Registration number
/…../…../…../…../…..

Master Thesis

To obtain the master's degree

Sheep Detection, Tracking and counting from aerial images using Deep Learning

Publicly supported on ……/……/…………

By
MEHAYA Mohammed Elmehdi
DJEKABA Fatima

Jury Members

M. MAHDJOUB Youcef	MCB	Univ. Ghardaia	President
M. BOUHANI Abdelkader	MAA	Univ. Ghardaia	Examiner
M. ADJILA Abderrahmane	MAA	Univ. Ghardaia	Supervisor

University year 2020/2021

ABSTRACT

Object detection is widely used in the field of computer vision. Furthermore, it can be harnessed in agriculture and farming, especially with the new methods that achieve promising results. Nowadays, the problem is tackled using either traditional machine learning methods that use computer vision techniques or deep learning methods. In this work, we investigate the deep learning state-of-the-art tools to create a smart system for detecting, tracking and counting sheep using aerial images captured by a drone. In the process, we gather sufficient data with good quality and use it to train a model dependent on the YOLOv4 network. Next, we tackle the counting stage directly using an innovative method that uses an imaginary line cutting the processed frame incrementing the counter whenever an intersection between the bounding box and the gate happens. However, we had to introduce an intermediate stage because of low performance. That intermediary is called tracking. The results obtained by the experiment are highly promising in detection with an mAP of 71% and 16.1274 % of avg loss function.

Keywords : Object detection, Object tracking, Object counting, deep learning, YOLO, Deep SORT, Aerial images, sheep.

RÉSUMÉ

La détection d'objets est largement utilisée dans le domaine de la vision par ordinateur. De plus, il peut être exploité dans l'agriculture et l'élevage, en particulier avec les nouvelles méthodes qui donnent des résultats prometteurs. De nos jours, le problème est résolu en utilisant soit des méthodes traditionnelles d'apprentissage automatique qui utilisent des techniques de vision par ordinateur, soit des méthodes d'apprentissage en profondeur. Dans ce travail, nous étudions les outils de pointe d'apprentissage en profondeur pour créer un système intelligent de détection, de suivi et de comptage des moutons à l'aide d'images aériennes capturées par un drone. Dans le processus, nous recueillons suffisamment de données de bonne qualité et les utilisons pour former un modèle dépendant du réseau YOLOv4. Ensuite, nous abordons l'étape de comptage directement en utilisant une méthode innovante qui utilise une ligne imaginaire coupant la trame traitée en incrémentant le compteur chaque fois qu'une intersection entre la boîte englobante et la porte se produit. Cependant, nous avons dû introduire une étape intermédiaire en raison des faibles performances. Cet intermédiaire s'appelle le suivi. Les résultats obtenus par l'expérience sont très prometteurs en détection avec un mAP de 71 et 16,1274 de fonction de perte moyenne.

Mots clés : Détection d'objets, Suivi d'objet, comptage, l'apprentissage en profondeur, YOLO, SORT en profondeur, troupeau de moutons.

ملخص

يستخدم اكتشاف الكائنات على نطاق واسع في مجال رؤية الكمبيوتر. علاوة على ذلك، يمكن تسخيرها في الزراعة، خاصة مع الأساليب الجديدة التي تحقق نتائج واعدة. في الوقت الحاضر، يتم التعامل مع المشكلة إما باستخدام طرق التعلم الآلي التقليدية التي تستخدم تقنيات رؤية الكمبيوتر أو طرق التعلم العميق. في هذا العمل، نتحرى عن أحدث أدوات التعلم العميق لإنشاء نظام ذكي لاكتشاف وتعقب وعد الأغنام باستخدام الصور الجوية التي تم التقاطها بواسطة طائرة بدون طيار. في هذه العملية، نجمع بيانات كافية بجودة جيدة ونستخدمها لتدريب نموذج يعتمد على شبكة اليولو النسخة الرابعة.بعد ذلك، نتعامل مع مرحلة العد مباشرةً باستخدام طريقة مبتكرة تستخدم خطًا وهميًا يقطع الإطار المعالج ويزيد العداد كلما حدث تقاطع بين الصندوق المحيط والبوابة. ومع ذلك ، كان علينا تقديم مرحلة وسيطة بسبب الأداء المنخفض. هذا الوسيط يسمى التعقب. النتائج التي تم الحصول عليها من خلال التجربة واعدة للغاية في الكشف مع متوسط الدقة الرئيسي 71 ٪ و 16.1274٪ من دالة الخسارة المتوسطة.

الكلمات المفتاحية : اكتشاف الكائنات، تتبع الكائنات،عد الكائنات، التعلم العميق،الصور الجوية، الاعنام.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

LIST OF ABBREVIATIONS

ML Machine Learning

ANN Artificial Neural Network

NN Neural Network

DNN Deep Neural Network

AI Artificial Intelligence

DL Deep Learning

CNN Convolutional Neural Network

RNN Recurrent neural network

LSTM Long short-term memory

CV Computer Vision

KF Kalman Filter

RBM Restricted Boltzaman Machine

ReLu Rectifed Linear Unit

TanH Tangent Hyperbolic

SSD Single Shot MultiBox Detector

YOLO You only look once

R-CNN Region-based Convolutional Neural Network

SORT Simple Online and Realtim

Deep SORT Simple Online and Realtime Tracking with a Deep Association Metric

SVM Support Vector Machine

MOT Multiple Object Tracking

mAP Mean Average Precision

MOTA	Multi-object tracking accuracy
MOTP	Multi-object tracking precision
IoU	Intersection over Union
FPS	Frames Per Second

GENERAL INTRODUCTION

Given the economic conditions that our country is going through, and the continuous attempts to get rid of the full dependency on oil and migrate to investments in agriculture. And given the services offered by modern Artificial Intelligence nowadays due to Deep learning we decided to investigate the problems encountered by farmers and contribute to a solution.

Among the difficulties facing farmers, there is detecting and counting the number of sheep flocks periodically and continuously. This real-life situation can be easily interpreted as an object detection problem which is the task of detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos.

Several researchers had several approaches for similar problems, as some of them used hand-engineered features and classic machine learning algorithms. While others, who were more successful exploited deep learning neural networks.

In this research, we design a deep learning based system that is destined to be integrated on a drone that count the number of sheep starting with a stream of images. To train it we collect up to 3,679 images in different climatic conditions, we choose an altered simplified version of YOLOv4 as a predictor as well as for tracking we choose a deep SORT algorithm to reach our goal, which is counting, through a special method that we called the gate.

This document is structured as follows :

In the first chapter we will explain the general concepts related to our project, namely Machine learning, Deep learning and computer vision.

In the second chapter we define the problems Object Detection, tracking and counting, and explain the state of the art approaches followed by researchers in the context of object counting systems.

In the third chapter we implement a smart sheep detection and counting system, using images captured by a drone. We talk about the architecture of our model. Till the dataset that we gathered and pre-processed for the task, the model trained and tested, not only the method we developed for the counting and the difficulties we encountered, but also solving these challenges through the tracking algorithm and last but not least we reveal the experimental results for our system.

CHAPITRE 1 PRELIMINARY

1.1 Introduction

In this chapter we will explain the general concepts related to our project, namely Machine learning, Deep learning and computer vision

1.2 Machine learning

1.2.1 Definition

Machine Learning (ML) is a subfield of Artificial Intelligence (AI) which enables computers to learn from their past experiences [59]. Arthur Samuel said "the field of study that gives computers the ability to learn without being explicitly programmed" meaning that ML enables machines to monitor a pattern and attempt to imitate it in a direct or an indirect way [75].
Moreover, ML uses statistical methods to allow machines to learn by them selves using the provided data and to make accurate predictions. Machine learning introduced a new approach to programming where the system is trained rather than explicitly programmed as opposed to to classic programming.

