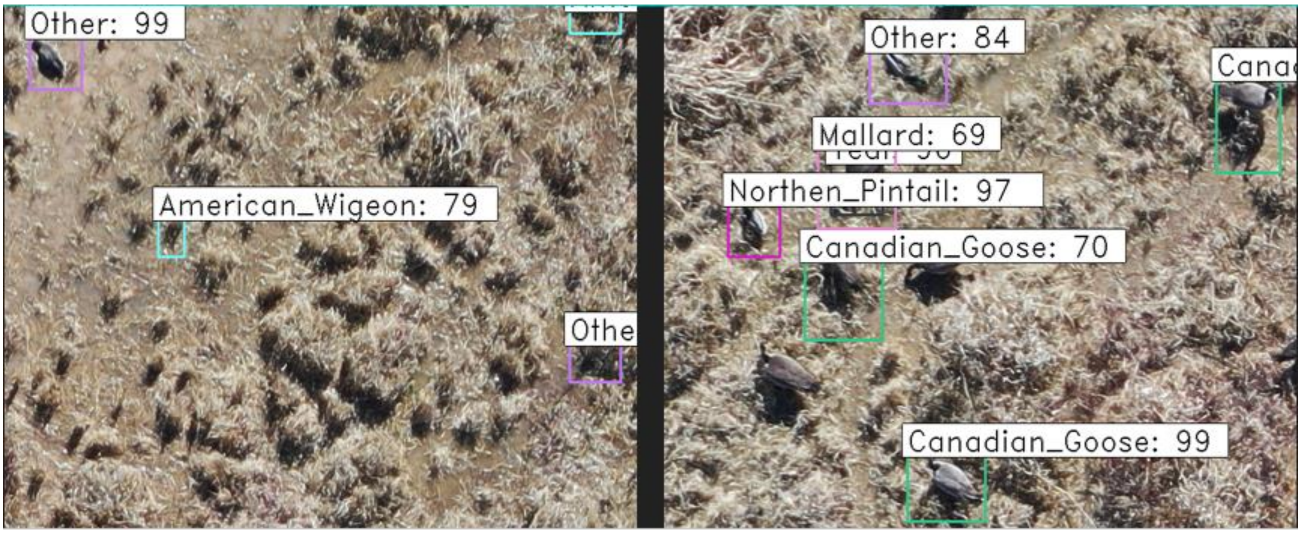


Figure 42: sample of Faster R-CNN results for sub-species

5. Results and error analysis

In this chapter, we evaluate the results of CNN on the waterfowl (counting), species and sub-species. To evaluate the results properly, we use cross validation, where each prediction label is compared to the corresponding ground truth label then summarize the results in a confusion matrix. We performed three assessments in this chapter. The first assessment is the count performance, where we test how the accuracy changes in the different levels. The second and third assessments are to evaluate CNN’s ability to classify on the species and sub-species level

Count Performance:

The first assessment of the results is the waterfowl count performance. A waterfowl count is defined as a generated prediction label that bounds a labelled waterfowl in the test set. This assessment can be done to all levels (sub-species, species, and waterfowl) to obtain insights on what is the effect of increasing the number of classes on the general counting performance.

It should be mentioned here that number of detection labels generated in the results does not necessarily match the number of ground truth labels in the test set. A ground truth label can be undetected and additional labels can be generated that do not refer to a target object. Thus, the assessment of the results is to be carried in terms of Waterfowl Labels, Undetected Waterfowl, and Non-Waterfowl Detected. Figure 43 shows the counting performance for all CNNs against the number of classes per level.

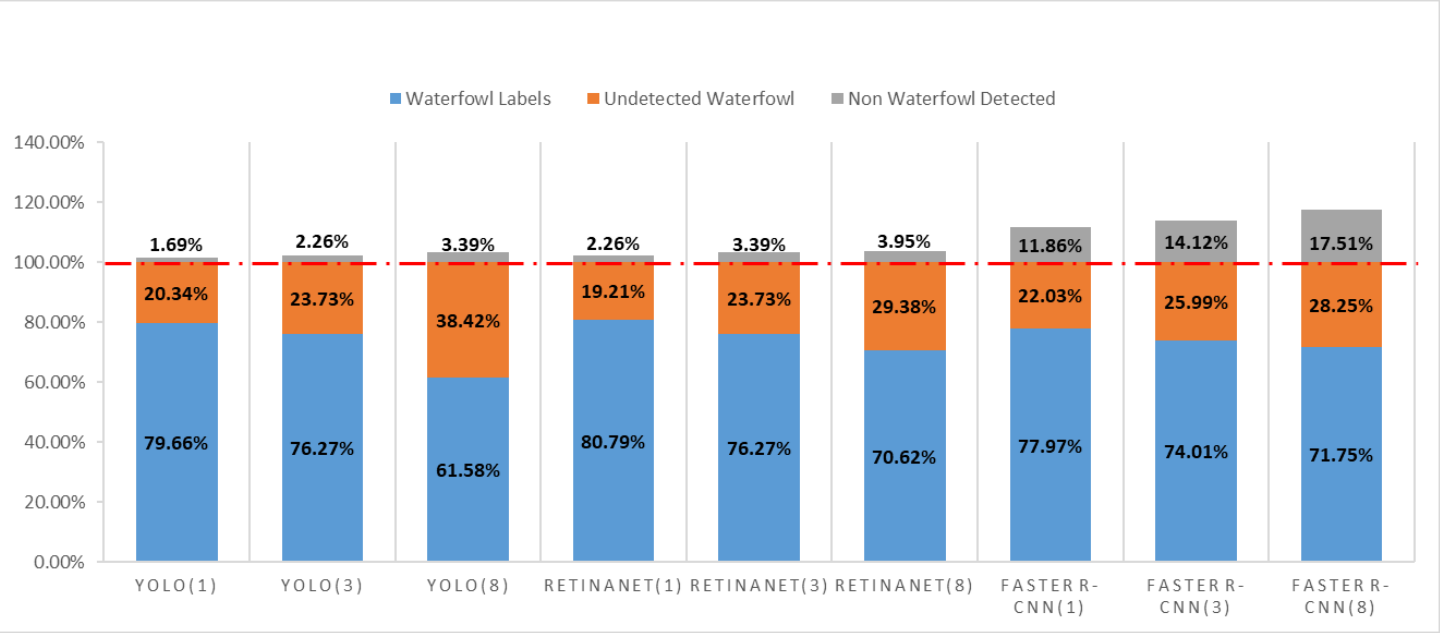


Figure43: Count Performance

We can see that increasing the number of classes to be detected limits the ability of CNNs to detect waterfowls and increases the number of undetected waterfowl. Also, this change increases the number of non-waterfowl species detected.

