Counting Sheep in Aerial Imagery Under Adverse Conditions

Bachelor Thesis
Faculty of Science, University of Bern

submitted by

Lars M. Wüthrich

from Jegenstorf, Switzerland

Supervision:

PD Dr. Kaspar Riesen
Institute of Computer Science (INF)
University of Bern, Switzerland

Abstract

Building upon previous research focused on counting sheep on grassland captured by unmanned aerial vehicles (UAVs), this thesis aims to expand the scope to counting sheep in aerial imagery under adverse conditions using common object detection algorithms. The goal is to assess the accuracy of Faster R-CNN, RetinaNet, and YOLOv9 in counting sheep in challenging environments and to evaluate whether their accuracy is sufficient for real-world applications. Additionally, the necessity of datasets featuring sheep in adverse conditions is evaluated. It is important to note that this thesis focuses only on counting sheep in individual images, not on entire pastures.

This study develops novel datasets featuring sheep on snow, during dusk, and in rough terrain. Additionally, an existing dataset of sheep on grassland is used for comparison. These datasets are used to train and test the algorithms in various configurations.

The results indicate that detectors trained solely on grassland images perform poorly in adverse conditions, achieving a counting accuracy below 0.4. However, detectors trained specifically on adverse conditions perform significantly better, with counting accuracies between 0.9 and 0.94, although still below the 0.95 threshold deemed necessary for real-world applications. Generalized models trained on all datasets together perform similarly or better as specialized models, suggesting that a single comprehensive model could suffice for varied conditions.

The study also explores a simple method to enhance performance by averaging predictions from multiple images taken from different perspectives, improving the counting accuracy to above 0.95 in snowy conditions.

Despite the detectors struggling to accurately detect sheep in extremely adverse conditions, the results suggest that UAVs and object detection algorithms are viable for sheep counting under moderately adverse conditions, such as on snow-covered ground and at early dusk. This indicates potential for practical applications in improving the efficiency of livestock monitoring.

Acknowledgements

I am grateful to the University of Bern for supplying the UAV used to capture the novel datasets and for providing computation time on UBELIX, their high-performance computing cluster. Additionally, I extend my thanks to Esther Bader for her assistance with image annotation and to PD Dr. Kaspar Riesen for supervising this thesis.

Contents

1 Introduction **1**

2 Background and Related Work **4**

2.1 Related Work 4

2.2 Object Detection Using CNNs 8

2.2.1 Introduction to ANNs 8

2.2.2 Basic Principles of CNNs 9

2.2.3 Two-Stage Detectors 13

2.2.4 One-Stage Detectors 14

2.3 Comparison of Selected Algorithms 15

2.3.1 Faster R-CNN 15

2.3.2 RetinaNet 16

2.3.3 YOLOv9 17

3 Novel Datasets **18**

3.1 Grassland 18

3.2 Snow 19

3.3 Dusk 20

3.4 Rough Terrain 21

4 Empirical Evaluation **22**

4.1 Experimental Setup 22

4.1.1 Train-Test Split 22

4.1.2 Hyperparameter Tuning 23

4.1.3 Performance Metrics 24

4.2 Results and Discussion 25

4.2.1 Quantitative Results 25

4.2.2 Qualitative Results 27

4.2.3 Simple CA Performance Improvement 29

5 Conclusions and Future Work **31**
5.1 Conclusions 31
5.2 Future Work 33

A Best Hyperparameters **34**
A.1 Faster R-CNN 34
A.2 RetinaNet 34
A.3 YOLOv9 35

Bibliography **37**

Chapter 1

Introduction

The present thesis is located in the field of artificial intelligence (AI). This is a field of research in computer science with the goal of developing machines capable of performing tasks that would necessitate intelligence if carried out by humans. According to Negnevitsky [1], intelligence is "the ability to learn and understand, to solve problems and to make decisions." Such tasks include reasoning, generalizing, understanding natural language, and processing visual information. Contributions to AI come from diverse fields, including mathematics, neuroscience, psychology, and computer engineering [2].

Among the various subfields of AI lies computer vision, which focuses on the development of algorithms that enable the analysis and understanding of real-world environments through the use of cameras [3]. It finds application in a wide range of domains, such as autonomous driving, where it is crucial for the detection of lanes and nearby vehicles, thereby enabling navigation on the road. In security, facial recognition systems rely on computer vision to accurately identify individuals and either grant or deny access to private property. Furthermore, in healthcare, computer vision plays a vital role in medical image analysis, assisting in the diagnostic process.

Computer vision comprises various subdomains, each focusing on a specific task:

- Human pose estimation seeks to locate human body parts in an image and create a representation of the body's configuration, position, and orientation. It is applied in areas like human-computer interaction and athletic training [4].
- Object tracking aims to estimate the trajectory of objects through successive video frames. It finds applications in automated surveillance, traffic monitoring, and sports [5].

- Object detection strives to identify objects within an image and to determine their precise location, which is typically marked by bounding boxes [6]. It thereby answers the very important question of what objects are where. Since this information is vital for the semantic understanding of images, object detection provides a basis for various other computer vision tasks, such as object tracking. It is applied in many real-world areas, including surveillance, autonomous driving, and agriculture [7].

Nowadays, most object detection algorithms use convolutional neural networks (CNNs), which are a type of artificial neural network (ANN). They are computational processing systems inspired by biological nervous systems, such as the human brain. That means ANNs are comprised of interconnected computational nodes, called neurons, that self-optimize through learning. Developed in the field of machine learning, ANNs are a method to solve a wide variety of problems, often far exceeding the performance of previous approaches. Convolutional neural networks are a specialized form of ANNs tailored to learning robust and high-level feature representations from images [8].

In recent years, the integration of AI into agricultural practices has expanded significantly, enhancing the capability to explore, understand, and analyze agricultural data. This has drastically improved decision-making in the field. A notable data source are unmanned aerial vehicles (UAVs), commonly known as drones. Unmanned aerial vehicles and AI are increasingly utilized for various purposes, thereby aiding in the optimization of agricultural operations [9]:

- Assessing different plant stresses, such as water scarcity, diseases, nutrient deficiencies, and pest infestations, through the use of multi-spectral and thermal imaging sensors [10].
- Mapping weed distribution to develop site-specific herbicide application plans, which helps to reduce herbicide use and more effectively target weed hotspots [11].
- Monitoring livestock, where animals are not only autonomously detected and counted, but their distribution, health, and behavior are also assessed to enhance animal welfare [12].

The subdivision of AI into its various fields and subfields is depicted in Figure 1.1. It is important to note that each level encompasses a broader range of fields than illustrated and that some areas overlap. For example, object detection uses CNNs. The present thesis is located in the subdomain of object detection, applied to an agricultural context, specifically livestock monitoring.

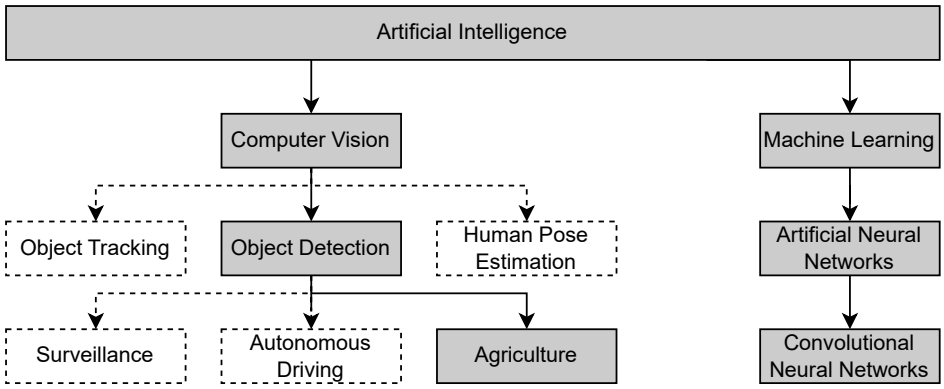


Figure 1.1: The hierarchy of selected subfields in AI

Currently, most farmers rely on manually counting their animals. This process is time-consuming and prone to errors. Especially large herds get counted rather infrequently, with some farmers not knowing the number of sheep on their pasture at all. Unmanned aerial vehicles, in conjunction with object detection technology, could help overcome these problems by providing a fast, frequent, and non-disruptive method for counting animals. This would allow farmers to detect any loss of animals sooner and enable faster responses to mitigate further losses.

Numerous studies examine the capabilities of object detection systems to count sheep in aerial imagery captured by UAVs. While the results are promising, the research has largely focused on images of white sheep on green grassland [13].

The goal of this thesis is to assess the accuracy of common object detection algorithms in counting sheep on aerial imagery under adverse conditions. The results are evaluated to determine whether the observed performance is sufficient for real-world applications under less-than-ideal conditions. To address this question, three novel datasets comprising aerial imagery of sheep on snow, during dusk, and in rough terrain are being developed as part of this thesis. The algorithms are tested under various configurations of training and test data to determine the necessity of new datasets in comparison to an already existing grassland dataset. In addition, a simple method to increase the counting accuracy (CA) is being explored.

The following chapter introduces the topic of object detection, which includes an explanation of CNNs, as well as a comparison of the employed algorithms. Chapter 3 describes the used datasets with a special emphasis on the creation of the snow, dusk, and rough terrain datasets. Following that, Chapter 4 discusses the empirical evaluation of the selected object detection algorithms on the four datasets. Finally, Chapter 5 concludes this thesis and provides an outlook on future research.

Chapter 2

Background and Related Work

This chapter introduces the theoretical background needed for the rest of this thesis. Section 2.1 reviews existing research concerning object detection, UAVs, and livestock. Subsequently, Section 2.2 strives to explain the general principles of object detection algorithms, incorporating an introduction to CNNs. Following this, Section 2.3 examines the selection of object detection algorithms which are used in the evaluation and provides a comparison of them.

2.1 Related Work

All papers reviewed in this section are related to agriculture and fall into at least two of the following three categories: object detection, UAVs, and livestock, as depicted in Figure 2.1.

The first three presented studies are located at the intersection between two of the three topics, while the latter four reports represent the intersection of all three topics.