1.2.2 Pipeline process of Machine learning

The pipeline process of ML goes through he several stages, represented in Figure 1.1.

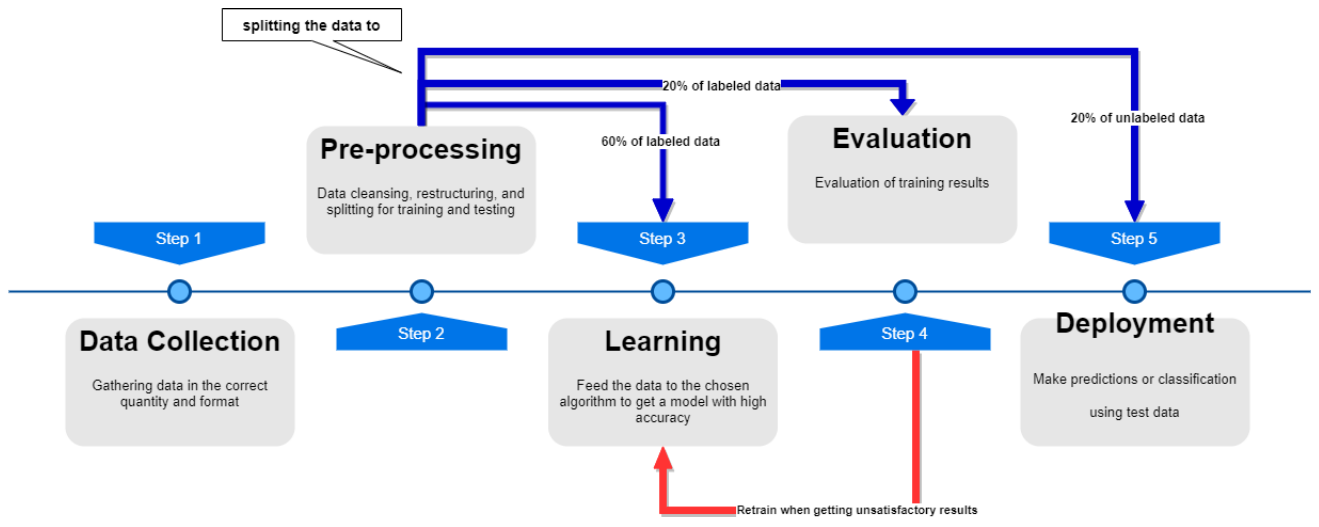

splitting the data to
20% of labeled data
60% of labeled data
20% of unlabeled data
Pre-processing
Data cleansing, restructuring, and splitting for training and testing
Evaluation
Evaluation of training results
Step 1
Step 2
Step 3
Step 4
Step 5
Data Collection
Gathering data in the correct quantity and format
Learning
Feed the data to the chosen algorithm to get a model with high accuracy
Deployment
Make predictions or classification using test data
Retrain when getting unsatisfactory results

FIGURE 1.1 – Machine learning pipeline process.

1. Data Collection

This stage is the starting point where a set of raw data is collected and processed in the upcoming phase. the quantity and quality of the collected data will highly effect. the accuracy of the model.

2. Pre-processing

Data preprocessing is a strategy for data mining involving the conversion to an intelligible format of raw data. collected data generally contains numerous errors and it is partial, inconsistent, and missing particular conduct or trends. The method of obtaining useful data for the ML algorithm includes feature removal and sizing, selection of features, dimensionality reduction, and sampling. The Data Pre-Processing result is used for model training, evaluating and testing.

3. Learning (Machine learning Algorithm)

In this stage, one of the learning algorithms is used to process pre-processed data aiming to extract appropriate patterns for different situation use. The learning algorithm is chosen according to the type of data and the problem to be solved. The purpose is to of the learning phase is to reach the best model out of an ensemble of infinite possible models.

4. Evaluation

Throughout the training phase, the model is continuously evaluated using the labels attached to the training data and the correct/wrong predictions made by the model and these are mandatory for calculating the training accuracy and loss and other performance criterions.

5. Deployment

In order to approximate the performance of the model in a real unseen use-case, we use a never used set of data called test data.

1.2.3 Types of Machine Learning

Supervised Learning

Assuming that each example in the training data comprises of a pair (x_i, y_i), where x_i is the input to be fed into the predictor and y_i is the ground-truth label of the input, this kind of data is used in supervised learning algorithms.
While training, the prediction parameters are tuned to get outputs $f(x_i)$ as close to (y_i) as possible, This category encompasses the majority of classification and regression issues [59]. Among the common application of supervised learning is the example of detecting spam/non spam emails.

Unsupervised Learning

Unsupervised learning consists of deducing a function to represent a hidden structure from unlabeled training data. We only have a collection of input examples and the model tries to find any correlations between instances and detect outliers, aiming to cluster the data set instances in groups that behave similarly.

Semi-Supervised Learning

Learning in a Semi-Supervised Environment The dataset for semi-supervised learning includes both labeled and unlabeled instances. The number of unlabeled examples is usually much greater than the number of labeled examples. A semi-supervised learning algorithm's goal is the same as a supervised learning algorithm's goal. The assumption is that by using a large number of unlabeled examples, the learning algorithm will be able to find (compute) a better model.

Reinforcement Learning (RL)

Reinforcement learning is a type of machine learning in which algorithms choose the best actions to take in a given situation in order to maximize reward. Unlike supervised learning, where the predictor is given ground-truth labels, the algorithm here must learn by trial and error to find the best outputs [59],There are several RL algorithms that achieve great results by creating models with amazing performance.

1.2.4 Computer Vision

Computer Vision (CV) is sub-field of AI that concentrate on the problem of helping computers to see.

As Simon J et al, knew him, where he said : ”At an abstract level, the goal of computer vision problems is to use the observed image data to infer something about the world.” In addition to Jan Reik Solem, when he knew him as follows, he said : ”Computer vision is the automated extraction of information from images. Information can mean anything from 3D models, camera position, object detection and recognition to grouping and searching image content”, That is, it depends on understanding the content of the image and extracting a description for it.
In recent periods, this field has witnessed rapid development due to the huge amount of data. The era has witnessed an explosion of information so that this much data is used in training and raising the effectiveness of computer vision.

1.2.5 Metrics of measuring the accuracy

There are many metrics that are used to evaluate the effectiveness of work, including :

— Metrics of measuring the accuracy of object detectors.

1. Precision : Precision is a measure of what proportion of predicted positives were truly positive.It is calculated according to the following equation1.1

$$Precision = (TP)/(TP + FP). \tag{1.1}$$

2. Recall : Recall is a measure of what proportion of actual positives were classified correctly.It is calculated according to the following equation1.2

$$Recall = (TP)/(TP + FN). \tag{1.2}$$

3. F-measure :

$$F - measure = (2 \times Precision \times Recall)/(Precision + Recall). \tag{1.3}$$

4. Average Precision(AP) : Average precision is a measure that combines recall and precision for ranked retrieval results.For one information need.
5. Mean Average Precision(mAP) : mean average precision is the average of average precision in some situations for each class. But they imply the same thing in some contexts.
   whereas :
   -TP :true positive .
   - FP : false positive.
   - FN :false negative .

6. Intersection Over Union (IOU) : The values IOU is between zero and one as it is calculated by dividing the intersection area by the union area of two boxes. The higher the IOU, the more successful the algorithm, and vice versa.

— Metrics of measuring the accuracy of object tracking .