One of the useful insights that can be derived from the count results is the consistency of the number of labels generated by a CNN against different levels. Consistency does not necessarily mean better accuracy but when the CNN is trained on a much larger dataset, it can be an indication of the ability of a the CNN to still capture the same number of targeted objects as the number of classes to be detected increases. As seen in table 8 Faster R-CNN generated the most consistent labels whereas YOLO and Retinanet behaved fairly the same.

Table8: Number of labels generated for each CNN

CNN	1-CLASS	3-CLASS	8-CLASS
Faster R-CNN	159	156	158
Retinanet	147	141	132
YOLO	144	139	115

Species Level:

The second assessment is the Duck, Goose and Crane detection accuracy. The confusion matrix reveals how the CNN behaved in detection and if a certain class was confused with another class.

Table9: YOLO confusion matric for species level

Class	Duck	Goose	Crane	undetected	Ground Truth Total	non WF
Duck	95	14	0	34	143	**3**
Goose	1	21	0	7	29	**1**
Crane	0	0	4	1	5	**0**
Detection Total	**96**	**35**	**4**	**42**	**177**	**4**

Table10: Retinanet confusion matric for species level

Class	Duck	Goose	Crane	undetected	Grand Truth Total	non WF
Duck	106	4	0	33	143	**6**
Goose	4	17	1	7	29	**0**
Crane	0	0	3	2	5	**0**
Detection Total	**110**	**21**	**4**	**42**	**177**	**6**

Table11: Faster R-CNN confusion matric for species level

Class	Duck	Goose	Crane	undetected	Grand Truth Total	non WF
Duck	106	3	0	34	143	**25**
Goose	8	10	0	11	29	**0**
Crane	1	0	3	1	5	**0**
Detection Total	**115**	**13**	**3**	**46**	**177**	**25**

From the above matrices, we can notice that class Duck was confused with class Goose but not Crane by all CNNs. This is a result of the very different look cranes have from ducks and geese.

Next, per CNN, each class would be categorized as Correctly Classified, Incorrectly Classified, Undetected, and non-Waterfowl Detected. The non-Waterfowl Detected category refers to how many times a class was confused with non-Waterfowl surroundings.

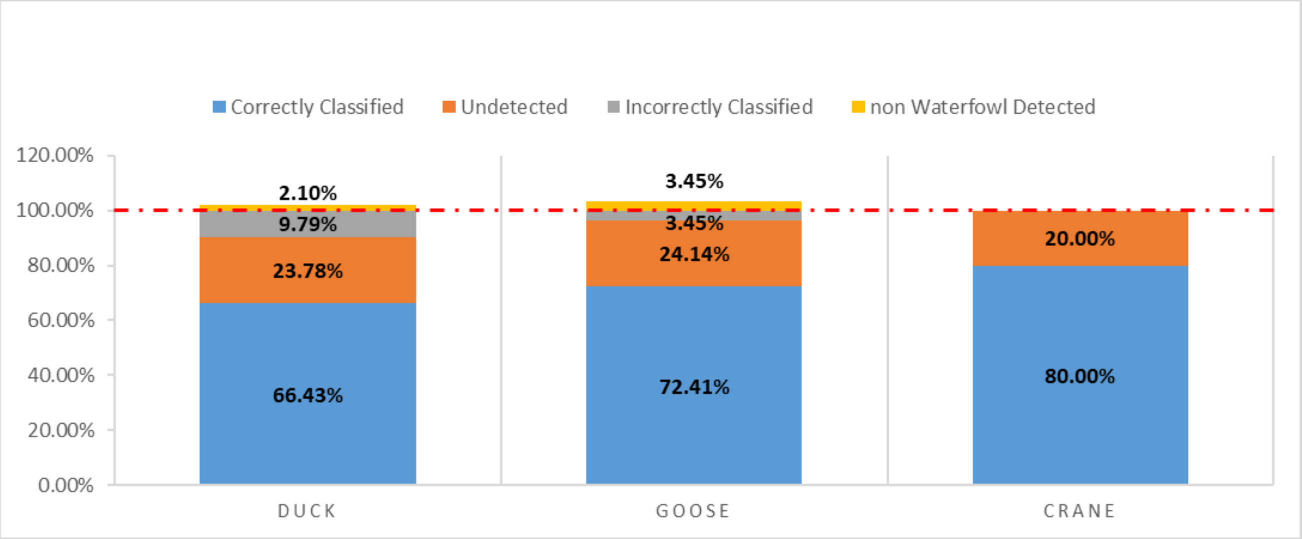


Figure44: YOLO species level Performance (Class vs Accuracy Measurements)

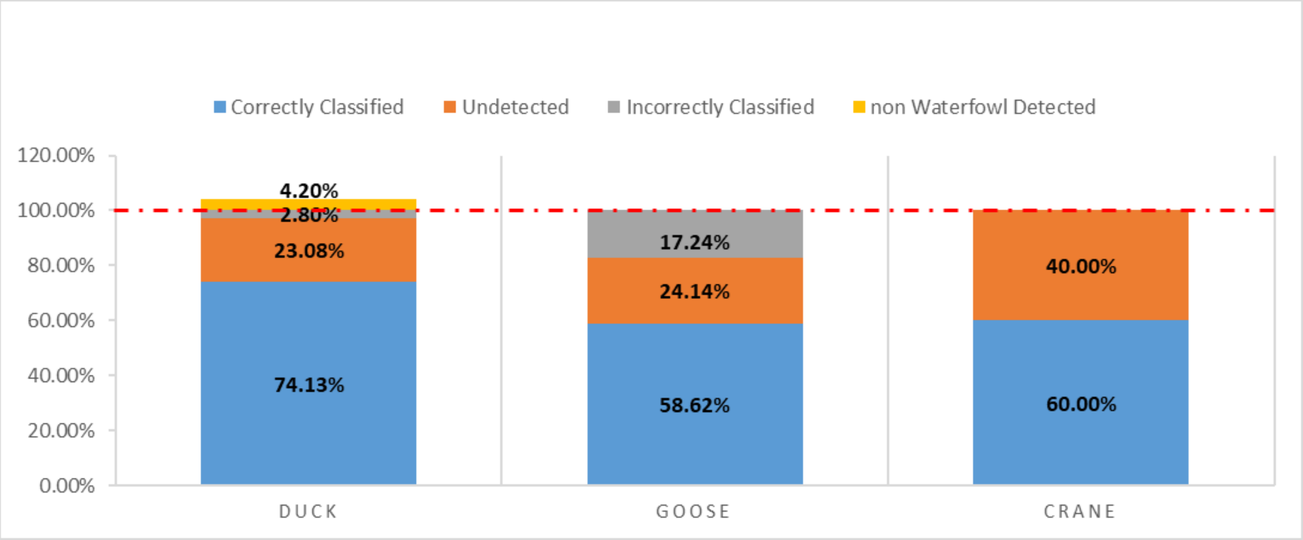


Figure45: Retinanet species level Performance (Class vs Accuracy Measurements)