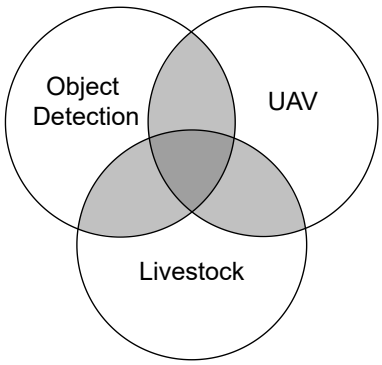


Figure 2.1: The organization of related work reviewed in this section

Unmanned Aerial Vehicles are widely used to manually observe animals without using AI. For instance, Nyamuryekung'e et al. [14] assess the potential of UAV video monitoring to predict the consumption of specific food items by rangeland beef cows in a controlled foraging environment. The experimental setup features twelve feed containers arranged in an open semi-circle, each spaced 1 meter apart and filled alternately with 200g of alfalfa hay, 200g

of sudangrass hay, or 50g of cottonseed cake. Cows are allowed into the arena individually and in random order to feed from the bowls for four minutes, limited by the UAVs' battery life of 5-6 minutes per flight. After each session, the bowls are weighed to measure the amount of feed consumed. The UAV footage is later analyzed to determine the frequency of visits to each bowl relative to the total test duration. The results indicate a strong linear correlation between the feeding frequency observed by the UAV and the measured amount of food consumed. The study further shows that UAV monitoring does not affect the cows' natural feeding behaviors, suggesting that this method is an effective and non-invasive approach to monitor cows' feeding patterns.

As mentioned earlier, Gallo et al. [11] study the use of the object detection algorithm YOLOv7 to detect weeds on high-resolution aerial imagery captured by UAVs. The study introduces the novel Chicory Plant dataset, consisting of 3,373 RGB images with an average of 3,561 annotated weeds per image, captured at a flight height of 65 meters above ground level. All images are recorded on a 5-hectare field in Belgium cultivated with chicory. This novel dataset is used in conjunction with the existing Lincoln Beet dataset, which contains 4,405 aerial images of sugar beet and weeds captured 50 centimeters above ground. On the Chicory Plant dataset, the object detector YOLOv7 achieved a recall of 58.1% and an $mAP^{0.5}$ score of 56.6%. The mean average precision (mAP) is a widely used metric for object detection accuracy, ranging from zero to one, with one being perfect. A formal definition is provided in Section 4.1.3. The results achieved by Gallo et al. are promising and demonstrate the feasibility of using object detection algorithms to achieve a satisfactory level of weed detection accuracy on high-resolution aerial imagery. A model trained on the Lincoln Beet dataset fails to accurately predict weeds in the Chicory Plant dataset, yielding very poor results. This failure can be attributed to the significant differences between the two datasets, suggesting that task-specific datasets are needed for satisfactory performance.

Wang et al. [15] focus on the challenges of manual sheep counting and introduce a novel architecture designed to count sheep at a passage. Based on the SSD architecture, they present the Sheep's Head-Single Shot MultiBox Detector (SH-SSD), which, combined with a tracking algorithm, allows for the accurate computation of sheep quantity statistics. The dataset used consists of 5,735 images featuring various breeds of sheep, sourced from internet repositories as well as 11,624 video frames captured with a fixed camera at a 2 meter wide passage in a pasture in Mongolia. Compared to the standard SSD network, the SH-SSD network demonstrates significantly improved performance, achieving 96.11% in $mAP^{0.5}$ and 63.41% in

$mAP^{0.5:0.95}$, yielding gains of seven and twelve percentage points respectively. In addition, SH-SSD is feasible for real-time use, achieving 84 frames per second on a computing setup featuring one 48GB NVIDIA RTX A6000 graphics card.

Nowadays, livestock counting is usually done manually. This procedure is labor-intensive and errors, such as duplication and omission are prone to occur. Consequently, most farmers currently count their animals only a few times per year. The integration of object detection systems with UAVs offers a promising solution to address the problems associated with manual herd counting, enabling a faster, more frequent, less labor-intensive, and more accurate assessment of animal count. This would make an earlier detection of livestock losses possible, which, in turn, enables a faster implementation of countermeasures to prevent further losses caused by predators, damaged fences, or theft. Research in this area is ongoing, encompassing various object detection systems and animal species.

A notable example is the work conducted by de Lima Weber et al. [16]. They present a novel dataset containing 878 images of Nelore cattle. All images are captured in a Brazilian feedlot by a UAV flying at altitudes between 20 and 100 meters above ground level, with the camera pointing straight downward. The object detection networks utilized are four models of YOLOv5, with various weight sizes. Each model is trained for 2,000 epochs using the Adam optimizer with standard settings. For the selected performance metrics, including precision, recall, and $mAP^{0.5}$, all tested models performed similarly. However, for all metrics, the largest model, YOLOv5-x, achieves the best results with a precision of 0.939, a recall of 0.981, and an $mAP^{0.5}$ of 0.974. These promising results show that UAVs and AI can enable farmers to obtain frequent herd counts with minimal effort.

The following studies on counting sheep in UAV aerial imagery using object detectors are reviewed with special attention on the datasets as shown in Table 2.1.

	Xu et al. [17]	Sarwar et al. [18]	Doll et al. [13]
Number of images	1,000	–	1,727
Resolution	4096×2160	4096×2160	3840x2160
Flight height	–	80 m & 120 m	various
Animals	Sheep & Cattle	White Sheep	White Sheep
Ground	Grassland	Grassland	Grassland
Location	Australia	New Zealand	–
Weather	Clouds & Sun	Clouds & Sun	Sun
Availability	Not found	Not found	Public Domain

Table 2.1: Properties of selected datasets containing sheep captured by UAVs

Xu et al. [17] focus on achieving high accuracy in both classifying as well as counting livestock in aerial imagery captured by UAVs. They introduce a novel dataset comprising 1,000 high-resolution images, equally split between cattle and sheep in Australian pastures. The images are annotated with polygons that accurately outline the animals' contours, providing detailed shape information. The images are resized to 512×512 pixels for processing, and no data augmentation is performed. The researchers employ the Mask R-CNN algorithm, which predicts the animals' contours, enabling further analysis of their shape, pose, and direction, which is useful for monitoring abnormal behaviors. A pretrained Mask R-CNN model is fine-tuned on the dataset using stochastic gradient descent (SGD) for 1,000 epochs. Afterwards, it achieves a total CA of 96% and a classification accuracy of 92%.

Sarwar et al. [18] explore UAV-based sheep counting in a series of research articles. Their recent study builds upon earlier work and introduces a new dataset composed of aerial images captured at altitudes of 80 m and 120 m. These high-resolution images depict white sheep on green grassland in New Zealand, captured under various weather conditions. For processing efficiency, the images are divided into overlapping sub-images and resized to 256×256 pixels. Augmentation techniques, such as translation and rotation, are applied to increase the number of training images. They train a fully connected network using centroids as ground truth data as well as a single-layered and a seven-layered CNN network for performance evaluation and comparison. Additionally, multiple existing networks are fine-tuned on the dataset. To enhance accuracy, two of the networks are combined, achieving 99% precision and 98% recall. According to the authors, these outcomes surpass all previously reported results for detecting livestock in aerial imagery.

Doll and Loos [13] compare 44 state-of-the-art object detectors on a publicly available dataset provided by RIIS. The dataset consists of 1,727 high-resolution aerial images of white sheep on grassland, primarily captured in direct sunlight. They trained and evaluated YOLOv5 to YOLOv8, SSD, EfficientDet, CenterNet, and Faster R-CNN in various network sizes and using the default settings recommended by the models' authors. All networks use pretrained weights from the COCO dataset. Their findings indicate that for six out of the eight detection architectures tested, the largest model does not achieve the highest $mAP^{0.5}$. This is, according to the authors, likely due to the larger models suffering from underfitting[1] because of the small dataset. The best performances are achieved by YOLOv5-lu and SSD RN50 v1 FPN, with $mAP^{0.5}$ values of 0.955 and 0.959, respectively.

[1]This might be a confusion as the problem is more likely to be overfitting.

2.2 Object Detection Using CNNs

Object detection aims to locate and classify objects in an image and label them using rectangular bounding boxes. The majority of object detection algorithms used today rely on CNNs, which are a specialized variant of ANNs that are optimized for handling image data.

Object detection with CNNs can be performed using two main methods: two-stage detectors, which use a multi-step process, and one-stage detectors, which unify the entire process in a single network.

This section starts with a brief review of ANNs. After that, an introduction to CNNs is provided and the two main approaches to detecting objects with CNNs are explained. The first two parts follow the structure and idea of O'Shea and Nash [8].

2.2.1 Introduction to ANNs

Artificial Neural Networks draw inspiration from biological nervous systems, particularly animal brains. These networks consist of numerous interconnected computational units, known as neurons, which collaboratively learn from the system's input to optimize the final output. Typically, ANNs are structured in layers, where each layer consists of multiple neurons that receive all outputs from the previous layers' neurons as their input, perform a mathematical operation, and pass the result to the subsequent layer. The first and last layers of the network are exceptions to that. In the first layer, the input layer, each neuron receives raw data as its input, while the last layer, called the output layer, provides the network's final result. This type of architecture is often referred to as a feed-forward network. See Figure 2.2 for an illustration of a typical ANN structure. Note that, in practice, there are usually more layers with more neurons than depicted here.

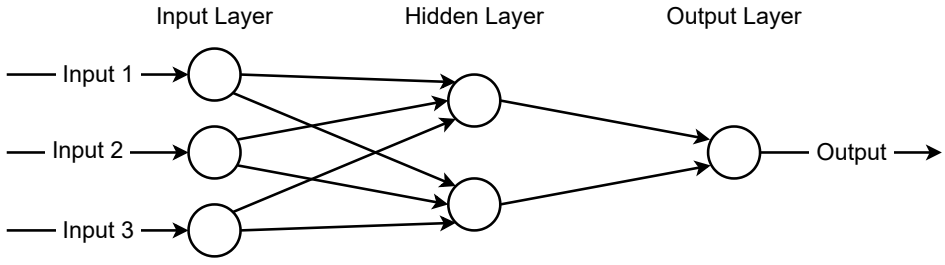


Figure 2.2: The basic structure of an ANN

There are two key learning paradigms in AI: supervised and unsupervised learning. Here, we only focus on supervised learning, where the model learns from pre-labeled

inputs. Each training example consists of a set of input values paired with the correct output values (annotations). The goal of the model is to learn a mapping from inputs to outputs. During training, the model makes predictions on the training data and corrects itself by comparing its predictions with the annotations.