1. Multi-object tracking accuracy (MOTA) : Summary of overall tracking accuracy in terms of false positives, false negatives and identity switches.
2. Multi-object tracking precision (MOTP) : Summary of overall tracking precision in terms of bounding box overlap between ground-truth and reported location.
3. Identity switches (ID) : Number of times the reported identity of a ground-truth track changes.

1.2.6 Machine learning application domains

Due to the tremendous development that the world has witnessed in the field of computer technology and science, machine learning techniques have spread due to their potential to solve problems in various fields such as health care, finance, retail, travel and media...etc. Figure **??**represents some of the applications of machine learning, we will explain the most important of them as follows :

1. Image Recognition It is an approach to catalog and detect a feature or object in the digital image. This technique is adopted for more advanced analyses, such as pattern recognition, face detection, or face recognition.

2. Speech Recognition It is the ability to recognize speech and convert it to text. It was a difficult problem to solve until recent advances in machine learning.

3. Self-driving cars The ability to navigate a vehicle without human intervention. This requires several capabilities that are made possible by recent advances in machine learning, including perception (detection and tracking of fixed and moving objects).

4. Automatic Language Translation understanding and forming linguistic constructions without formal grammar training, just like a child learning his mother tongue. Recent advances in machine learning have significantly improved the accuracy of translation over previous methods.

5. Sentiment Analysis a machine learning application that determines the emotion or opinion of the person speaking or writing. For example, if someone has written a review or an email (or some other form of document), a sentiment analyzer will instantly discover the actual thought and tone of the text. This sentiment analysis app can be used to analyse websites based on reviews, business intelligence apps,.. etc.

As another type of ML, We have deep learning which will be the topic of our discussion in the next section

1.3 Deep learning

Many applications of machine learning required the help of experts to extract hand engineered features from the data, and those were the input of machine learning algorithm. This task makes the operation very difficult. Fortunately, Big Data led to an increase in computing performances which allowed the resurrection of the artificial neural networks under t he new name **Deep Learning**. This new approach eliminates manual feature extraction, and works well with large amounts of data.

Deep Learning (DL) architectures made a revolution in AI due to their ability to process unstructured data (text, image, audio...) in large volumes[56], and they improved results in many domains such as computer vision, speech recognition, and machine translation. Deep learning techniques solve many problems in many areas of the economy, health, transportation, trade, finance, and energy [2].

1.3.1 Deep learning historic

The emergence of the concept of DL dates back to the forties, but it did not receive much interest and popularity in that period because there is no large data set in addition to the fact that the devices were not that powerful.

In the twentieth century, it received great attention due to the phenomenon of data explosion and the manufacture of devices with high capacity for data processing.

Deep learning is based on the notion of artificial neural networks, and from here we begin by mentioning the most important stations in the history of DL,Figure1.2 [57] represents the timeline of the history of deep learning, where :
- 1943. McCulloch Pitts Neuron-Beginning : J. Mc Culloch and W. Pitts modeling a biological neuron, they proved that a simple formal neuron can be realized as logic, arithmetic, and symbolic complex functions[55].
- 1958. F. Rosenblatt developed a perceptron model which is the first artificial system capable of learning from experience [69]. B. Widrow and Hoff also introduced the Adaline (ADAptive LINEAR) model [82], which is the basic model for multilayered networks.
- 1960. The First Backpropagation Model was completed by Henry J. Kelleyin in a research article under the title, "Gradient Theory of Optimal Flight Paths" which was the first version of this model[40].
- 1962. The backpropagation model based on a simple series derivative rule is explained, rather than the dynamic programming used by earlier backpropagation models.
- 1965. Brith of Multilayer Neural Network Created byAlexey Grigoryevich et al.
- 1980. The first convolutional neural network was created by Kunihiko Fukushima, which allows the recognition of visual patterns.
- 1982. John Hopfield creates Hopfield Network, a recurrent neural network that is a memory system that can process content.

- 1982. Proposal ForBackprogation In Artificial Neural Network (ANN).
- 1986. Implementation of Backpropagation in a neural network has been done. Resulting in the training of a complex deep neural network easily, This was done through a paper presented by Geoffrey Hinton et al.
- 1986.Paul Smolensky came up with a concept Restricted Boltzaman Machine (RBM) that came up and applied it for building recommender systems.
- 1989. Yann LeCun uses backpropagation to train convolutional neural networks to recognize handwritten numbers.
- 1991. Sepp Hochreiter identifies the problem of vanishing gradient which can make the learning of Deep Neural Network (DNN) extremely slow and inefficient.
- 1997. where a type of Recurrent neural network (RNN) called Long short-term memory (LSTM) by Sepp Hochreiter et al[31].
- 2006. Geoffrey Hinton et al published the paper [29] in which they stacked multiple RBMs together in layers and called them Deep Belief Networks. The training process is much more efficient for large amounts of data.
- 2009. a data set ImageNet is launched.
- 2011. Solving the problem for vanishing Gradient By relying on the Rectifed Linear Unit (ReLu) activation function [23].
- 2012.AlexNet is a Convolutional Neural Network (CNN) model, wins in Imagenet is an images classifications contest with an accuracy equal to 84%. This victory gave a new breath to the field of deep learning.
- 2014. Generative Adversarial Neural Network also known as GAN is created by Ian Goodfellow.
- 2015. The big companies have recall artificial neural network experts to develop new applications with high efficiency. AlphaGo 1 relied on CNN to improve the performance of game Go. It achieved results that exceeded accuracy and speed than humans.
- 2019. Due to their huge contribution in the areas of DL and artificial intelligence AI, Geoffrey Hinton, Yoshua Bengio, and Yann LeCun are the winners of the Turing Award 2018.

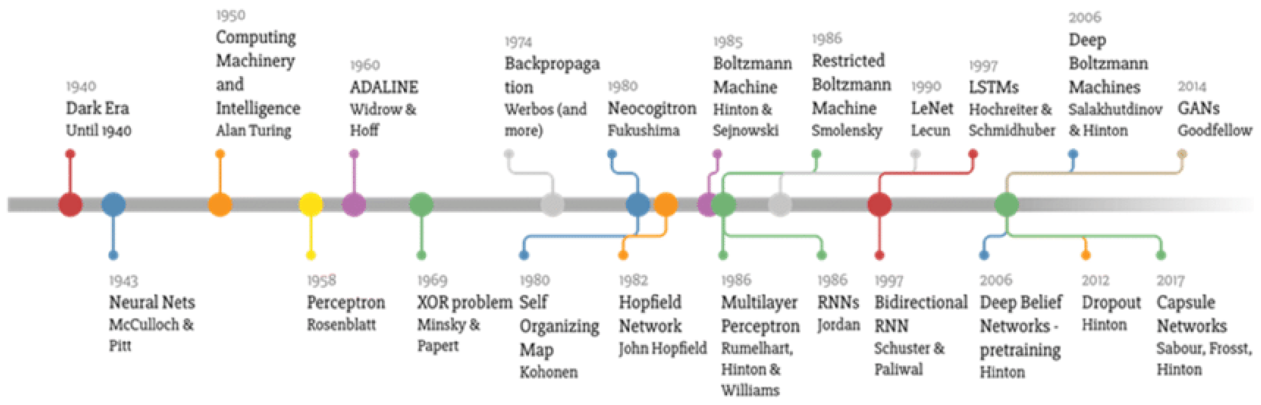

1940 Dark Era Until 1940
1943 Neural Nets McCulloch & Pitt
1950 Computing Machinery and Intelligence Alan Turing
1958 Perceptron Rosenblatt
1960 ADALINE Widrow & Hoff
1969 XOR problem Minsky & Papert
1974 Backpropagation Werbos (and more)
1980 Self Organizing Map Kohonen
1980 Neocogitron Fukushima
1982 Hopfield Network John Hopfield
1985 Boltzmann Machine Hinton & Sejnowski
1986 Multilayer Perceptron Rumelhart, Hinton & Williams
1986 Restricted Boltzmann Machine Smolensky
1986 RNNs Jordan
1990 LeNet Lecun
1997 Bidirectional RNN Schuster & Paliwal
1997 LSTMs Hochreiter & Schmidhuber
2006 Deep Belief Networks - pretraining Hinton
2006 Deep Boltzmann Machines Salakhutdinov & Hinton
2012 Dropout Hinton
2014 GANs Goodfellow
2017 Capsule Networks Sabour, Frosst, Hinton

Figure 1.2 – Deep learning history time line.