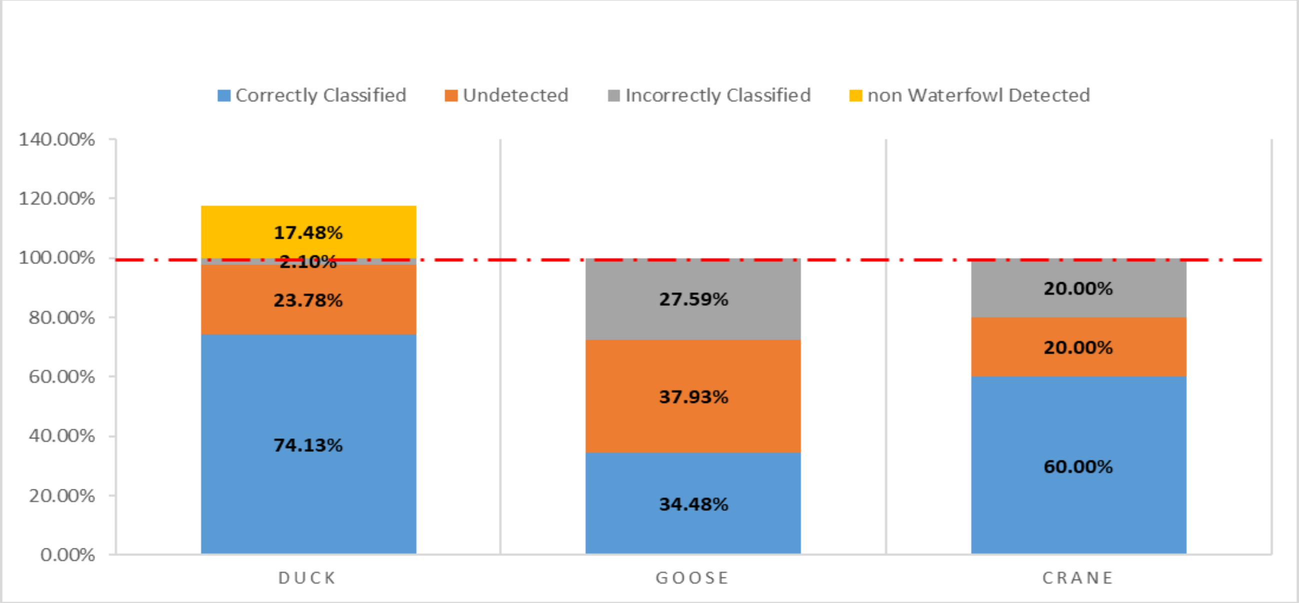


Figure46: Retinanet species level Performance (Class vs Accuracy Measurements)

Sub-species Level:

This level will be assessed in a similar manner to the species Level.

Table 12: YOLO confusion matric for sub-species level

Class	American_Wigeon	Canadian_Goose	Gadwall	Mallard	Northen_Pintail	Other	Sandhil_Crane	Teal	undetected	Grand Truth Total	non WF
American_Wigeon	0	1	0	0	2	1	0	0	4	8	0
Canadian_Goose	0	28	0	0	0	0	1	0	0	29	4
Gadwall	0	2	0	0	1	0	0	1	2	6	0
Mallard	0	8	0	13	23	4	0	1	46	95	0
Northen_Pintail	0	2	0	1	2	0	1	1	2	9	1
Other	0	3	0	0	5	0	0	0	10	18	0
Sandhil_Crane	0	1	0	0	0	0	4	0	0	5	1
Teal	0	0	0	0	3	0	0	0	4	7	0
Detection Total	**0**	**45**	**0**	**14**	**36**	**5**	**6**	**3**	**68**	**177**	**6**

Table 13: Retinanet confusion matric for sub-species level

Class	American_Wigeon	Canadian_Goose	Gadwall	Mallard	Northen_Pintail	Other	Sandhil_Crane	Teal	undetected	Grand Truth Total	non WF
American_Wigeon	2	0	2	0	0	0	0	2	2	8	3
Canadian_Goose	0	20	1	0	0	1	1	1	5	29	1
Gadwall	2	0	0		0	1	0	1	2	6	1
Mallard	24	0	14	10	5	1	0	13	28	95	0
Northen_Pintail	3	0	0		1	2	0		3	9	0
Other	4	0	1	2	0	1	0	3	7	18	0
Sandhil_Crane	0	0	0		0	0	3		2	5	0
Teal	1	0	3		0	0	0		3	7	2
Detection Total	**36**	**20**	**21**	**12**	**6**	**6**	**4**	**20**	**52**	**177**	**7**

Table 14: Faster R-CNN confusion matric for sub-species level

Class	American_Wigeon	Canadian_Goose	Gadwall	Mallard	Northen_Pintail	Other	Sandhil_Crane	Teal	undetected	Grand Truth Total	non WF
American_Wigeon	0	0	1	3	3	0	0	1	0	8	13
Canadian_Goose	2	12	1	0	0	4	0	4	6	29	0
Gadwall	0	0	1	1	0	2	0	1	1	6	2
Mallard	4	1	2	13	25	10	0	12	28	95	0
Northen_Pintail	0	0	0	1	0	4	0	1	3	9	1
Other	1	0	0	1	0	5	0	1	10	18	15
Sandhil_Crane	1	0	0	0	0	0	3	0	1	5	0
Teal	0	0	0	0	3	2	0	1	1	7	0
Detection Total	**8**	**13**	**5**	**19**	**31**	**27**	**3**	**21**	**50**	**177**	**31**

As we have six out of the eight classes belonging to super class Duck. It is no surprise that these sub classes were highly confused with each other as opposed to classes Canadian Goose and Sandhill Crane.