Every neuron has a weight parameter associated with each of its input connections. These numerical values determine the importance of all input features in producing the neuron's output. During training, these weights are adjusted to optimize the model's performance enabling the network to learn.

Predicting a discrete value, such as an assignment to a finite set of classes, is called classification, while predicting a continuous number is known as regression.

2.2.2 Basic Principles of CNNs

One major limitation of traditional ANNs is their struggle with the computational complexity involved in processing image data. For instance, an RGB input image with 600×600 pixels would require each neuron in the first hidden layer to have 1,080,000 connections, one for every base-color and every pixel. To manage such a large input, the network would need to be very large with numerous extensive hidden layers. This is not feasible due to limitations in computational power and the risk of overfitting. Overfitting happens when the network memorizes too specific details of the training data to the extent that it performs poorly on new, unseen data. Reducing the number of parameters required for training decreases the likelihood of overfitting.

The primary distinction between traditional ANNs and CNNs is that the latter are specialized to work with images as input data. This allows the incorporation of image-specific features into the architecture, enhancing the network's suitability for image-focused tasks while also reducing the complexity of the model, thus reducing the risk of overfitting

A key distinction is that neurons in CNN layers are organized in three dimensions: height, width, and depth. Height and width correspond to the spatial dimensions of the input (i.e., an image's dimensions), while depth refers to the number of channels, such as base colors or feature maps. For the example provided earlier, the input volume has a dimensionality of 600×600×3. Unlike in standard ANNs, where each neuron in a layer is typically connected to all neurons in the previous layer, neurons in a CNN layer are only connected to a small, localized region of the preceding layer. This localized connection is intended to capture local features, such as edges or textures in the input data.

Typical Architecture

A CNN is made up of three types of layers: convolutional layers, pooling layers, and fully-connected layers. A combination of these layers forms the architecture of a CNN. Figure 2.3 illustrates a simplified CNN architecture.

- **Input layer**: Similar to other ANNs, this layer provides the starting point of the network and holds the pixel values of the image.
- **Convolutional layer**: Here, the input image gets scanned in small sections to detect local patterns, including edges or textures using the convolution operation. This process results in a feature map that highlights where a specific pattern occurs.
- **Pooling layer**: The objective of this layer is to reduce the spatial dimensionality (height and width) of the input by downsampling in order to reduce the number of parameters while retaining essential features.
- **Fully connected layer**: These layers use the feature maps extracted by previous layers and perform classification or regression to compute the final output of the network. In order to improve performance, nonlinear activation functions may be applied between these layers.

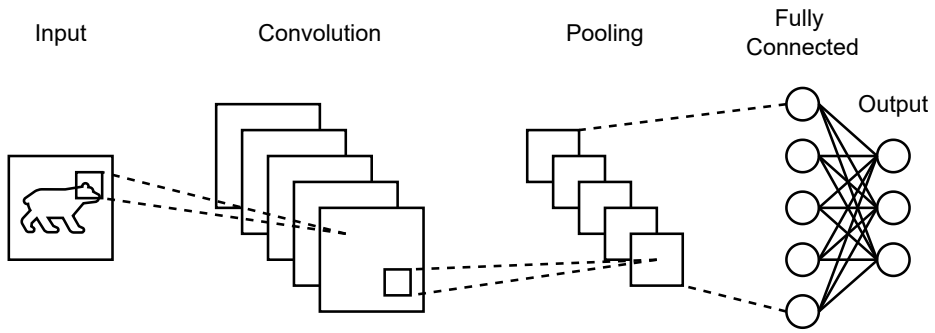


Figure 2.3: A simple CNN architecture

The remainder of this subsection provides more details on these individual layers and their hyperparameters.

Convolutional Layers

Since the CNN is named after the convolutional layer, it is evident that this layer plays a vital role in the network's operation. Convolutional layers use matrices, called filters or kernels, that are typically small in the spatial dimensions but extend across the entire depth of the input. These filters are convolved with the input data

across the spatial dimensions, resulting in a 2D feature map. This means that a filter slides over the input data with a predefined stride length. At each position, the filter performs an element-wise multiplication with the overlapping part of the input data, followed by a summation of these products. See Figure 2.4 for an example.

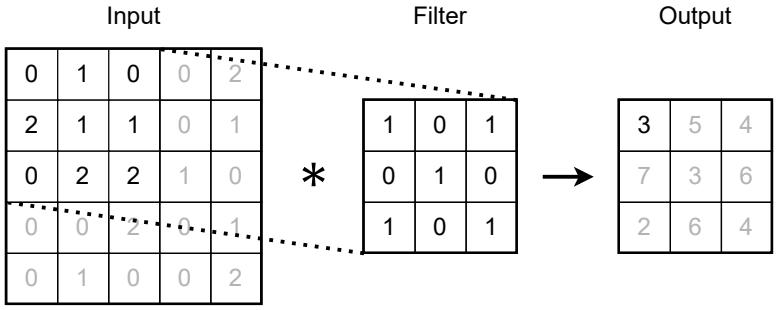


Figure 2.4: Visual example of a convolution with a stride length of one

The stride length determines the number of pixels by which the filter moves across the input image during the convolution operation. A stride length of one means that the filter moves one pixel at a time, resulting in a large output feature map. On the other hand, using a stride length of two would halve the dimensions of the output compared to a stride of one.

The borders around the input are often padded with zeros or with the values at the edge of the input. This helps to control the resulting feature map's dimensions and to preserve information at the borders.

A convolutional layer can have multiple filters, and every filter produces its own feature map, highlighting the occurrence of certain patterns, such as lines or circles. All feature maps are stacked in the depth dimension, resulting in the layer's output volume.

Usually, a non-linear activation function, such as ReLU, is applied to the output volume of the convolution layers in order to allow the network to learn more complex patterns and thus improve performance. ReLU is defined as:

$$ReLU(x) = \max(0, x) \tag{2.1}$$

As discussed earlier, to process an RGB image with a size of 600×600 pixels, each neuron in an ANN's first hidden layer would need to have 1,080,000 connections. A CNN reduces this drastically by only connecting each neuron to a small part of the input image. For example, when using a filter of size 5×5×3, each neuron in the convolutional layer only needs 75 connections.

Pooling Layers

The purpose of pooling layers is to reduce the spatial dimensionality of the input, which in turn lowers the number of parameters and the computational complexity of the model. In addition, pooling layers also help the model to become invariant to small translations of the input. They work similarly to convolution layers in that a pooling filter slides over the input and performs an operation.

There are two main types of pooling operations:

- **Max pooling**: Selects the maximum value from each patch of the input covered by the filter. This helps in retaining the most significant features detected in the previous layers.
- **Average pooling**: Computes the average of each patch of the input covered by the filter. This helps in reducing noise and preserving the overall spatial structure of the input.

Most of the time, the pooling filters in a CNN have a dimensionality of 2×2×1 and are applied with a stride length of 2, reducing the spatial dimensionality of their input to 25% while preserving the depth dimension. Typically, one or two convolutional layers alternate with a pooling layer in a repeated pattern, progressively extracting and condensing features from the input data.

Fully Connected Layers

In fully connected layers, each neuron is connected to every neuron in the adjacent layers, similar to traditional ANNs (see Figure 2.2). Typically positioned toward the end of a CNN, these layers use the feature maps extracted by the convolutional and pooling layers to make final predictions. The feature maps, which are generally multidimensional tensors, are flattened into a one-dimensional vector that serves as the input for the first fully connected layer. The output y_i for neuron i with the input vector $x = [x_1, x_2, \ldots, x_6]$ is computed as follows, where f is an activation function, such as ReLU. Note that w_{ij} represents the weight of the j-th input to the i-th neuron and b_i is the bias term, which allows shifting the activation function.

$$y_i = f(w_{i1}x_1 + w_{i2}x_2 + \ldots + w_{i6}x_6 + b_i) \tag{2.2}$$

Training

Before training can start, the architecture needs to be defined. This includes specifying the number and order of layers, the number and size of filters, as well as the

pooling and activation functions. Following this, the network's weights are initialized with small, random values. The training images are then passed through the network, and a loss function is used to compare the predicted output to the annotations. Finally, the network's weights are improved via backpropagation. This involves calculating the gradient of the loss function with respect to each of the network's weights and updating these weights using an optimization algorithm, such as SGD. This process of passing images through the network and updating the weights is repeated for many iterations to improve the network's performance.

2.2.3 Two-Stage Detectors

Two-stage detectors rely on a two-step process for locating and classifying objects. In the first stage, an algorithm generates a set of regions that may contain an object of any kind, known as region proposals. The goal is to narrow down the areas of interest and thus reduce the number of regions that need to be analyzed in detail by the second stage, which classifies the objects and refines the bounding boxes for each region proposal individually. This approach generally achieves high accuracy but is rather slow.

Stage One

The input image is first processed by the backbone network, a CNN, which extracts a set of feature maps. The feature maps generated by the first layers of the backbone network capture fine details, whereas those produced by the later layers capture higher-level features. The feature maps are subsequently used to compute region proposals.

A grid is laid over the feature maps, and at each grid cell, multiple predefined anchor boxes with various scales and aspect ratios are placed. This ensures that the network can detect objects of different sizes and aspect ratios.

For each anchor box, a small CNN called the region proposal network (RPN) predicts the objectness score, which indicates the likelihood that the box contains an object. Anchor boxes that surpass a predefined threshold are considered region proposals. The RPN also refines each anchor box to fit the potential object more precisely.

If multiple region proposals overlap beyond a certain threshold, only the proposal with the highest objectness score is retained. This process, known as non-maximum suppression (NMS), prevents multiple region proposals for one object.

Stage Two

First, the feature maps, which are cropped to the region proposals generated by stage one, are rescaled to a uniform size. These feature maps are then processed by fully connected layers, referred to as the network's head, which refine the extracted features in preparation for classification and bounding box adjustments. Finally, a fully connected layer outputs class scores for each region proposal, indicating their likelihood of belonging to specific categories of objects. Simultaneously, the network predicts offset values for the bounding boxes to fit the detected objects precisely. The final output of the object detector consists of multiple bounding boxes, usually described by their width, height, and the location of the top left corner, as well as the class label with the highest classification score along with its score.