1.3.2 Deep learning advantages

ML algorithms work well for a wide variety of problems. However, they have failed to solve some major AI problems such as speech recognition and object recognition. deep learning solves those problems. The real potential of deep Learning was discovered after the appearance of Big Data and the increase of computing performances.
First, feature extraction in deep learning is performed automatically. Contrary to ML, where the feature extraction is performed, in general, manually which is a difficult and costly in time and it requires a domain expert.
DL is also popular for its ability to process big amounts of data with high performance. Opposite to traditional ML algorithms which are only useful for small amounts of data. This fact is depicted in Figure 1.3([3]). the Chinese scientist Andrew Ng, which is a famous researcher of DL said : "The analogy to deep learning is that the rocket engine is the deep learning models and the fuel is the huge amounts of data we can feed to these algorithms."

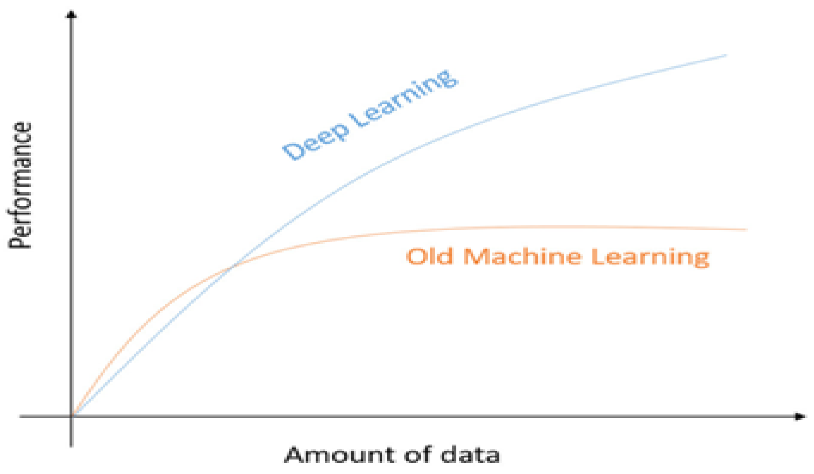


FIGURE 1.3 – The performance of deep learning and Machine learning with respect to the amount of data.

We will next introduce the most important deep learning tool, that is the artificial neural network.

1.3.3 Neural Networks

By inspiration, a Neural Network (NN) is simulated from the human brain. it has somewhat an ability of distinguishing between things.

Biological Neural Networks

The human brain has billions of interconnected neurons. Each neuron is composed of three main elements[7] as shown in Figure1.4 :

— Dendrites : input canals which receive signals from previous neurons.

— Body (Soma) : contains the kernel (Nucleus) which is responsible for the conduct of vital cellular activities.

— Axon : send signals into the synapse (link between two neurons).

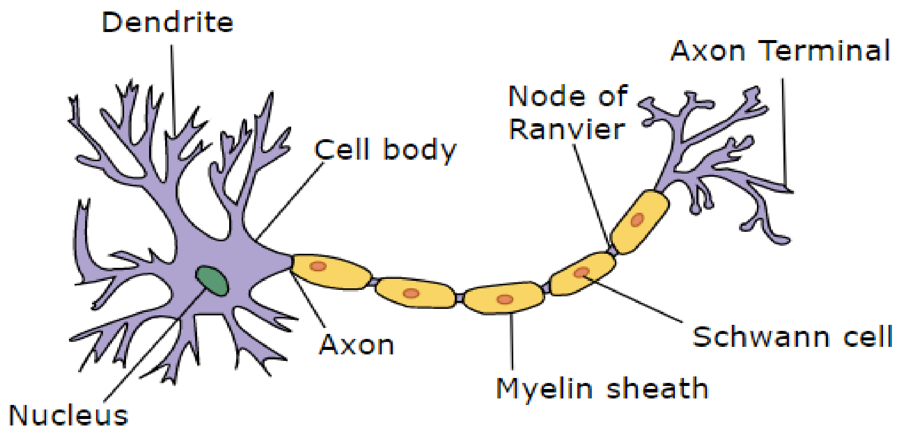


FIGURE 1.4 – The biological neuron.

Biological neurons are cells found in the cerebral cortex, dendrites receive electrical signals as weighted inputs. The inputs are used to compute an output signal by the cell body. Once the output signal reaches a specific value, it passes through the axon wire which is connected to other neurons using the synaptic terminal.

Artificial Neural Network (ANN)

An artificial neural network (ANN) is an artificial representation of the biological neural network. Its architecture can be represented as a directed graph that has a number of neurons interconnected forming many layers (Input layer, Hidden layers, Output layer), as shown Fig.1.5. Each artificial neuron takes a set of inputs, multiplies each input by a weight, computes the sum of all these weighted inputs plus a constant value called a Bias. The weighted sum is then fed to a function called an Activation Function, which forwards its output to another neuron as input.

Activation Functions

The activation function or transfer function is a mathematical function. It determines whether or not the neuron should be activated, Non-linearity is necessary for this function. There are many types of nonlinear activation functions Such as Sigmoid, Softmax, ReLu, TanH, etc. The type of

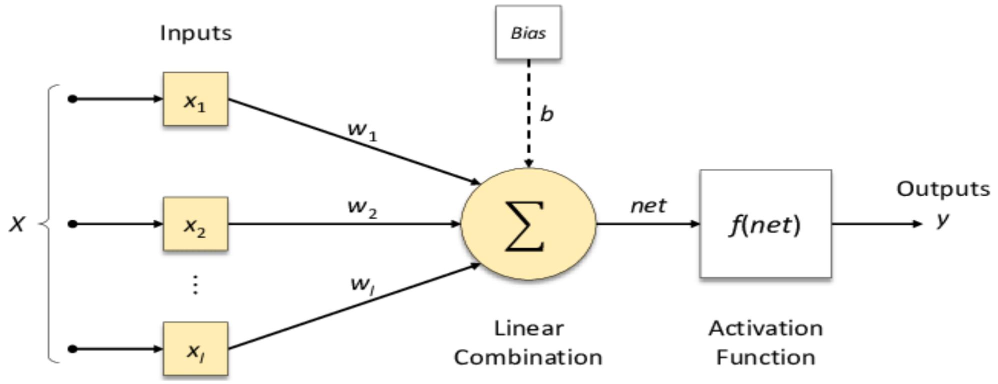


FIGURE 1.5 – The artificial neuron.

activation function is chosen according to the problem to be solved. [7], [60] Let's take a look at some of the most common activation functions.