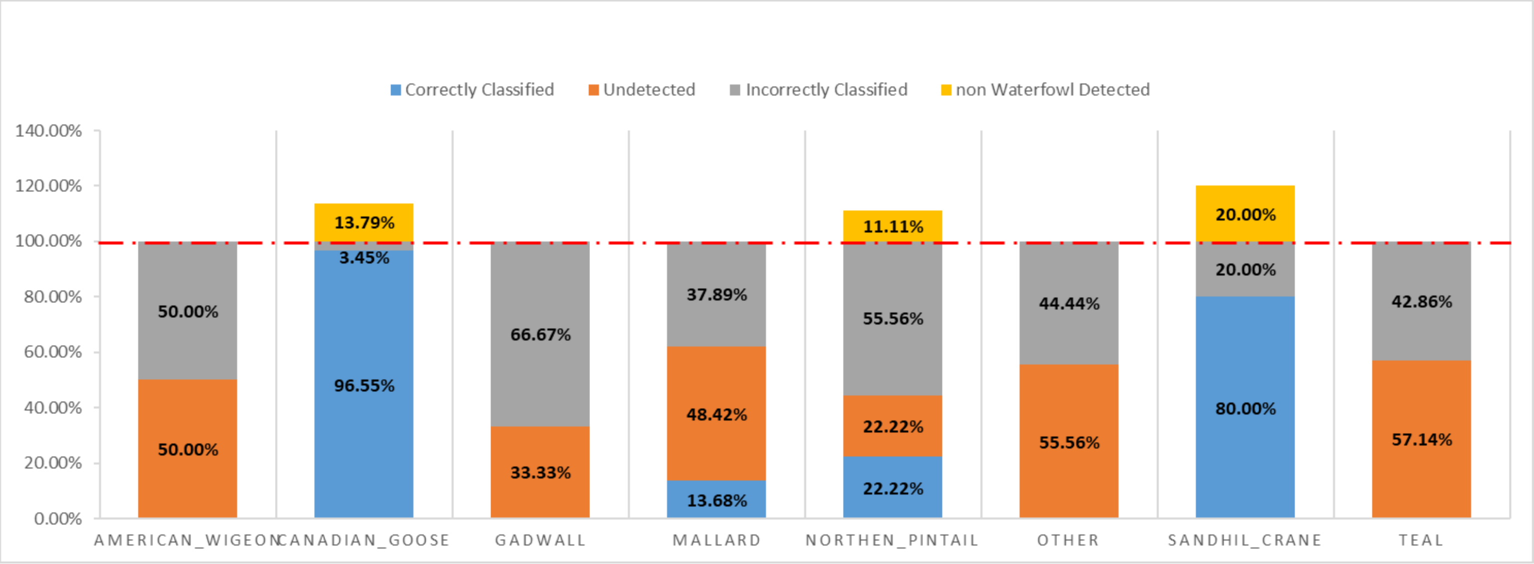


Figure47: YOLO sub-species level Performance (Class vs Accuracy Measurements)

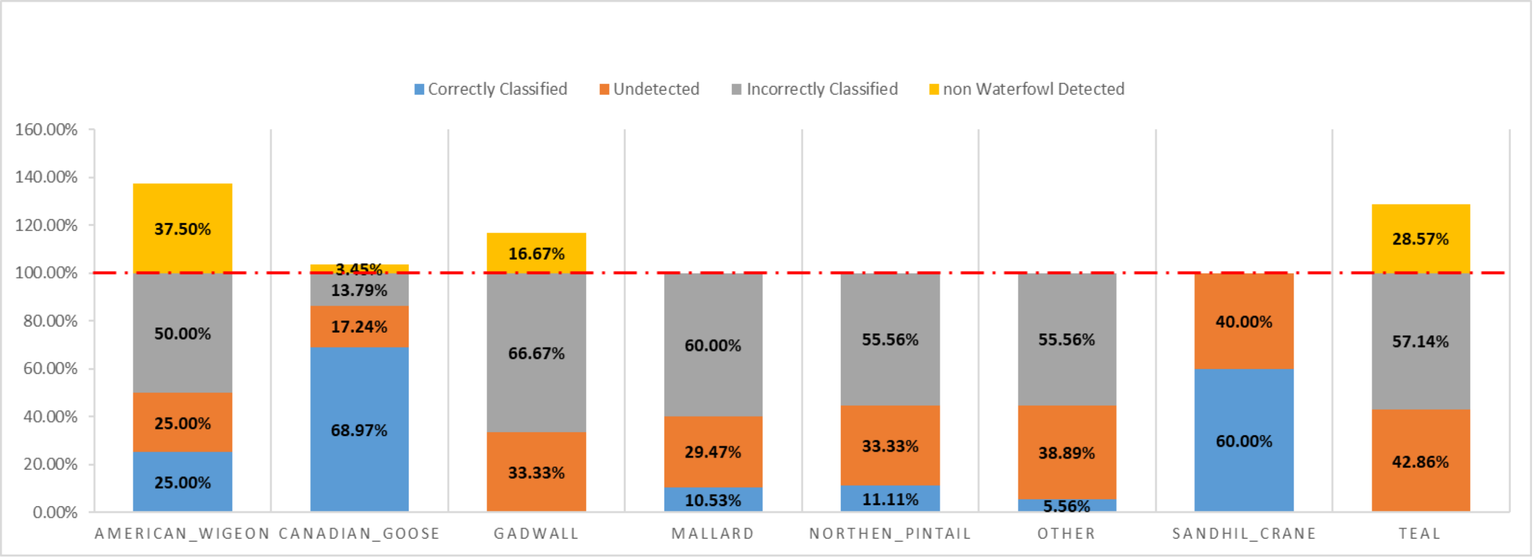


Figure48: Retinanet sub-species level Performance (Class vs Accuracy Measurements)