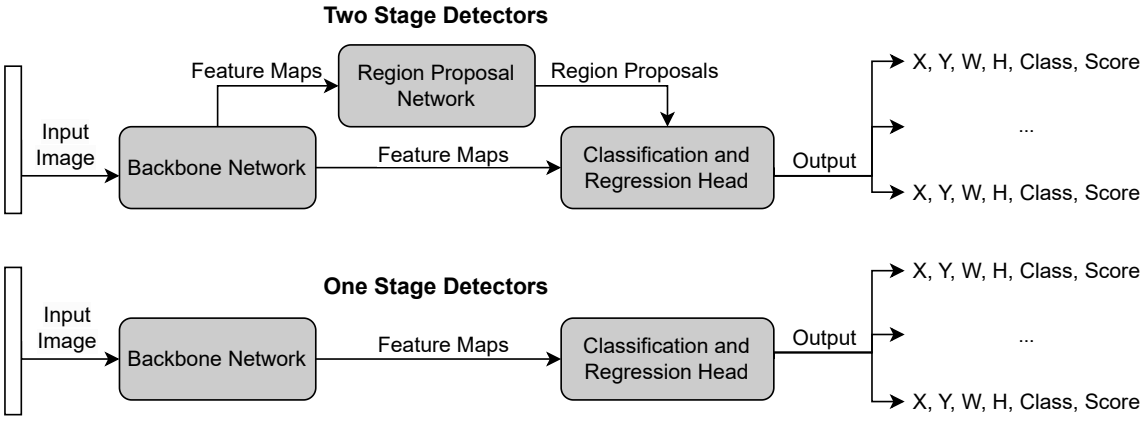


Figure 2.5: A visualization of the two main object detection approaches using CNNs

2.2.4 One-Stage Detectors

One-stage detectors unify the entire object detection process in a single network and skip the region proposal generation. This usually sacrifices some accuracy but increases the inference speed – the time it takes to generate new predictions – compared to two-stage detectors.

Similar to two-stage detectors, a backbone CNN first extracts feature maps from the input image. After that, the detection network applies additional convolutional layers to the feature maps to prepare them for the subsequent layers. Again, anchor boxes are used as reference points to predict the actual bounding boxes. Depending on the specific architecture, convolutional or fully connected layers are used to predict the class scores as well as offsets for all bounding boxes. Finally, NMS is applied to remove redundant boxes.

See Figure 2.5 for a visual comparison between the two approaches.

2.3 Comparison of Selected Algorithms

Having established the fundamental principles of object detection algorithms, this section focuses on a detailed examination and comparison of three specific algorithms. These algorithms — Faster R-CNN, RetinaNet, and YOLO — are subsequently trained and evaluated using aerial sheep datasets. Faster R-CNN is designed as a two-stage detector, whereas RetinaNet and YOLO are one-stage detectors. Table 2.2 provides a quick reference for comparing the main attributes of each algorithm.

	Faster R-CNN [19]	Retina Net [20]	YOLOv9 [21]
Architecture	two-stage	one-stage	one-stage
Backbone Network	ResNet or VGG	ResNet	Darknet
Anchor Boxes	yes in RPN	yes	no, direct regression
Loss Function	–	–	–
Speed	slow	moderate	fast
$mAP^{0.5:0.95}$ on COCO	0.467 (0.215)	0.436 (0.391)	0.556
Year of publication	2015	2017	2024

Table 2.2: Properties of selected object detectors

2.3.1 Faster R-CNN

Introduced by Ren et al. [19] in 2015, the two-stage detector Faster R-CNN builds on its predecessors, R-CNN and Fast R-CNN. The original R-CNN model, proposed by Girshick et al. [22] in 2014, is the pioneering algorithm to demonstrate the effectiveness of CNNs in object detection tasks. Gershick refines this method in 2015 with Fast R-CNN [23], which features a Region of Interest (RoI) pooling layer that improves speed and accuracy. Building on this progression, Faster R-CNN integrates an RPN into the architecture, allowing end-to-end training and further performance enhancements.

Faster R-CNN is renowned for its accuracy, particularly in detecting small objects. However, it requires substantial computational resources, and its inference speed is relatively slow.

On Microsoft COCO, a popular benchmarking dataset for object detectors, Faster R-CNN achieves an $mAP^{0.5:0.95}$ of 21.5%. This dataset includes 80k training, 40k validation, and 20k test images featuring objects from 80 different categories [24].

As its backbone network, Faster R-CNN uses VGG or ResNet. These networks are also used in other computer vision tasks and are not specific to object detection. The feature maps output by the backbone network serve as the input to the RPN, which generates a set of region proposals using a 3×3 convolutional filter and anchor boxes as explained in Section 2.2.3. Each region proposal is mapped onto the corresponding location in the feature maps. Because the proposals can vary in size, in the RoI pooling layer, a max-pooling operation is used to standardize them. After that, the fixed-size feature maps are flattened into a single vector and passed through two fully connected layers. The output of the second layer is then branched into two separate heads: one fully connected layer that outputs a probability distribution over the object classes, and another layer that outputs four values representing the bounding box coordinates (x, y, width, and height) for each proposal.

For the evaluation on the aerial sheep datasets, the fasterrcnn_resnet50_fpn_v2 implementation of Faster R-CNN provided by Torchvision is utilized. This model incorporates enhancements suggested by Li et al. [25], including the use of an improved backbone network, two convolutional layers in the RPN, and employing four convolutional layers followed by a fully connected layer to predict bounding boxes. This improved model achieves an $mAP^{0.5:0.95}$ of 46.7% on the COCO dataset.

2.3.2 RetinaNet

In 2017, Lin et al. [20] introduce the one-stage detector RetinaNet. Upon its introduction, this algorithm achieves the speed of earlier one-stage detectors and exceeds the accuracy of all state-of-the-art two-stage detectors. On COCO, this model achieves an $mAP^{0.5:0.95}$ of 40.8%.

RetinaNet uses ResNet as its backbone network, which outputs various feature maps at different scales. These feature maps are processed by the Feature Pyramid Network (FPN) to create a multi-scale feature hierarchy that enables the detection of objects of different sizes. The FPN first reduces the depth of each feature map using 1×1 convolutional layers. Then, it upsamples the lower-resolution feature maps and combines them with the next higher-resolution maps through element-wise addition. Anchor boxes are used as starting points for predicting the bounding boxes. The head of the network is split into two subnetworks: one for classification and one for regressing the box offsets. Each subnetwork comprises five 3×3 convolutional layers.

RetinaNet introduces a novel loss function that improves handling of the class imbalance between background and actual objects. Focal loss modifies the traditional

cross-entropy loss by down-weighting easy examples and, in turn, focusing on hard negatives. This loss function is defined in Equation 2.3, where p_t represents the predicted probability for the true class and γ is a predefined focusing parameter.

$$FL(p_t) = -(1 - p_t)^{\gamma} \log(p_t) \tag{2.3}$$

In the evaluation, the retinanet_resnet50_fpn_v2 implementation from torchvision is employed. This model incorporates the improvements proposed by Zhang et al. [26], which include a novel way of defining positives and negatives, i.e., objects and background. With these improvements, the model achieves an $mAP^{0.5:0.95}$ of 43.6% on COCO.

2.3.3 YOLOv9

The initial version of YOLO, introduced in 2016 by Redmon et al. [27], marks the debut of one-stage detectors. It offers substantially faster performance compared to all common object detectors available at that time, but this comes at the cost of accuracy. Subsequently, numerous advanced versions are developed, culminating in the release of YOLOv9 by Wang et al. [21] in 2024. YOLOv9 is available in four differently sized versions, with the largest one achieving an $mAP^{0.5:0.95}$ of 55.6% on COCO.

YOLOv9 incorporates numerous complex features, the full explanation of which would exceed the scope of this thesis. Nevertheless, intuitive explanations are offered for several key features:

It does not rely on anchor boxes and instead directly estimates an object's center, thereby simplifying the model. The Programmable Gradient Information (PGI) feature mitigates data loss during training, resulting in enhanced efficiency and improved performance. Additionally, the Generalized Efficient Layer Aggregation Network (GELAN) allows for the flexible selection of computational blocks tailored to specific devices for improving inference speed without sacrificing accuracy. Furthermore, YOLOv9 uses the CSPDarknet53 backbone network, which provides a strong and efficient foundation for feature extraction.

For the evaluation phase, this study utilizes the YOLOv9 implementation provided by Ultralytics. Due to computational limitations and the findings reported by Doll et al. [13], only the second-largest model variant, YOLOv9c, is employed.

Chapter 3

Novel Datasets

This chapter provides an overview of the datasets used for evaluation. The first section (3.1) reviews the existing dataset published by RIIS, which comprises images of white sheep on grassland. The following sections (3.2 - 3.4) highlight the development of novel datasets that feature sheep in more adverse conditions in Switzerland, such as on snow, during dusk, and in rough terrain. See Table 3.1 for a quick comparison between the four datasets.

	Grassland	Snow	Dusk	Rough Terrain
Images	1,727	200	200	200
Background	grassland	snow	grass & snow	grass & dirt hills
Sheep color	white	various	white & black	black
Lighting	sun	clouds	dusk	sun & clouds
Resolution	3840×2160	8064×4536	8064×4536	8064×4536
Annotations	55,435	3,444	4,604	4,775

Table 3.1: Characteristics of utilized datasets

3.1 Grassland

As previously mentioned, several studies examine the use of object detectors for counting white sheep on grassland. Consequently, publicly available datasets featuring white sheep in such environments exist. For further evaluation in this thesis, the Aerial Sheep dataset provided by RIIS on Roboflow [28] is employed. Refer to Figure 3.1 for illustrative examples.

The dataset comprises a total of 1,727 aerial images captured by a UAV, featuring 55,435 annotated sheep. All images are taken under direct sunlight conditions.

While the majority of images depict white sheep on plain grassland, some also include other elements, such as tree shadows, buildings, fences, and cows. The annotations lack precision, with many bounding boxes being considerably larger than the sheep they mark.