1. **Sigmoid :** It is the most popular and oldest activation method. It restricts the values to a limit of 0 to 1. If the values are extremely large positive numbers, it returns 1. If the values are extremely large negative numbers, it returns 0. It's mostly used in binary classification. The Sigmoid function is defined mathematically by equation 1.4 and its derivative by equation 1.5. Figure1.6([7]) represent its plot.

$$sigmoid(x) = f(x) = \frac{1}{1+e^{-x}} \tag{1.4}$$

$$f^{'}(x) = f(x)(1-f(x)) \tag{1.5}$$

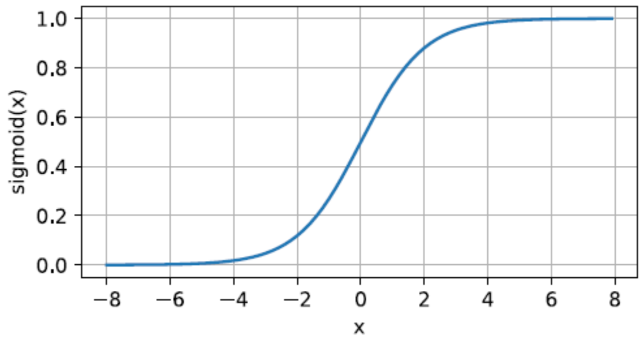


FIGURE 1.6 – The Sigmoid function plot.

The advantages of the Sigmoid function are making clear predictions as well as normalizing the output values. However some of its downfalls are the vanishing gradients problem plus its computational cost caused by its use for the exponential function. Despite these disadvantages, Sigmoid is still very popular with classification problems

2. **Tangent Hyperbolic (TanH) :** The hyperbolic tangent function, like the sigmoid function, squashes its inputs, converting them into elements in the -1 to 1 range. One of the pros

TanH has over sigmoid is that its gradients are stronger and the derivatives are steeper and that makes it less prone to cause the vanishing gradients problem. The TanH function is defined mathematically by equation 1.6 and its derivative by equation 1.7 and Figures 1.7([7]) represent its plot.

$$tanh(x) = f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{1.6}$$

$$f^{'}(x) = 1 - f(x)^2 \tag{1.7}$$

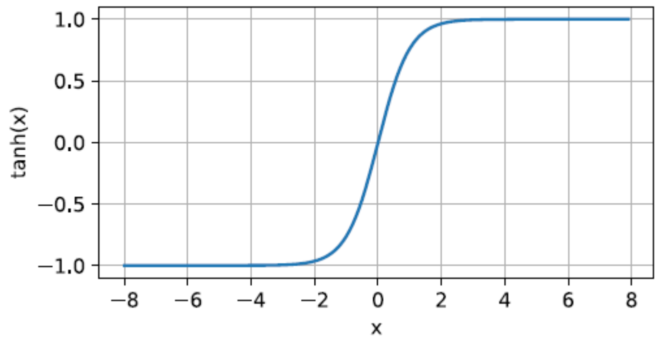


FIGURE 1.7 – The tanch function plot.

3. **ReLu :** The ReLu function Figure 1.8 [7]is the most used activation function, and it is defined by equation 1.8. The function outputs 0 if the input is negative, and it outputs the input its if it was positive.

$$ReLu(x) = f(x) = \max(0, x) \tag{1.8}$$

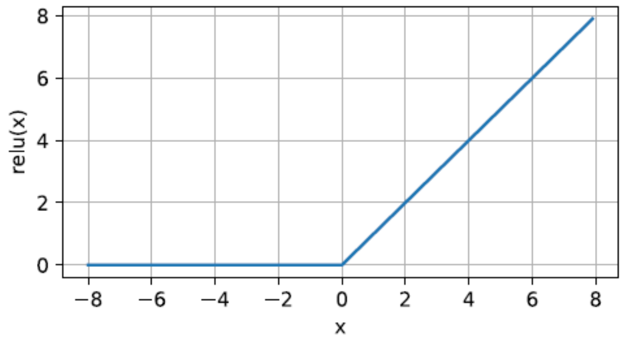


FIGURE 1.8 – The ReLu function plot.

4. **Softmax :** The softmax function usually used (but not only) for classification tasks. It allows building neural networks with several standardized outputs by given probabilities. Used in multiple class classification, k is the number of classes, The Softmax function is defined by equation 1.9.

$$f(x_i) = \frac{e^{x_i}}{\sum_{c=1}^{k} e^{x_c}} \tag{1.9}$$

It has many pros like the ability to handle multi class classification and the compatibility for output neurons.

Artificial Neural Networks iterations

There are two ways of visiting artificial neurons inside an artificial neural network :

1. **Forward Propagation :** In Forward propagation, signals travel only one way, from input to output. Otherwise, the output of any layer does not affect in the previous layer.
2. **Backward Propagation :** The backward propagation (feedback) has a signal traveling in both directions by inserting loops in the networks. That makes the network interactive and recurrent. On one hand, this property improves the network's performance, on the other hand the architecture can get much more complex.

In the next section, we will discuss the two most important types of artificial neural networks, starting with Convolutional neural networks.

1.3.4 Deep Neural Network architectures

There are several deep neural network architectures. Under this section we mention the two most important ones. Those being Convolutional Neural Networks and Recurrent Neural Networks.

Convolutional Neural Networks (CNN)

Convolutional neural networks (CNN) are one of the deep neural network architectures. It has a feed-forward propagation introduced in 1995 by Yann LeCun and Yoshua Bengio [45]. Before presenting the details of this network type, we first introduce the definition of the convolution operation.

The Convolution Operation Convolution is a mathematical operation is defined as follows : The convolution of two signals x and w with respect to time t is a signal calculated by equation 1.10 :

$$s(t) = (t)(x \times w) \tag{1.10}$$

Realistically, computers cannot process time in a continuous from, therefore, it is discrete. Thus, assuming that time t can take on only integer values, we can define the discrete convolution using equation 1.11.

$$s(t) = (x \times w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{1.11}$$

In equation 1.12, the signal x is the input, the signal w is called the convolution kernel whereas the output s(t) is referred to the feature map. In CNNs, the input and the kernel are usually multi-dimensional arrays containing a finite number of integer values. For that, we usually assume that their signals (x and w) are zero everywhere but the finite set of points for which we store the

values. As a result, we can transform the infinite summation into a finite summation.
Finally, we also use convolutions of more than one axis at a time. For example, if we use a two-dimensional input image I, we possibly also want to use a two-dimensional kernel K

$$S(i,j) = (I \times K)(i,j) = \sum_m \sum_n I(m,n)K(i-m,j-n) \tag{1.12}$$

We can put it another way, equation 1.13, because convolution is commutative :

$$S(i,j) = (K \times I)(i,j) = \sum_m \sum_n I(i-m,j-n)K(m,n) \tag{1.13}$$

Many Convolutional neural network (CNN) libraries use an alternative process to this function called cross-correlation, which is similar to the convolution process with a slight modification represented by equation 1.14, In addition to it Many machine learning libraries implement cross-correlation but call them a convolution.[24]

$$S(i,j) = (I \times K)(i,j) = \sum_m \sum_n I(i+m,j+n)K(m,n) \tag{1.14}$$

CNN Layers A CNN (Fig. 1.9) consists of several layers : convolutional layer, poling layer, flatting layer,and fully connected layer.