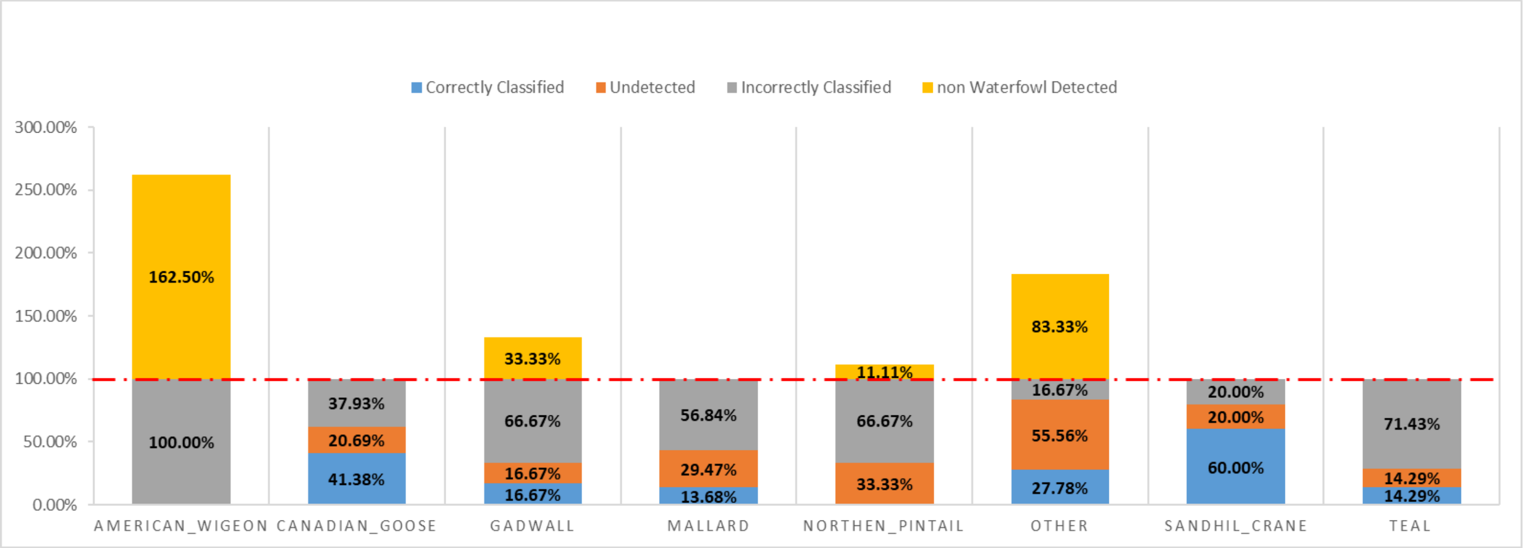


Figure49: Faster R-CNN sub-species level Performance (Class vs Accuracy Measurements)

Population density and surroundings:

We can see from figure 50 that CNNs' performance more limited in images with high population density where almost the same number of labels were generated for images with different population densities (High density on the left and Low density on the right).

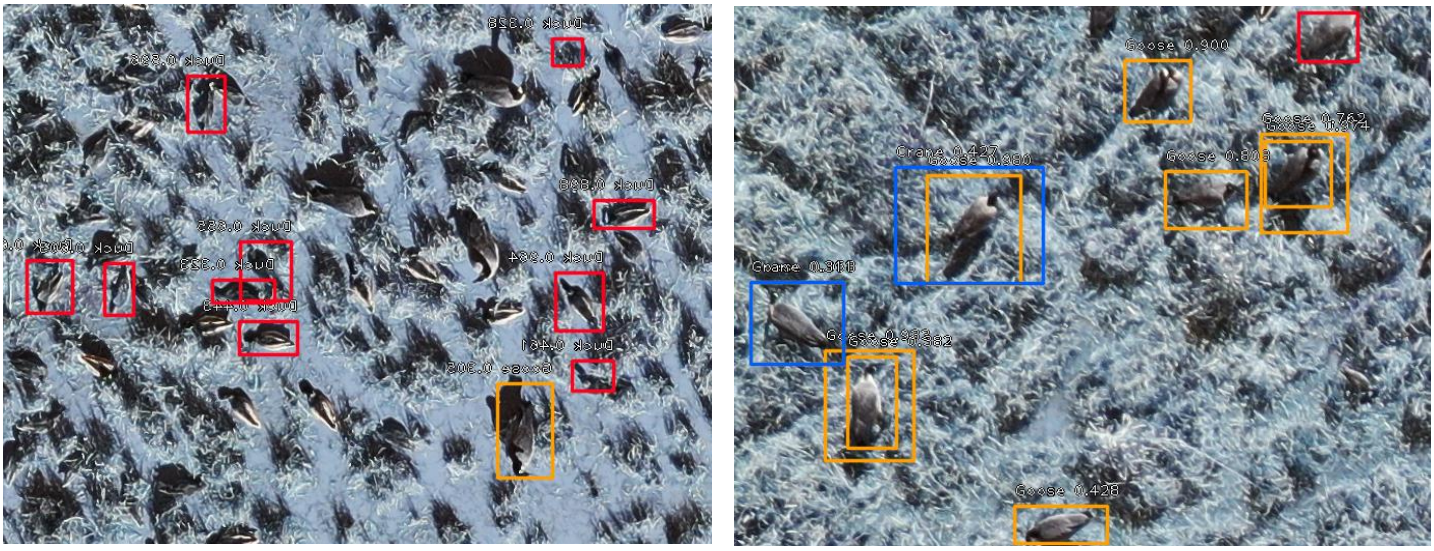


Figure 50: label generation in different population densities (High on the left and low on the right)

Also, better counting was recorded with clean surroundings (less shadows, plants etc.) as seen in figure 51