A random selection of 70% of all images is used for training, 20% is used for validating the hyperparameters, and 10% is reserved for final testing. This split is predefined by RIIS. The training images are augmented to generate three training examples per original image using horizontal and vertical flipping, 0% to 20% cropping, adjusting the hue from -15° to +15°, saturation and brightness adjustments of -15% to +15%, and exposure compensation of -10% to +10%.

Figure 3.1: Selected images of the Aerial Sheep dataset by RIIS [28]

3.2 Snow

The first novel dataset of sheep in adverse conditions features white, brown, and black sheep on snow-covered ground. Figure 3.2 displays selected images from the dataset.

All images are captured with a DJI Mavic Air 3 [29]. This commercially available UAV is equipped with two cameras. The wide-angle camera offers a field of view of 82° and an aperture of f/1.7, while the telephoto camera has a 35° field of view and an aperture of f/2.8. Both cameras use sensors with a resolution of 8064×6048 pixels and are stabilized by a motorized gimbal. This dual-camera setup allows for capturing more diverse images from various perspectives.

The images in this dataset are captured in December 2023 and January 2024 across seven different locations in Switzerland: Utzenstorf, Kernenried, Wiler bei Utzenstorf, Aefligen, two locations in Lohnstorf, and Gerzensee. Both the wide-angle and telephoto cameras are employed, with flight heights ranging from 20 to 50 meters. At heights below 20 meters, some sheep show signs of disturbance, while above 50 meters, the sheep are not visible enough for reliable annotation. The weather conditions during capture include snowfall and overcast skies. All 937 images are taken during daylight hours.

A random selection of 200 images is annotated using bounding boxes on Roboflow, resulting in 3,444 annotated sheep. Animals at the edge of images are annotated only if they are clearly identifiable as such. The dataset is divided into 70% training, 15% validation, and 15% test data, and the same augmentation settings as used on the grassland dataset by RIIS are applied. The annotated images are publicly available on Roboflow as the SnowSheepUAV dataset [30], while the remaining images can be provided upon request.

This dataset presents challenges due to the similarity in appearance between the sheep and their surroundings. White sheep blend into the white snow, while dark sheep are difficult to distinguish from sleep spots, where the snow is melted in sheep-sized patches, exposing the dark ground beneath. Moreover, differentiating individual sheep is challenging because they are often clustered closely together in groups.

Figure 3.2: Selected images of the SnowSheepUAV dataset

3.3 Dusk

The second novel dataset comprises aerial imagery of sheep during dusk, both on snow-covered ground and on grassland. Figure 3.3 depicts selected images.

In December 2023, 386 images of black and white sheep on snow are captured in Kernenried and Aefligen at flight heights between 20 and 50 meters. In May 2024, 543 images featuring white sheep on grassland are captured in Utzenstorf and Wiler at flight heights between 20 and 100 meters. All images are taken between 30 minutes before and one hour after sunset. Beyond this time frame, the sheep are not visible enough for accurate annotation.

A random selection of 100 snow images and 100 grassland images are annotated with a total of 4,604 bounding boxes, and the same annotation rules, split settings, and augmentations as used in the snow dataset are applied. The dataset is available for further research on Roboflow as DuskSheepUAV dataset [31]. The poor lighting

conditions, which minimize the contrast between the sheep and their surroundings, present challenges.

Figure 3.3: Selected images of the DuskSheepUAV dataset

3.4 Rough Terrain

The third and final novel dataset contains pictures of black sheep in rough terrain, such as on steep hills with trees, tall grass, dirt, stones, and streams. See Figure 3.4 for illustrative examples.

All images are captured in May 2024 at three different locations around Wattenwil, Switzerland. Both cameras of the DJI Mavic Air 3 are utilized, and due to the significant contrast between the sheep and the ground, flight heights of up to 80 meters are possible. The images are taken during daylight hours with direct sunlight or overcast skies.

Out of the 286 captured images, a random selection of 200 images is annotated with a total of 4,775 sheep annotations. The annotation rules, split settings, and augmentations used are identical to those applied in the snow dataset. This dataset is publicly available on Roboflow as RoughTerrainSheepUAV dataset [32].

The challenges in this dataset are mainly due to partial occlusions of the animals by trees and tall grass. Furthermore, many adult sheep are accompanied by their lambs in close proximity, complicating differentiation. Because of the steep terrain and the presence of both adult sheep and lambs, sheep within the same image appear in a wide range of sizes.

Figure 3.4: Selected images of the RoughTerrainSheepUAV dataset

Chapter 4

Empirical Evaluation

This chapter describes the conducted experiment. Section 4.1 outlines the experimental setup, including the performance metrics and hyperparameter tuning strategy. After that, Section 4.2 presents and discusses the achieved results.

4.1 Experimental Setup

This experiment aims to evaluate the accuracy of widely-used object detection algorithms like Faster R-CNN, RetinaNet, and YOLOv9 in counting sheep on aerial imagery under adverse conditions. It also seeks to assess if this accuracy meets the requirements for real-world applications. Additionally, the experiment aims to determine the necessity and impact of the novel datasets.

Training & Validation	Testing
Grassland	Grassland
Grassland	Snow
Grassland	Dusk
Grassland	Rough Terrain
Snow	Snow
Dusk	Dusk
Rough Terrain	Rough Terrain
All	Grassland
All	Snow
All	Dusk
All	Rough Terrain

Table 4.1: Experimental setup

4.1.1 Train-Test Split

As described in Chapter 3, all datasets are randomly split into 70% training, 15% validation, and 15% test data, except for the grassland dataset, which follows a 70-20-10 split. The assignment of images to these splits remains consistent across all configurations. To improve speed and reduce model complexity, the images are rescaled to 600×600 pixels.

All algorithms use pretrained weights from COCO [24]. They are then trained individually on each of the four datasets to create specialized models, as well as on a combined dataset to create a generalized model.

The best hyperparameters are determined based on their performance on the validation split. Subsequently, the optimal configuration for each algorithm and training dataset is evaluated on the test split of various datasets. This evaluation also examines the detectors under unfamiliar conditions, thereby assessing the importance of the novel datasets. Table 4.1 provides an overview of the different combinations of training, validation, and test data examined for each detector.

4.1.2 Hyperparameter Tuning

The hyperparameters for all three algorithms are tuned individually for each training/validation dataset. For Faster R-CNN and RetinaNet, which are implemented using Torchvision, a gridsearch is conducted to tune the learning rate, momentum, and weight decay of the SGD optimization function. Refer to Table 4.2 for the tested values. Both algorithms are trained for 50 epochs with every hyperparameter combination. The hyperparameters and epoch which achieved the best CA on a given validation dataset are then selected as the final model.

Parameter	Practical Range	Tested values
Learning rate	[0, 1]	0.0005, 0.001, 0.005, 0.01, 0.05
Momentum	[0, 1]	0.8, 0.9, 0.95, 0.97, 0.99
Weight Decay	[0, 0.1]	0.0001, 0.0005, 0.001, 0.005, 0.01

Table 4.2: Gridsearch values for Faster R-CNN and RetinaNet

The tuning procedure proposed by Smith [33], which uses a learning rate range test, does not yield sufficient results in the current task. Nonetheless, following Smith's recommendation, the largest feasible batch size is employed. On a single Nvidia RTX 3090 from UBELIX, the high-performance computing cluster at the University of Bern [34], the maximum achievable batch size is 16 images per batch.

The YOLOv9 implementation by Ultralytics uses a genetic algorithm to optimize its hyperparameters. This class of algorithms takes inspiration from genetics and natural selection and applies small, random modifications to the hyperparameters to generate new candidates for evaluation [35]. On every training/validation dataset, YOLOv9 is tuned for 300 iterations of 30 epochs. Despite YOLOv9 applying its own augmentations, the additional augmentations outlined in Chapter 3 enhance performance, as evidenced by preliminary experiments.

4.1.3 Performance Metrics

Two performance metrics are used in this thesis: counting accuracy (CA) and mean average precision (mAP). While CA is one of the most important metrics for the task at hand, mAP is widely used in object detection and is therefore also reported.

Counting Accuracy (CA)

This thesis primarily focuses on the accuracy of sheep counting. Bounding box precision and other metrics are of secondary importance, as the system is designed to count sheep rather than evaluate their position. Therefore, the CA, defined in Equation 4.1, best answers the main question of this thesis in a single numerical value. For multiple images, the mean CA of all images is reported.

$$CA = 1 - \frac{|detected\ count - ground\ truth\ count|}{ground\ truth\ count} \tag{4.1}$$

Mean Average Precision (mAP)

The mAP is one of the most widely used metrics for evaluating the performance of object detectors. It is the mean of the average precision across all classes. The average precision represents the area under the precision-recall curve, which is obtained by plotting precision on the y-axis against recall on the x-axis. This curve is smoothed by replacing $P(r)$, the precision at recall level r, with the maximum precision for any recall level $\hat{r} \geq r$, as defined in Equation 4.2.

$$P_{interp}(r) = max_{\hat{r} \geq r} P(\hat{r}) \tag{4.2}$$

The definitions of precision and recall are given by the following equations, where TP denotes true positives (correctly detected sheep), FP denotes false positives (incorrectly detected sheep), and FN denotes false negatives (unrecognized sheep).

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN} \tag{4.3}$$

A prediction is regarded as true if it has an intersection over union (IoU) with a ground truth box larger than a given threshold, which is usually denoted in superscript, e.g., $mAP^{0.5}$ for an IoU threshold of 0.5. The IoU is outlined in Equation 4.4.

$$IoU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{▣}}{\text{▣}} \tag{4.4}$$

In the special case of $mAP^{0.5:0.95}$, popularized by COCO [24], the average mAP over different IoU thresholds ranging from 0.5 to 0.95 with a step-size of 0.05 is reported.

4.2 Results and Discussion

This section presents and discusses the results achieved by Faster R-CNN, RetinaNet, and YOLOv9 on the various experimental setups outlined in Section 4.1. Additionally, a simple method for improving CA performance is presented.

4.2.1 Quantitative Results

Table 4.3 displays the CA and $mAP^{0.5}$ for the three object detection models Faster R-CNN, RetinaNet, and YOLOv9 across various training and testing configurations, as outlined in Section 4.1. The table is structured with rows indicating different combinations of training and test data, such as grassland, snow, dusk, and rough terrain. The columns are organized to show the results for each model, with separate columns for CA and $mAP^{0.5}$ under each model. The leftmost column lists the training and validation datasets, followed by the testing datasets, while the subsequent columns provide the performance metrics for the three models. The highest value per configuration and metric is highlighted in bold.