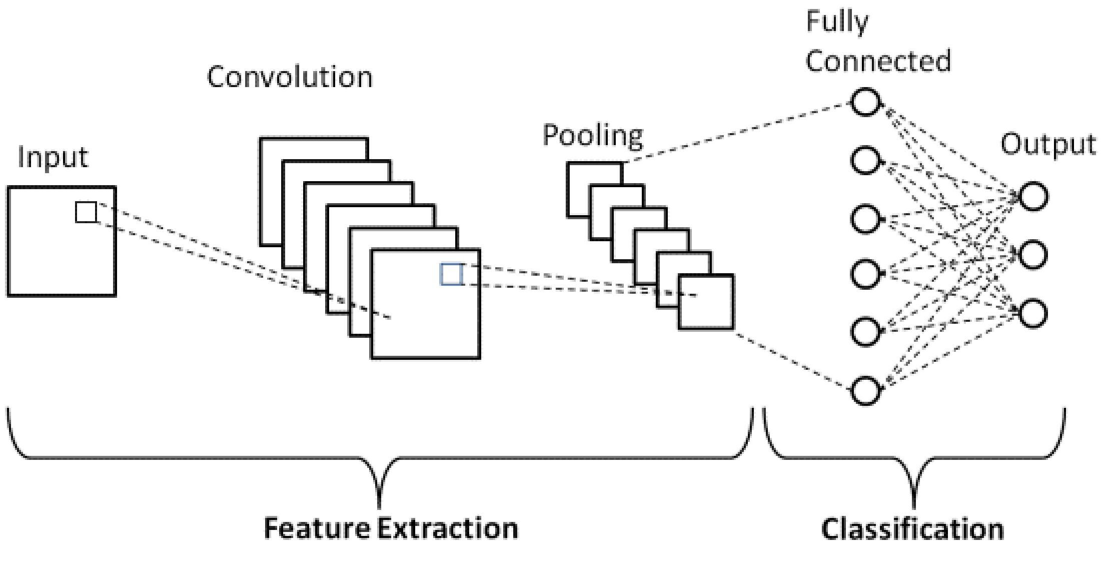


Figure 1.9 – Convolutional Neural Networks.

a **Convolutional Layer :** composed by a set of filters that minimize the complexity of calculation :

1. Filters : or convolution kernels. They are formed from the weights that connect the receptor field to a neuron,those weights are chosen according to the features to be extracted

from the input image.
We do not need to manually define the filters because the Convolutional layer will learn to choose the most suitable for its task through training[21],Figure 1.10 shows [21] the feature map (right) resulting from convolution using a filter that looks for horizontal and feature map(left) resulting from convolution using vertical filter. We denote it with a letter f.

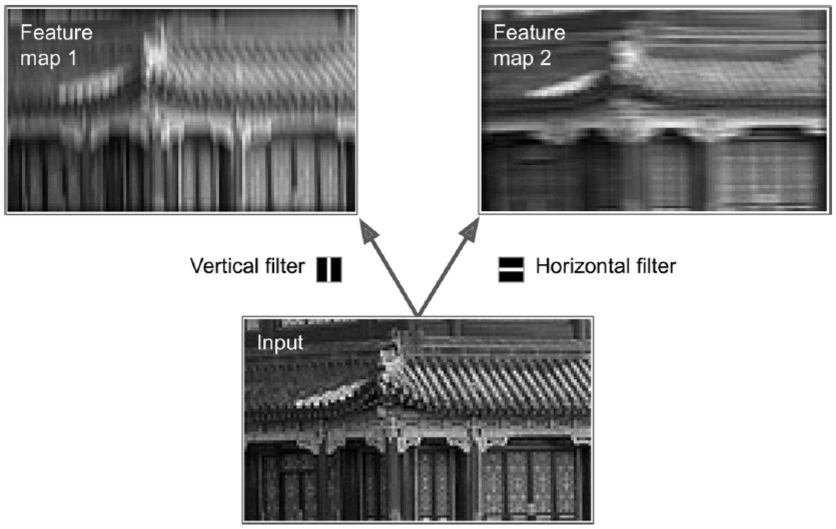


FIGURE 1.10 – The effect of a convolution using a filter that enhances horizontal edges and vertical edges.

2. Padding : It is considered an important process that is used a lot in examining images. For better understanding, we present the following example : if we have a picture with the dimensions [6×6] and a filter of [3×3]. The result of that convolution will be [4× 4] and if the process is repeated, the dimensions get minimized. The image shrinks until it disappears at some level. As a result, the operation cannot be carried out. Add to that, the convolution makes no use of the outside edges of the input image and to solve the two problems together we fill the edges of the image with additional pixels and the matrix [6×6] is surrounded by a full line of numbers increasing the number of columns and rows by 2. Often, the value 0 is placed in the empty cells, and this solves the two problems, We denote it with a letter P, P= n-f+1.

3. Same Convolution : In order for a feature map to have the same height and width as the input layer, And therefore it is necessary to use the Padding.
As a result, the matrix will be of dimensions n+2P × n+2P and the filter will be f × f
n+2P× n+2P × f × f =n+2P-f+1 × n+2P-f+1
so :

$$n+2P-f+1= n$$

$$2P-f+1 = 0$$
$$P=1/2\ (f-1)$$

4. Valid Convolution : Using a filter of an odd size causes the feature map to be smaller then the input image (the padding is not used).
So if the original matrix dimensions are n$\times$ n and the filter f $\times$ f the result is f $\times$ f $\times$ n $\times$ n ...... n - f +1 $\times$ n - f +1.

5. Strides Convolution : Strides is the distance between two receptive fields of two consecutive neurons in a Convolutional layer. In order to make the connections between two layers more sparse, we could use larger strides, We denote it S.
If we convolve an n $\times$ n image padded with P pixels using an f $\times$ f filter and with a stride of S the resulting feature map will be of height $\lfloor (n+2P-f)/S]$ and of width $\lfloor (n+2P-f)/S]$ same for length and width.
As we show the Fig. 1.11 ([21]), the difference between a stride of one and a stride of two.

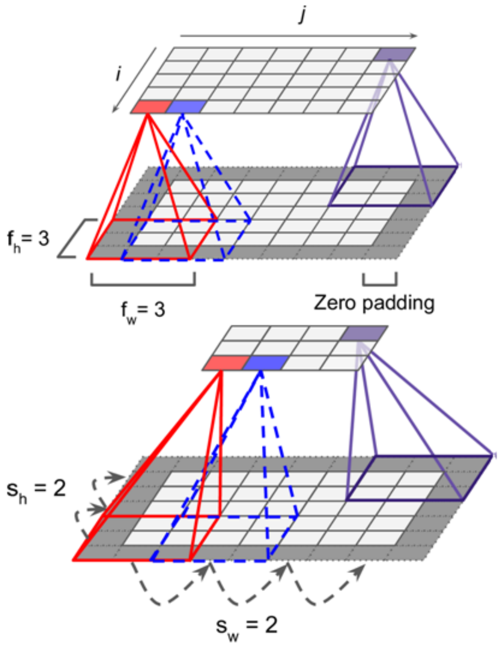


FIGURE 1.11 – The difference between a stride of 1 and a stride of 2.

b **Pooling Layer :** A simple operation it is the least used, but the easiest and the simplest, as it reduces data and speeds up the process (Minimizes the computational load, the memory usage, and the number of the parameters) Thus, it reduces the risk of overfitting.
Pooling also introduces the location invariance concept to the network which means that the network becomes tolerant to small shifts in pixel values, because the values of the pooled

output do not change if the input values shifted a little. There are many pooling layers types that exist each one uses a different pooling function Figure 1.12([21]).