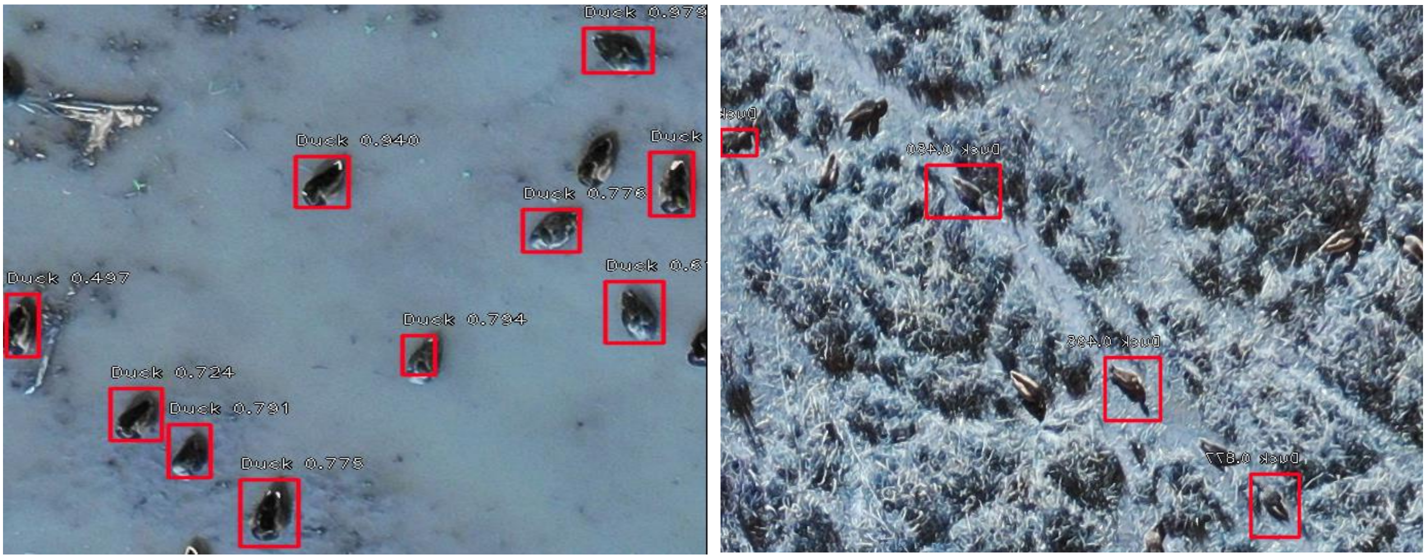


Figure 51: detection ability in different surrounding setup (clear on the left and rough on the right)

6. Discussion

We will discuss the findings of this project as questions, we will use the insights of the results to answer questions regarding the nature of the waterfowl dataset, the behaviour of the CNN's and the evaluation of the results.

What are the special characteristics in the waterfowl dataset? How do these characteristics affect the implementation, especially preprocessing? What is the effect of crowdsourcing (LabelBox) on the quality of the data?

The waterfowl dataset is an aerial imaging data captured by ultra-high-resolution sensors. This, combined with nature of the waterfowl species, produced large images with very small target objects that need to be detected. As the input image gets resized by the CNN, we noticed a limitation in the ability detection in all implemented CNN's. As seen in figure 32, the waterfowl in the original image cover a very small percentage of total image area (0.014%) and therefore, when the image is resized by the CNN to say 416x416, it becomes very hard to detect and harder to classify as seen below in figure 52.

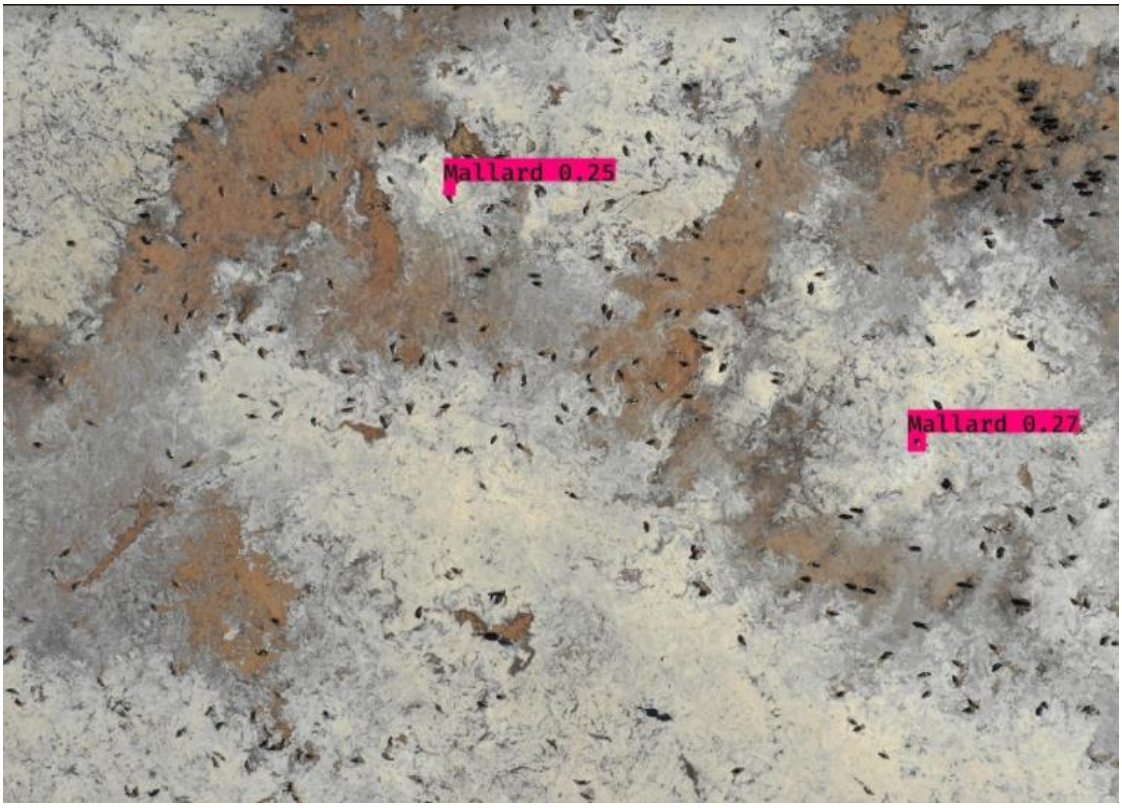


Figure52: Limitation of detection performance in original images

Even though the above image has plenty of waterfowl species, the CNN (YOLO in this image) could only detect two of them. This has forced us to crop the images such that we have the target object cover a sufficient area in the input image so that they are identifiable by the CNN. It is clear to us that this is an essential step in the pre-processing of dataset of any aerial imaging of waterfowl in the application of detection if taken at a comparable drone altitude to that done by USFWS (40 m).

Another important feature in waterfowl dataset is the vertical view of the object. This indeed adds a difficulty to the detection, especially, when the target objects have large degree of similarity (waterfowl species).