Training & Validation	Testing	Faster R-CNN		RetinaNet		YOLOv9	
		CA	$mAP^{0.5}$	CA	$mAP^{0.5}$	CA	$mAP^{0.5}$
Grassland	Grassland	**0.977**	0.942	0.970	0.912	0.975	**0.968**
Grassland	Snow	0.078	0.002	**0.188**	0.009	0.011	**0.010**
Grassland	Dusk	**0.409**	0.204	0.394	0.227	0.376	**0.331**
Grassland	Rough Terrain	0.141	**0.013**	**0.155**	0.012	0.098	0.000
Snow	Snow	0.913	0.879	0.885	0.716	**0.942**	**0.896**
Dusk	Dusk	**0.905**	0.843	0.806	0.643	0.869	**0.895**
Rough Terrain	Rough Terrain	0.921	0.890	0.895	0.778	**0.928**	**0.906**
All	Grassland	**0.977**	0.932	0.971	0.910	0.9761	**0.967**
All	Snow	**0.942**	0.859	0.790	0.646	0.933	**0.886**
All	Dusk	**0.930**	0.840	0.859	0.649	0.928	**0.875**
All	Rough Terrain	0.901	0.842	0.874	0.685	**0.921**	**0.885**

Table 4.3: Counting accuracy and $mAP^{0.5}$ of Faster R-CNN, RetinaNet, and YOLOv9 on different experimental setups

Note that the results reported here should be interpreted with caution, given that the test data originates from the same flights as the training data, resulting in a high degree of similarity. Furthermore, the snow, dusk, and rough terrain datasets each contain only 30 images in their test splits, which is relatively low.

The confidence threshold, i.e., the minimum confidence score required for a prediction to be counted as valid, has a significant impact on the number of predictions. Instead of the common value 0.5, the optimal threshold is determined for each model based on the CA on the validation dataset. Refer to Appendix A for more detailed information about the confidence thresholds and the hyperparameters used for training.

According to the farmers supplying their sheep for this thesis, a UAV system for sheep counting would be valuable if it achieves a CA of at least 0.95. They note that this is the same level of accuracy they attain with manual counting. Therefore, the algorithms are considered adequate for real-world applications if they achieve a CA of at least 0.95. This threshold is referred to as CA95.

All detectors trained on the grassland dataset achieve a CA above 0.97 when predicting grassland images, significantly surpassing the CA95 threshold and slightly outperforming the results achieved by Xu et al [17]. Faster R-CNN achieves the highest CA of 0.977, while YOLOv9 achieves the best $mAP^{0.5}$ of 0.968, moderately surpassing the results reported by Doll and Loos [13]. However, the performance reported by Sarwar et al. [18] surpasses the results achieved in this thesis. The YOLOv9 algorithm achieves a precision of 0.979 and a recall of 0.961, which are one and two percentage points lower, respectively, than the results reported by Sarwar et al. It is important to note that while the dataset used by Doll and Loos is very similar to the one used in this thesis, the datasets employed by Xu et al. and Sarwar et al. are not publicly available. Therefore, their results cannot be compared to the results obtained in this thesis with full accuracy.

The algorithms trained exclusively on grassland images exhibit poor performance when predicting sheep in the other three datasets. Comparing their performance to the detectors specialized for snow, dusk, and rough terrain reveals that the specialized detectors significantly outperform the grassland-trained detectors. This finding supports the assertion by Gallo et al. [11] that task-specific datasets are highly important. For all three adverse conditions, the best specialized detectors achieve a CA of at least 0.9, but do not meet the CA95 threshold. Thus they cannot be considered sufficient for real-world applications.

Among all setups with identical training and testing datasets, the highest performance is achieved on the grassland dataset, whereas the lowest performance is observed on the dusk dataset. As shown in Figure 3.3, the dusk images are the most challenging for human visual detection due to the low contrast between the sheep and the background. Furthermore, this dataset comprises images of both

snow-covered ground and grassland, necessitating the models to generalize across significantly different terrains.

When analyzing the performance of detectors trained on the training data from all four datasets together, the results reveal notable strengths of such generalized detectors. They are evaluated on the test data of each individual dataset separately. The Faster R-CNN model achieves the highest CA in all but one test case. These CA scores match or surpass the performance of the best specialized detectors in most cases, indicating that the generalized models outperform the specialized ones. This might be due to the larger training dataset. Regarding $mAP^{0.5}$, YOLOv9 achieves the highest score of all generalized detectors across all test cases, although it falls slightly short of the specialized detectors' performance. These results suggest that training a single detector on a mixed dataset encompassing various conditions may be sufficient. This approach would significantly simplify deployment for real-world applications compared to using specialized detectors for each condition.

In most experimental setups, Faster R-CNN and YOLOv9 outperform RetinaNet on the CA metric, particularly when trained and tested on the same dataset. Regarding the $mAP^{0.5}$ metric, YOLOv9 performs best across almost all setups. For instance, in the Grassland training and Grassland testing setup, YOLOv9 attains the highest $mAP^{0.5}$ of 0.968, surpassing both Faster R-CNN and RetinaNet. This trend is also evident in most other setups. This observation aligns with the results achieved on the COCO dataset, as shown in Table 2.2, where YOLOv9 outperforms the other two algorithms.

4.2.2 Qualitative Results

As an illustrative example, a closer examination focuses on the Faster R-CNN model trained on the dusk dataset. This analysis includes an evaluation of the gridsearch, the loss and accuracy of the best configuration during training, and a detailed look at predicted images to assess the algorithm's strengths and weaknesses.

As illustrated in Figure 4.1, there is a substantial variation in performance depending on the hyperparameters used for training, with the CA on the validation set ranging from 0.7 to 0.92. The left plot shows the CA on the z-axis, with the learning rate and weight decay varying across the entire tested range on the y- and x-axes, respectively. The momentum is fixed at 0.97, which achieves the best performance. The right plot shows the CA on the z-axis, with the learning rate on the y-axis and momentum on the x-axis. The weight decay is fixed at its optimal value of 0.0005. Note that for hyperparameter evaluation, the confidence threshold is consistently

set at 0.5, and both plots show the CA at the best epoch for each hyperparameter configuration. As indicated by the white dot, the highest CA is achieved by the Faster R-CNN model using momentum=0.97, weight decay=0.0005, and learning rate=0.01 at iteration 25. This model is selected for final evaluation on the test set, and its performance is reported in Table 4.3.

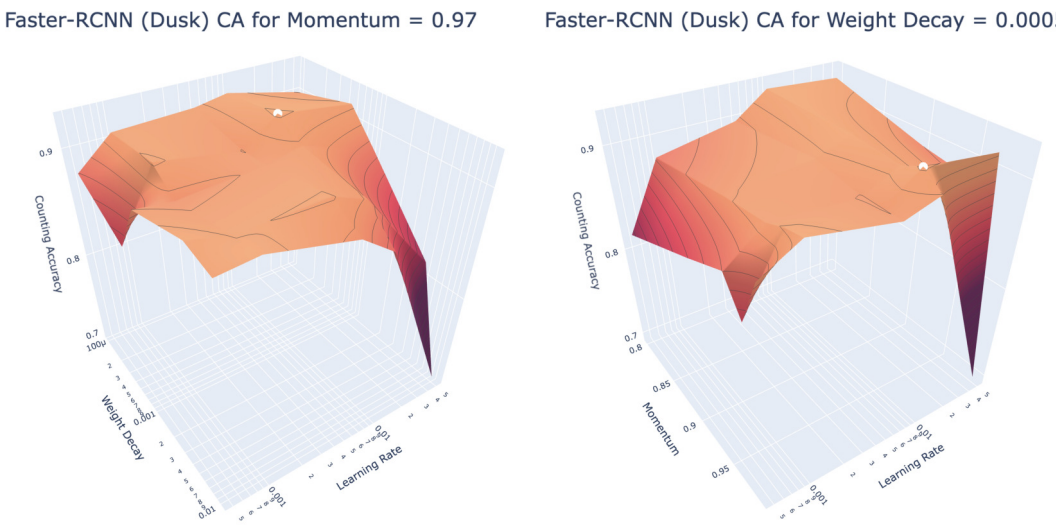


Figure 4.1: The CA for Faster R-CNN with various hyperparameter configurations trained on the dusk dataset. Since the gridsearch covers three parameters, one is fixed on the value of the optimal configuration for each plot. The coloring corresponds to CA, ranging from dark for the lowest to bright for the highest CA.

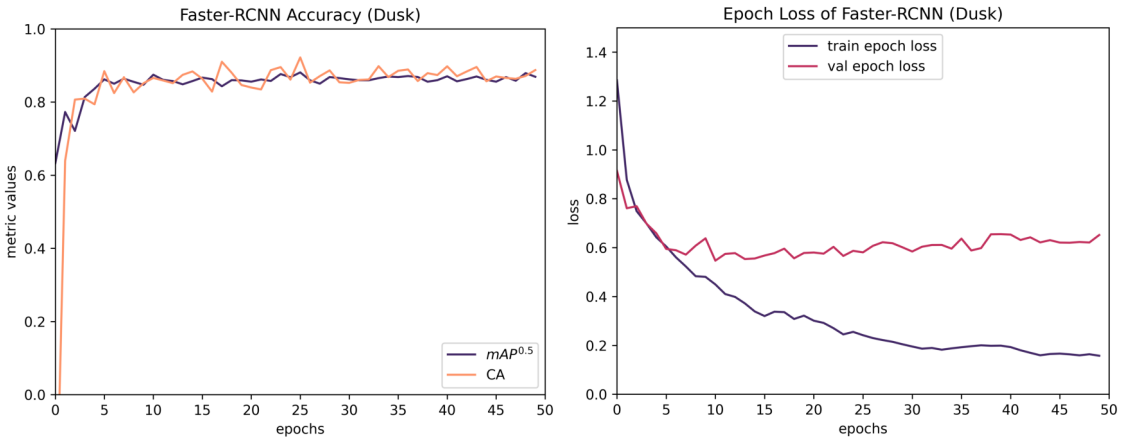


Figure 4.2: Accuracy (left) and epoch loss (right) of Faster R-CNN during training on the dusk dataset

Figure 4.2 illustrates the loss and performance of the best Faster R-CNN model during training on the dusk dataset. The left plot shows the CA (orange) and $mAP^{0.5}$ (blue) achieved on the validation set at the end of each epoch. The right plot displays the training (blue) and validation (red) loss per epoch. Both plots reveal that after ten epochs of training, performance improvements on the validation

set plateau, and the validation loss even begins to increase slightly. Notably, the training loss continues to decrease until epoch 35, suggesting that the model is likely overfitting the training data. This observation explains why the best performance for all Faster R-CNN and RetinaNet models is achieved before epoch 50.