1. Max Pooling : take the max value of the pixels in selected square.

2. Average Pooling : take the average value of the pixels in selected square.

3. Sum pooling : take the sum value of the pixels in selected square.

4. L2 Norm Pooling Returns the L2 norm of a rectangular neighborhood.

5. Weighted Average Pooling Returns the weighted average based on the distance from the central pixel.

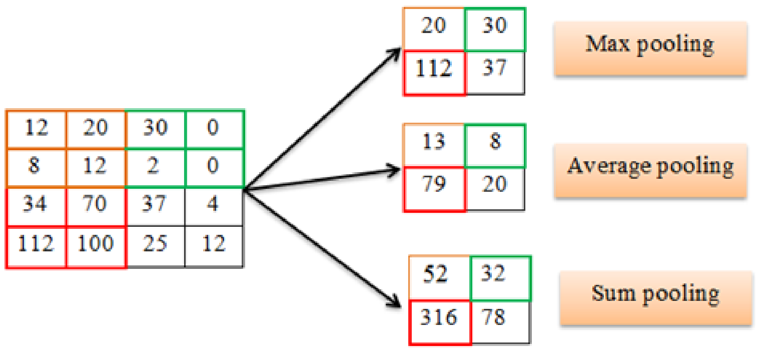


FIGURE 1.12 – Pooling functions types.

c **Flattening Layer :** The pooling layer output feature maps are usually flattened and converted to a single-dimensional (1D) number list (or vector). The final vector is constructed by recovering the pixels of images line by line. The flattening Layer is considered as the Last step in the feature extraction phase.

d **Fully Connected Layer** : The output from the final flatting layer is an input to the fully connected layer. Each fully connected layer is followed by a nonlinear function, such as softmax. The Fully Connected Layers form the last few layers in the network. Fully connected means that every neuron in the next layer is connected to every individual neuron in the previous layer.

Convolutions Over Volumes Usually, the input images are represented in gray in one channel, but if the input images are in color, they are represented by three channels, each channel in red, green, blue, respectively (RGB), and there are also some images that represent more than three channels, such as satellite images.
In general, CNNs use 3D filters to convolve RGB input images and produce one layered feature map, The filters must have the same depth as the input image. The depth of the output volume must be equivalent to the number of filters used, and that volume can be the input to another convolutional layer for extracting even higher-level features.
The Figure 1.13 shows how CNN works in this situation.

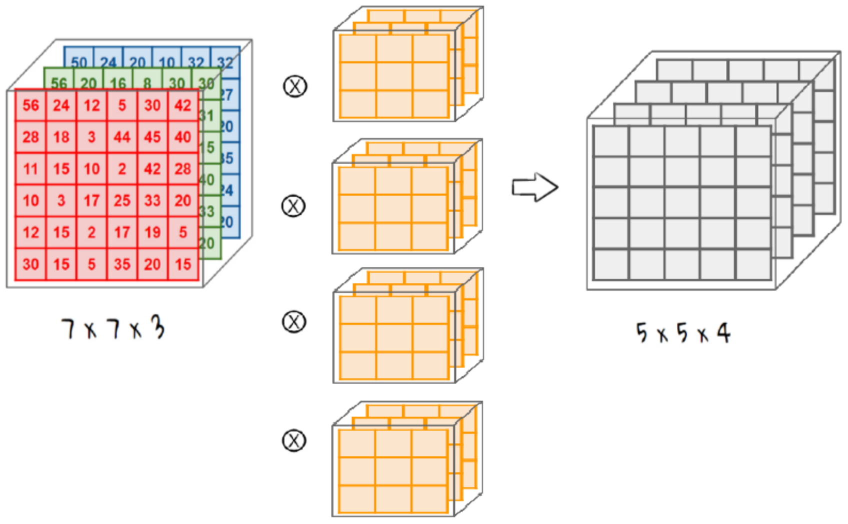


FIGURE 1.13 – Convolutions Over Volumes.

Convolutional Neural Network State of the Art Architectures

LeNet-5

LeNet-5 is the first and most popular CNN architectural design, it was developed in 1998 by Yann LeCun and used widely for the identification of handwritten digits (MNIST). LeNet-5 consists of 7 layers, all of which have trainable parameters(weights), so that they are Three convolutional layers, two pooling layers followed by two fully connected layers[46]. For more information about layers are given by table 1.1. Figure 1.14 represents the Architecture of LeNet-5.

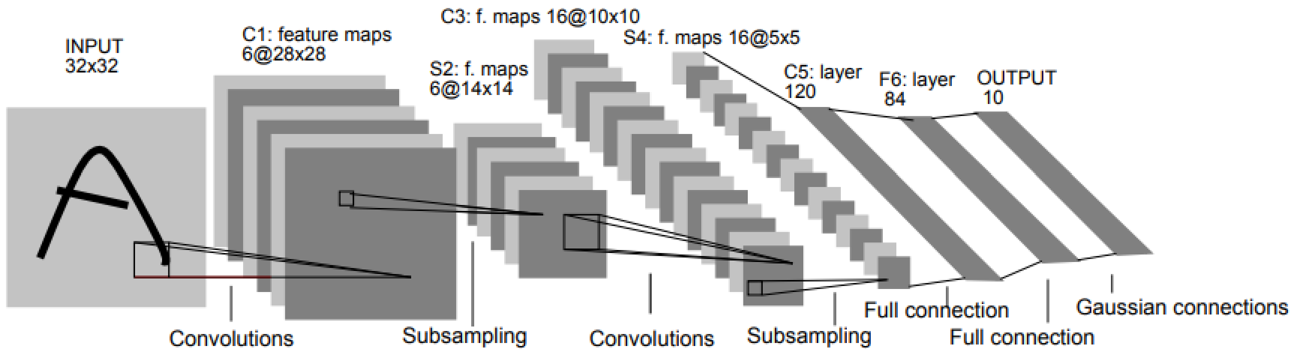


FIGURE 1.14 – The architecture of LeNet-5.

TABLE 1.1 – Le Net-5 architecture

Layer	Type	map	size	Kernel Size	stride	Activation
out	Fully connected	-	10	-	-	RBF
F6	Fully connected	-	84	-	-	TanH
C5	Convolution	120	$1n\times1$	5×5	1	TanH
S4	Avg pooling	16	5×5	2×2	2	TanH
C3	Convolution	16	10×10	5×5	1	TanH
C2	Avg pooling	6	14×14	2×2	2	TanH
C1	Convoulution	6	28×28	5×5	1	TanH
In	Input	1	32×32	-	-	TanH

AlexNet

AlexNet It was developed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton [42]. The AlexNet CNN architecture is quite similar to that of LeNet-5 except that it is only much larger and deeper, and was the first to stack convolutional layers directly on top of each other, rather than stacking an assembly layer on top of each convolutional layer,More details on AlexNet layers are given in Table 1.2 [21] and Figure 1.15([42]) shows their architecture.

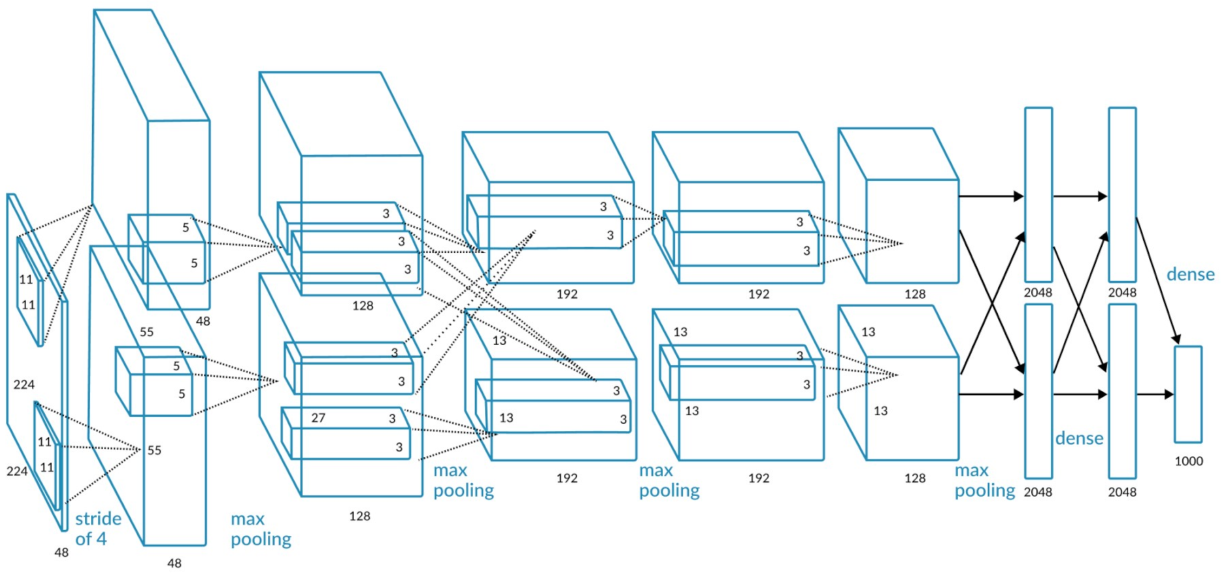


FIGURE 1.15 – The architecture of AlexNet.