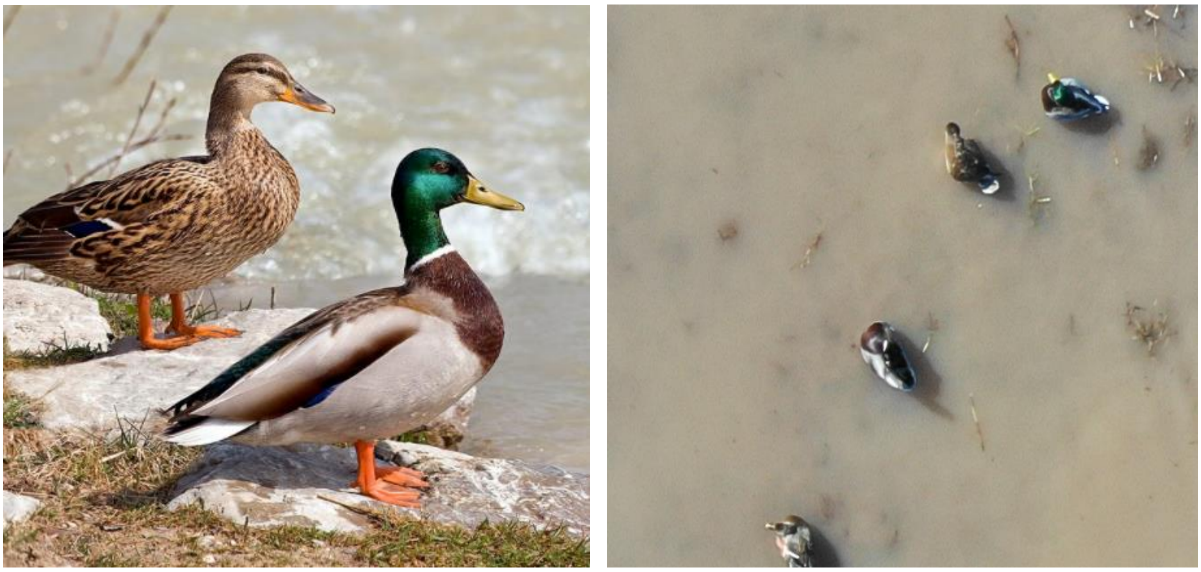


Figure 53: Horizontal view on the left and vertical view (aerial imagery) on the right

Because if that, we can say that in the case of detection in waterfowl aerial imaging, the dataset should be larger than that used in detection in normal images to produce high accuracy level (>90%).

The second part of the dataset is the LabelBox labels. These has been done by several experts to make sure the data is labelled properly. Since every expert label all the images, this has caused almost all waterfowl objects to be labelled multiple times. As seen in figure 31, birds are labelled with different bounding box sizes. The larger the label is, the larger the amount of noise that will be introduced to the learning process gets. Thus, we managed that by selecting the smallest label amongst those who intersect over one object. This method of selection does not work sufficiently unless we assume two facts. First, is that all labels cover the area of the waterfowl. Second is that all experts agree on the type of the waterfowl. Without those two assumptions, we cannot proceed with the data pre-processing and the dataset becomes unusable by the CNN as we do not have an exact answer on what is the type of each captured waterfowl.

it is beneficial to the training process to have a balanced training set where each class has enough, and similar training sample count compared to the other classes. However, since we have one dataset but three training levels, it is almost impossible to balance the classes in the three training levels. For example, we had 7.9% and 9% of total training samples from classes "Canadian Goose" and "Gadwall" respectively. Given that, we can say that the CNN will learn about "Canadian Goose" as much as what it would learn about "Gadwall". But when we move to the species level, "Gadwall" belongs to the "Duck" class, which has 88% share of total labels while class "Goose" shares 7.9% only causing class imbalance.

Whet benefits can we gain from automating the re-formatting of the labels from LabelBox JSON to Darknet?

We achieved a solution by automating the label extraction procedure by converting the LabelBox JSON labels to Darknet format using a Python script. The major advantage of this script is that it can be scaled to bigger label files to ease preprocessing. This is important because it paves the way to use much more training data without being concerned about the effort needed for preprocessing. Adding much more data is something waterfowl dataset needs to prevent confusion between different bird types that look similar and decrease the number of undetected objects.

What are the main insights that are taken from the results? How changing the detection level between waterfowl to sub-species affected the performance?

The first observation on the results was the effect of increasing classes on the number of labels generated by the CNN's. The number of detected waterfowl decreased as we increase the number of classes to be detected (waterfowl level to sub-species level). This is due to the decrement in the number of training examples per waterfowl type as we segregate waterfowl species. For example, when we train the net work on a single class of waterfowl, the single class will have the largest amount of training examples which gives the CNN a chance to capture general patterns that are common to all types of waterfowl, whereas in the species and sub-species level, the CNN tries to capture class-specific patterns with less amount of training samples.

The second observation is that result images often have waterfowls that are labelled multiple times e.g. figure 36. This due to the limited ability of the CNN to distinguish between different types of waterfowl species. This limitation ca be fixed by inserting more data for each class.

The third insight is the effect of surroundings and population density on the performance. As seen in figures 50 and 51, it was clear that better performance is achieved in low population densities and clear surroundings (no shadows, plants etc.). it is an interesting question to ask surveyors about how controllable these two features are and if there are methods to capture waterfowl species apart from each other or avoid areas with rough surroundings.

How does the two-stage detector (Faster R-CNN) differ from the single-stage detector (YOLO and Retinanet). What is the impact of that on the results?

In the case of Faster-RCNN, it uses the first stage to perform binary classification of (target or background) to recognize general patterns that apply to all classes, then a second stage learns class-specific patterns and performs classification. This caused Faster R-CNN to generate a consistent number of labels for the three implementations. In the case of YOLO and Retinanet, they learn to recognize patterns that are specific to each class. Combined with further dividing the dataset into more classes, the number of training samples for a class decreases and thus fewer patterns for these objects can be recognized. This has caused the CNNs to be confused between different types and in some cases not being able to detect a bird due to insufficient patterns supplied in the training samples. Nevertheless, the accuracy of detection has also decreased with Faster R-CNN less severely than YOLO and Retinanet.

Why do the confusion matrices look untraditional? What is the effect of that on the evaluation process?

The confusion matrices generated after evaluation have a non-traditional setup. The number of detected objects does not match the number of objects in the ground truth. Some objects were not detected, and some others were detected but they do not represent a waterfowl. This has forced us to add an undetected and non-waterfowl class which have no ground truth reference. We reported the results in terms of correct/incorrect classification, undetected waterfowl, and non-waterfowl detection.