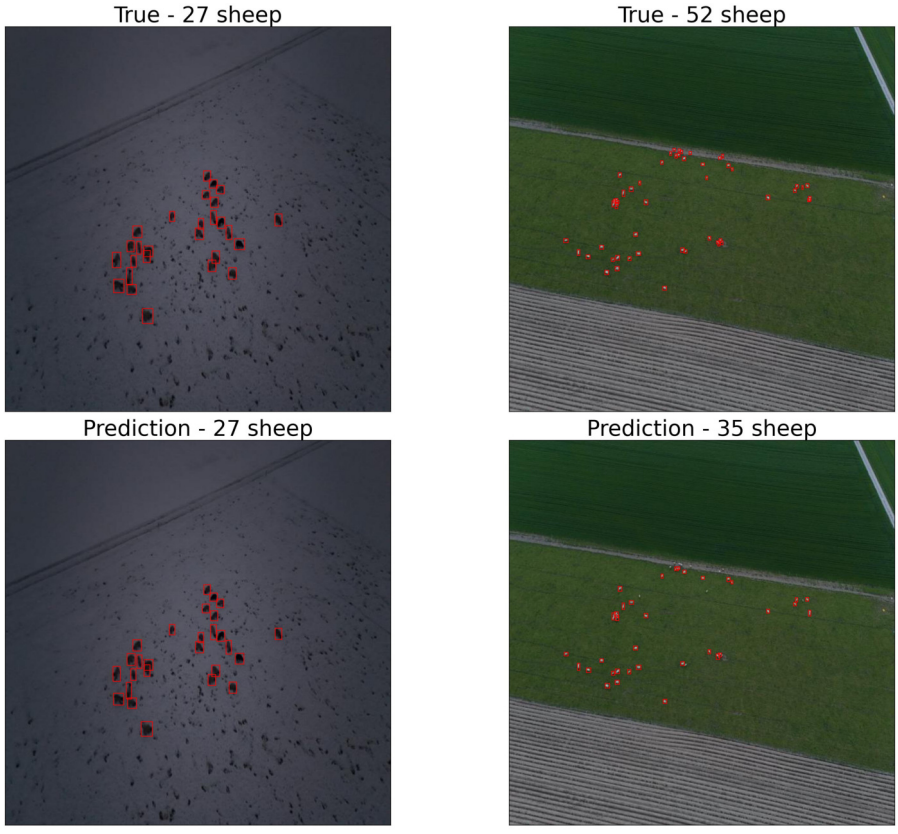
Figures 4.3 (a) – (d) provide illustrative examples of images from the dusk dataset. The top image is annotated using the true annotations, while the bottom image depicts Faster R-CNN's predicted bounding boxes.

Comparing the predicted bounding boxes by Faster R-CNN to the true annotations reveals the algorithm's strengths and weaknesses. The algorithm excels at detecting large, isolated objects, as shown in Figure 4.3 (a), but it has difficulty identifying small appearing objects as seen in Figure 4.3 (b). This may be due to the fact that all images have to be resized to 600×600 pixels before being fed into the network, resulting in significant compression of small appearing sheep. If multiple sheep are close together, the detector may recognize them as a single sheep, as depicted in Figure 4.3 (c). Additionally, some objects on the field, such as water tanks, are incorrectly identified as sheep by the detector. See Figure 4.3 (d) for an example. Such false positives are also observed with RetinaNet and YOLOv9.

4.2.3 Simple CA Performance Improvement

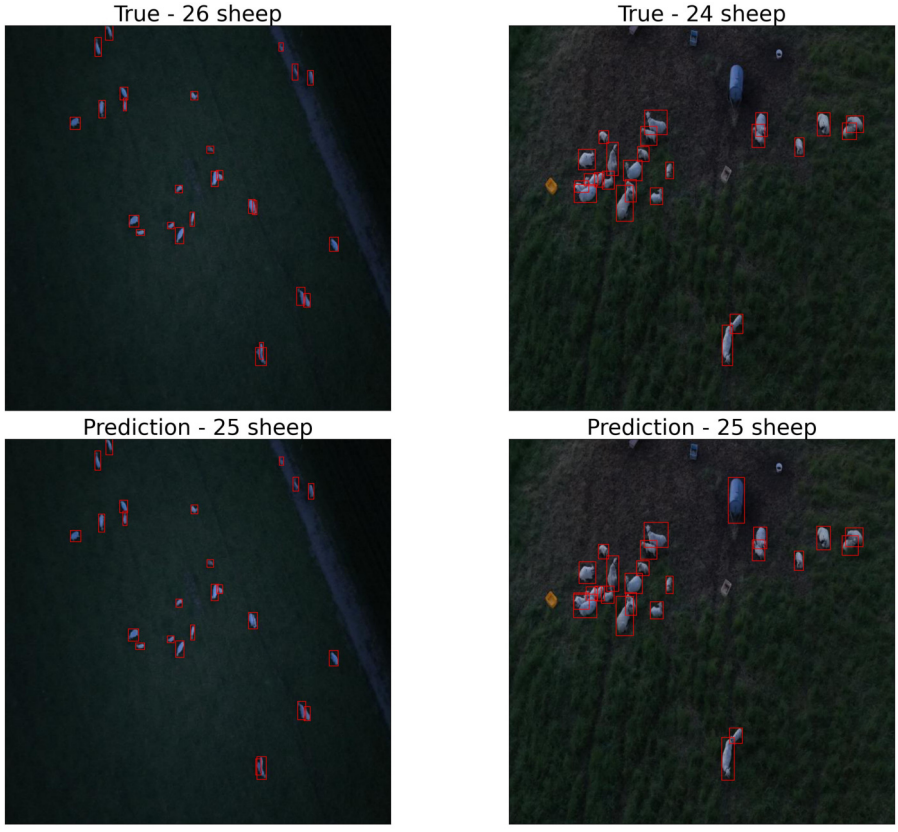
Enhancing the CA can be achieved through several straightforward methods beyond improving the detection models, such as averaging predictions. If capturing all sheep in a single image is feasible, multiple images from different perspectives can be taken. For each image, the sheep count is predicted. Due to occlusions or misleading objects, some images may yield predictions that are either too high or too low. Averaging the predictions across all images may result in a prediction closer to the true count.

The snow dataset is well-suited for this method, as the sheep are clustered closely together, allowing most images to capture all sheep in the pasture. However, the other datasets are not suitable for this method since the sheep are spread out, and the pastures are too extensive to be covered in one image with sufficient resolution. Applying this method to the snow dataset results in an improvement in CA by two to three percentage points. The highest score of 0.964 is achieved by YOLOv9, while Faster R-CNN achieves a score of 0.944, and RetinaNet achieves 0.918. Given that YOLOv9 now surpasses the CA95 threshold necessary for basic practicality in real-world use, the developed system may be sufficiently accurate for real-world applications in counting sheep on snow if the sheep are clustered closely together.



(a) Good predictions

(b) Sheep are too small



(c) Sheep are too close together

(d) False positive on water tank

Figure 4.3: True annotations and predictions by Faster R-CNN on dusk images

Chapter 5

Conclusions and Future Work

This concluding chapter revisits and discusses the central question of this research in Section 5.1. It starts with a summary of the work undertaken and the key results obtained. The main findings are reiterated and accompanied by a critical commentary on their implications and limitations. Subsequently, Section 5.2 provides recommendations for future research, proposing potential directions to further explore and build upon the outcomes of this study.

5.1 Conclusions

The primary objective of this thesis is to determine how accurately common object detection algorithms, such as Faster R-CNN, RetinaNet, and YOLOv9, can count sheep in aerial imagery captured under adverse conditions. To achieve this, three novel datasets featuring sheep during dusk, in rough terrain, and on snow-covered ground, captured by a UAV, are developed and published. Additionally, an existing dataset of white sheep on grassland is used for comparison. After tuning and training the algorithms on each dataset separately, their performance is evaluated on various test sets. The findings are analyzed to assess whether the observed accuracy is sufficient for real-world use and to evaluate the necessity of the novel datasets.

The algorithms trained exclusively on the grassland dataset perform well on this same dataset, matching the high precision levels previously reported by other researchers. All three tested detection algorithms substantially surpass the CA95 threshold required by farmers for real-world practicality on the grassland dataset. However, their performance degrades significantly when tested on all adverse datasets, suggesting that grassland datasets are not sufficient for reaching satisfactory per-

formance in adverse conditions.

When specifically trained on these unfavorable conditions, the algorithms perform significantly better, achieving a CA of 0.9 to 0.94 and an $mAP^{0.5}$ of around 0.9. These findings indicate that specific data is essential for achieving high accuracy in various adverse conditions.

Algorithms trained on all four datasets together achieve at least the same accuracy as those trained only on a single specific dataset, suggesting that condition-specific datasets and detectors may not be needed. This also indicates that datasets should include images captured under all possible conditions and from various perspectives. Despite the high CA achieved by these generalized detectors, they still fall just short of the CA95 threshold in adverse conditions. Therefore, none of the trained detectors are sufficient for real-world applications in such challenging environments.

Enhancing the CA may be achievable by averaging multiple sheep count predictions from different viewpoints of the same pasture. When applied to the snow dataset, this technique allows the CA to surpass the CA95 threshold. However, this approach requires that all sheep are visible in a single image taken from a reasonable distance. Therefore, it is only suitable for very specific conditions, and its practical use is very limited.