7. Conclusions

In this thesis, we explored the use of a modern state of the art CNNs for detection and classification of waterfowl UAS imagery. We discovered that waterfowl data has special characteristics such as the relative size of the target object, class similarity, and label formatting. We essentially need to take those characteristics into consideration for adequate preprocessing. We were able to utilize the LabelBox dataset of 13 images of size 5472x3648 labelled by 13 experts and fed it to three CNNs (YOLO, Retinanet and Faster R-CNN). We performed waterfowl, species and sub-species level prediction for each CNN, and it was noted in the results that, the confidence values of labels are close and this has caused multiple labelling problem that required post-processing. We also discovered that increasing the number of classes limits the accuracy of all CNNs as well as the number of labels generated by single-stage detectors (YOLO and Retinanet) as opposed to the two-stage detector (Faster-RCNN). Moreover, increasing the number of classes resulted in a higher number if undetected objects as well as non-waterfowl object detection. Finally, noted performance degradation in images with high population densities and images that have unclear surroundings such as shadows and plants.

8. Future work

In this project, we have collected valuable insights on the dataset level as well as the implementation and evaluation levels. However, there are still open. It is very important to investigate how a much larger waterfowl datasets will change the results and the behaviour of the CNN's. Also, how efficient this model is, when employed with different study areas with or different waterfowl types. One of the major improvements that could be done to the implementation is to investigate the possibility of entering aerial images of waterfowl to the CNN in their original size. The benefit of keeping images in their original size is to reduce the pre-processing task so the model becomes more efficient and eases the usage of the model. Moreover, if we want to solve this problem, we must investigate the computational feasibility entering ultra-high-resolution images. For example, what are the memory requirements and if it can be done as a modification of state-of-the-art CNN's or should we build a CNN architecture from scratch where the design of the hyperparameters is oriented toward this application domain. Finally, it is very important to improve the ability of the model to solve the class imbalance problem. As we can see in section 4.1, that the number of training samples for minor classes (such as goose and crane in the species level) is way less than the major class (duck). It would be beneficial to investigate the data manipulation and augmentation techniques to increase the number of training samples for these classes and therefore increase accuracy. Moreover, in section 3.2, we mentioned that 4 sub-species classes were not taken into consideration in the training process because they do not have a representative number of training samples. We believe that using data augmentation techniques to increase those classes represents a crucial benefit in applications where detecting those rare classes is a prime interest.

References

Abd-Elrahman, A., Pearlstine, L. and Percival, F., 2005. Development of pattern recognition algorithm for automatic bird detection from unmanned aerial vehicle imagery. *Surveying and Land Information Science*, *65*(1), p.37.

Aloysius, N. and Geetha, M., 2017, April. A review on deep convolutional neural networks. In *2017 International Conference on Communication and Signal Processing (ICCSP)* (pp. 0588-0592). IEEE.

Chabot, D. and Bird, D.M., 2012. Evaluation of an off-the-shelf unmanned aircraft system for surveying flocks of geese. *Waterbirds*, *35*(1), pp.170-174.

Chabot, D. and Francis, C.M., 2016. Computer-automated bird detection and counts in high-resolution aerial images: a review. *Journal of Field Ornithology*, *87*(4), pp.343-359.

Chen, K., Loy, C.C., Gong, S. and Xiang, T., 2012, September. Feature mining for localised crowd counting. In *BMVC* (Vol. 1, No. 2, p. 3).

Chen, Y., Li, W., Sakaridis, C., Dai, D. and Van Gool, L., 2018. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3339-3348).

Ferdowsi, M., Zargar, B., Ponci, F. and Monti, A., 2014, September. Design considerations for artificial neural network-based estimators in monitoring of distribution systems. In *2014 IEEE International Workshop on Applied Measurements for Power Systems Proceedings (AMPS)* (pp. 1-6). IEEE.

Girshick, R., 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).

Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

Grenzdörffer, G.J., 2013. UAS-based automatic bird count of a common gull colony. *International archives of the photogrammetry, Remote sensing and spatial information sciences*, *1*, p.W2.

Hodgson, J.C., Baylis, S.M., Mott, R., Herrod, A. and Clarke, R.H., 2016. Precision wildlife monitoring using unmanned aerial vehicles. *Scientific reports*, *6*(1), pp.1-7.

Hong, S.J., Han, Y., Kim, S.Y., Lee, A.Y. and Kim, G., 2019. Application of Deep-Learning Methods to Bird Detection Using Unmanned Aerial Vehicle Imagery. *Sensors*, *19*(7), p.1651.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. and Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Kingsford, R.T., 1999. Aerial survey of waterbirds on wetlands as a measure of river and floodplain health. *Freshwater Biology*, *41*(2), pp.425-438.

Laliberte, A.S. and Ripple, W.J., 2003. Automated wildlife counts from remotely sensed imagery. *Wildlife Society Bulletin*, pp.362-371.

Lin, T.Y., Goyal, P., Girshick, R., He, K. and Dollár, P., 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980-2988).

Linchant, J., Lisein, J., Semeki, J., Lejeune, P. and Vermeulen, C., 2015. Are unmanned aircraft systems (UAS s) the future of wildlife monitoring? A review of accomplishments and challenges. *Mammal Review*, *45*(4), pp.239-252.6

Mitchell, T.M., 2006. *The discipline of machine learning* (Vol. 9). Pittsburgh: Carnegie Mellon University, School of Computer Science, Machine Learning Department.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91-99).

Rothe, R., Guillaumin, M. and Van Gool, L., 2014, November. Non-maximum suppression for object detection by passing messages between windows. In *Asian conference on computer vision* (pp. 290-306). Springer, Cham.

Sasse, D.B., 2003. Job-related mortality of wildlife workers in the United States, 1937-2000. *Wildlife society bulletin*, pp.1015-1020.

Soviany, P. and Ionescu, R.T., 2018, September. Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction. In *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (pp. 209-214). IEEE.

van Gemert, J.C., Verschoor, C.R., Mettes, P., Epema, K., Koh, L.P. and Wich, S., 2014, September. Nature conservation drones for automatic localization and counting of animals. In *European Conference on Computer Vision* (pp. 255-270). Springer, Cham.

Wilson, R.P., Culik, B., Danfeld, R. and Adelung, D., 1991. People in Antarctica—how much do Adélie Penguins Pygoscelis adeliae care? *Polar Biology*, *11*(6), pp.363-370.

Zha, S., Luisier, F., Andrews, W., Srivastava, N. and Salakhutdinov, R., 2015. Exploiting image-trained CNN architectures for unconstrained video classification. *arXiv preprint arXiv:1503.04144*.