Achieving a CA above 0.95 on the grassland and snow datasets, and nearly reaching this threshold on the dusk and rough terrain datasets, the best detectors demonstrate promising results. While the detectors still have difficulties in correctly detecting sheep in extremely adverse conditions, UAVs and object detection algorithms might still be viable for sheep counting under moderately adverse conditions, such as on snow-covered ground and during early dusk. However, further testing is necessary to confirm the reported performance on slightly varied data, as the test split used in this evaluation is a random selection of images from the same flights as used for training. Additionally, the scope of this thesis is limited to counting sheep in single images. Consequently, if not all sheep in a pasture can be covered with a single image, new techniques beyond those presented here are necessary.

Beyond the presented results, it is important to note that at flight heights above 20 meters, sheep do not exhibit signs of disturbance from the commercial UAV used for capturing the adverse datasets. Furthermore, nearly all sheep captured in this thesis can be accurately annotated by an untrained human. Consequently, the use of UAVs for accurately counting sheep under adverse conditions may soon become feasible, given the continuous advancements in object detection algorithms.

5.2 Future Work

The results achieved in this thesis are promising and may be nearly sufficient for certain real-world applications. However, improving the performance, particularly the CA of the models, is necessary for larger-scale deployment in real-world scenarios.

Building upon the findings and limitations presented previously, there are several promising avenues for future research with the goal of enhancing the capabilities and applications of UAVs in monitoring sheep. These areas include:

- **Annotating all images:** Completing the annotation of the remaining 1,552 images in the snow, dusk, and rough terrain datasets would provide a more comprehensive foundation for training and evaluation, potentially leading to improved accuracy and robustness of the models.
- **Counting sheep on whole pastures:** While this thesis focuses on counting sheep in individual images, real-world applications require the sheep count of an entire pasture. Increasing the flight height to cover large pastures in a single image is often impractical due to flight restrictions and reduced resolution. Consequently, innovative solutions are needed to overcome this issue.
- **Detecting fences:** Instead of detecting missing sheep after the fact, a more effective strategy might involve models that detect fences and identify potential breaches. This would enable the implementation of countermeasures before any sheep are lost.
- **Assessing sheep health:** In addition to solely counting the sheep, UAVs could be equipped with algorithms that assess the health of sheep by detecting signs of limping or identifying sheep that are turned on their backs. Consequently allowing timely intervention and care.
- **Using infrared cameras:** Employing infrared cameras, which can detect the thermal signature of animals even in total darkness, could improve sheep detection in low-light-conditions or dense vegetation, where visual detection is challenging.

Most importantly, the developed solutions need to be integrated into a usable product. This involves designing a user-friendly interface that allows farmers to effortlessly deploy and operate UAVs for sheep monitoring with clear instructions, automated processes, and intuitive controls. Ensuring the product's usability is crucial for its adoption and effectiveness in real-world farming environments.

Appendix A

Best Hyperparameters

A.1 Faster R-CNN

Training & Validation Dataset	Momentum	Weight decay	Learning rate	Epoch	Confidence Threshold
Grass	0.99	0.0001	0.005	35	0.3
Snow	0.9	0.0001	0.05	45	0.57
Dusk	0.97	0.0005	0.01	25	0.5
Rough Terrain	0.97	0.01	0.0005	42	0.5
All	0.8	0.005	0.005	40	0.71

Table A.1: Best hyperparameters for Faster R-CNN

A.2 RetinaNet

Training & Validation Dataset	Momentum	Weight decay	Learning rate	Epoch	Confidence Threshold
Grass	0.97	0.001	0.005	34	0.5
Snow	0.9	0.0005	0.01	24	0.49
Dusk	0.9	0.005	0.005	41	0.51
Rough Terrain	0.95	0.01	0.005	36	0.51
All	0.97	0.0005	0.005	38	0.49

Table A.2: Best hyperparameters for RetinaNet

A.3 YOLOv9

Training & Validation Dataset	Momentum	Weight decay	Learning rate	Epoch	Confidence Threshold
Grass	0.90	0.0005	0.0100	50	0.29
Snow	0.89	0.0004	0.0100	50	0.27
Dusk	0.92	0.0005	0.0105	50	0.11
Rough Terrain	0.9500	0.0008	0.0081	50	0.11
All	0.9371	0.0005	0.0095	50	0.3

Table A.3: Best hyperparameters for YOLOv9

Bibliography

[1] Michael Negnevitsky. *Artificial intelligence: a guide to intelligent systems*. Addison-Wesley, New York Munich, 2. ed., [nachdr.] edition, 2005.

[2] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson, 4. ed. edition, 2020.

[3] Reinhard Klette. *Concise Computer Vision - An Introduction into Theory and Algorithms*. Undergraduate Topics in Computer Science. Springer, 2014.

[4] Ce Zheng, Wenhan Wu, Chen Chen, Taojiannan Yang, Sijie Zhu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah. Deep Learning-based Human Pose Estimation: A Survey. *ACM Comput. Surv.*, 56(1):11:1–11:37, 2024.

[5] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.

[6] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Networks Learn. Syst.*, 30(11):3212–3232, 2019.

[7] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object Detection in 20 Years: A Survey. *Proc. IEEE*, 111(3):257–276, 2023.

[8] Keiron O'Shea and Ryan Nash. An Introduction to Convolutional Neural Networks. *CoRR*, abs/1511.08458, 2015. arXiv: 1511.08458.

[9] Dimosthenis C. Tsouros, Stamatia Bibi, and Panagiotis G. Sarigiannidis. A Review on UAV-Based Applications for Precision Agriculture. *Inf.*, 10(11):349, 2019.

[10] Jayme Garcia Arnal Barbedo. A review on the use of unmanned aerial vehicles and imaging sensors for monitoring and assessing plant stresses. *Drones*, 3(2):40, 2019. Publisher: MDPI.

[11] Ignazio Gallo, Anwar Ur Rehman, Ramin Heidarian Dehkordi, Nicola Landro, Riccardo La Grassa, and Mirco Boschetti. Deep Object Detection of Crop Weeds: Performance of YOLOv7 on a Real Case Dataset from UAV Images. *Remote. Sens.*, 15(2):539, 2023.

[12] Mohammed A. Alanezi, Mohammad Shoaib Shahriar, Md Bakhtiar Hasan, Sabbir Ahmed, Yusuf A. Sha'aban, and Houssem Rafik El-Hana Bouchekara. Livestock Management With Unmanned Aerial Vehicles: A Review. *IEEE Access*, 10:45001–45028, 2022.

[13] Oliver Doll and Alexander Loos. Comparison of Object Detection Algorithms for Livestock Monitoring of Sheep in UAV images. In *Int. Workshop Camera traps, AI, and Ecology*, 2023.

[14] Shelemia Nyamuryekung'e, Andres F Cibils, Richard E Estell, and Alfredo Gonzalez. Use of a UAV-Mounted Video Camera to Assess Feeding Behavior of Raramuri Criollo Cows. In *10th International Rangeland Congress*, page 1113, 2016.

[15] Liang Wang, Bo Hu, Yuecheng Hou, and Huijuan Wu. Lightweight Sheep Head Detection and Dynamic Counting Method Based on Neural Network. *Animals*, 13(22):3459, 2023. Publisher: MDPI.

[16] Fabricio de Lima Weber, Vanessa Aparecida de Moraes Weber, Pedro Henrique de Moraes, Edson Takashi Matsubara, Débora Maria Barroso Paiva, Marina de Nadai Bonin Gomes, Luiz Orcírio Fialho de Oliveira, Sérgio Raposo de Medeiros, and Maria Istela Cagnin. Counting cattle in UAV images using convolutional neural network. *Remote Sensing Applications: Society and Environment*, 29:100900, 2023. Publisher: Elsevier.

[17] Beibei Xu, Wensheng Wang, Greg Falzon, Paul Kwan, Leifeng Guo, Zhiguo Sun, and Chunlei Li. Livestock classification and counting in quadcopter aerial images using Mask R-CNN. *International Journal of Remote Sensing*, 41(21):8121–8142, 2020. Publisher: Taylor & Francis.

[18] Farah Sarwar, Anthony Griffin, Saeed Ur Rehman, and Timotius Pasang. Detecting sheep in UAV images. *Comput. Electron. Agric.*, 187:106219, 2021.

[19] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual*

Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, pages 91–99, 2015.

[20] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(2):318–327, 2020.

[21] Chien-Yao Wang, I.-Hau Yeh, and Hong-Yuan Mark Liao. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. *CoRR*, abs/2402.13616, 2024. arXiv: 2402.13616.

[22] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 580–587. IEEE Computer Society, 2014.

[23] Ross B. Girshick. Fast R-CNN. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1440–1448. IEEE Computer Society, 2015.

[24] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[25] Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollár, Kaiming He, and Ross B. Girshick. Benchmarking Detection Transfer Learning with Vision Transformers. *CoRR*, abs/2111.11429, 2021. arXiv: 2111.11429.

[26] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the Gap Between Anchor-Based and Anchor-Free Detection via Adaptive Training Sample Selection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9756–9765. Computer Vision Foundation / IEEE, 2020.

[27] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las*

Vegas, NV, USA, June 27-30, 2016, pages 779–788. IEEE Computer Society, 2016.

[28] Riis. Aerial sheep dataset, 2022. https://universe.roboflow.com/riis/aerial-sheep [Accessed: 25.5.2024].

[29] DJI. Dji air 3 - specs, 2024. https://www.dji.com/ch/air-3/specs [Accessed: 10.6.2024].

[30] Lars Wuethrich. Snowsheepuav dataset, 2024. https://universe.roboflow.com/lars-wuethrich/snowsheepuav [Accessed: 25.6.2024].

[31] Lars Wuethrich. Dusksheepuav dataset, 2024. https://universe.roboflow.com/lars-wuethrich/dusksheepuav [Accessed: 25.6.2024].

[32] Lars Wuethrich. Roughterrainsheepuav dataset, 2024. https://universe.roboflow.com/lars-wuethrich/roughterrainsheepuav [Accessed: 25.6.2024].

[33] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. *CoRR*, abs/1803.09820, 2018. arXiv: 1803.09820.

[34] University of Bern. High performance computing (hpc), 2016. https://www.unibe.ch/universitaet/campus__und__infrastruktur/rund_um_computer/soft_und_hardware/hardware/hochleistungsrechner_hpc_grid/index_ger.html [Accessed: 15.5.2024].

[35] Ultralytics. Hyperparameter tuning, 2024. https://docs.ultralytics.com/guides/hyperparameter-tuning [Accessed: 10.5.2024].