TABLE 1.2 – AlexNet architecture

Layer	Type	map	size	Kernel Size	stride	Padding	Activation
out	Fully connected	-	1,000	-	-		Softmax
F10	Fully connected	-	4,096	-	-	-	ReLu
F9	Fully connected	-	4,096	-	-	-	ReLu
S8	Max Pooling	256	6×6	3×3	2	valid	-
C7	Convolution	256	13×13	3×3	1	same	ReLu
C6	Convolution	384	13×13	3×3	1	same	ReLu
C5	Convoulution	384	13×13	3×3	1	same	Rlue
S4	Max pooling	256	13×13	3×3	2	valid	-
C3	Convolution	256	27×27	5×5	1	same	Rlue
S2	Max pooling	96	27×27	3×3	2	valid	-
C1	Convolution	96	55×55	11×11	4	vaid	-
In	Input	3(RGB)	227×227	-	-	-	-

VGG-16

VGG16 created by VGG (Visual Engineering Group) in 2014,The VGG- 16 composed of 13 convolution layers and 3 fully connected layers,It also keeps an activation function (ReLu),Figure 1.16 Represent the Architecture of VGG 16

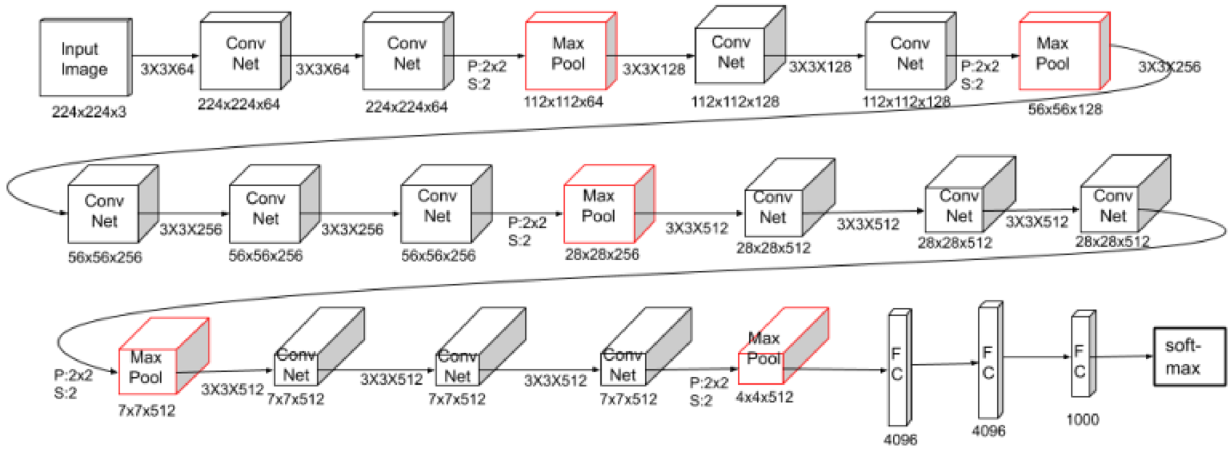


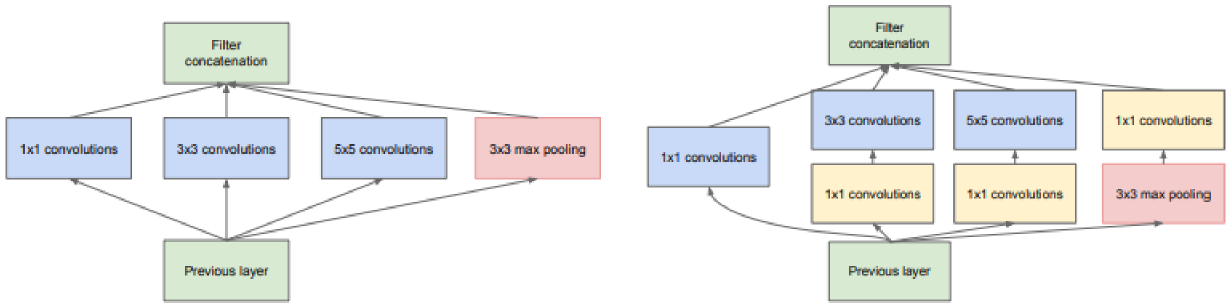
FIGURE 1.16 – The architecture of VGG-16.

GoogLeNet (Inception)

A google research team consisting of Christian Szegedy et al.[74] [21]created the Google Net architecture and won the ILSVRC competition in 2014. and it was the first of the series Inception (Inception V1).

The issue of overfitting could occur if a network with several deep levels is established. The authors of the study paper resolve this dilemma The Google Net design with the concept of making multiple-size filters that work on the same level was further developed by convolutions. The network is really broader than deeper for this idea. The picture below shows a module with a Naive Inception. Figure a 1.17 Where Inputs with three filter sizes [1×1], [3×3], and [5×5]and [5 :5,2] are used for a convolution operation, The convolutions also execute a max-pooling process and are then forwarded to the next inception module.

As neural networks are time-consuming and costly to train, authors restrict the number of input channels to minimize the size of the network and to perform quick computations with additional [1×1] convolutions before [3×3] and [5×5]. Figure b 1.17([74]) arepresents this addition[74]. GoogLeNet is composed of groups of inception modules linearly stacked [74]. See Figure 1.18.



(a) Inception module, naïve version (b) Inception module with dimension reductions

FIGURE 1.17 – Inception module.

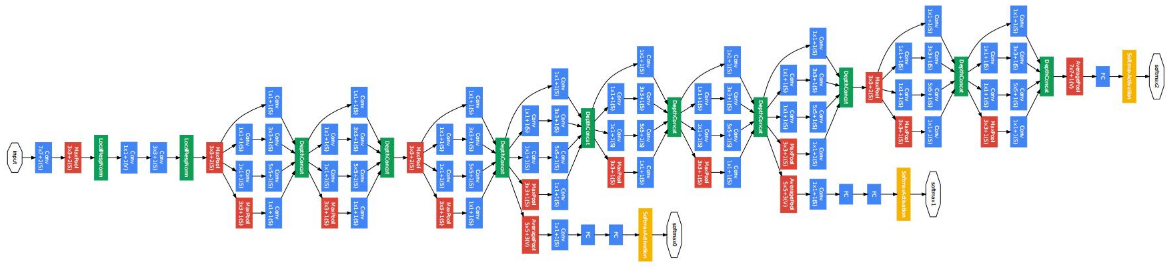


FIGURE 1.18 – The architecture of GoogLeNet (Inception).

ResNets

Residual Networks (ResNet) were developed by Kaiming He et al, in 2015. ResNets won the ILSVRC 2015 challenge [21] in image classification, detection, as well as the Microsoft Common Objects in Context (MS COCO) 2015 detection,segmentation, and localization [27][28], Figure 1.19 Represent the Architecture of ResNets.

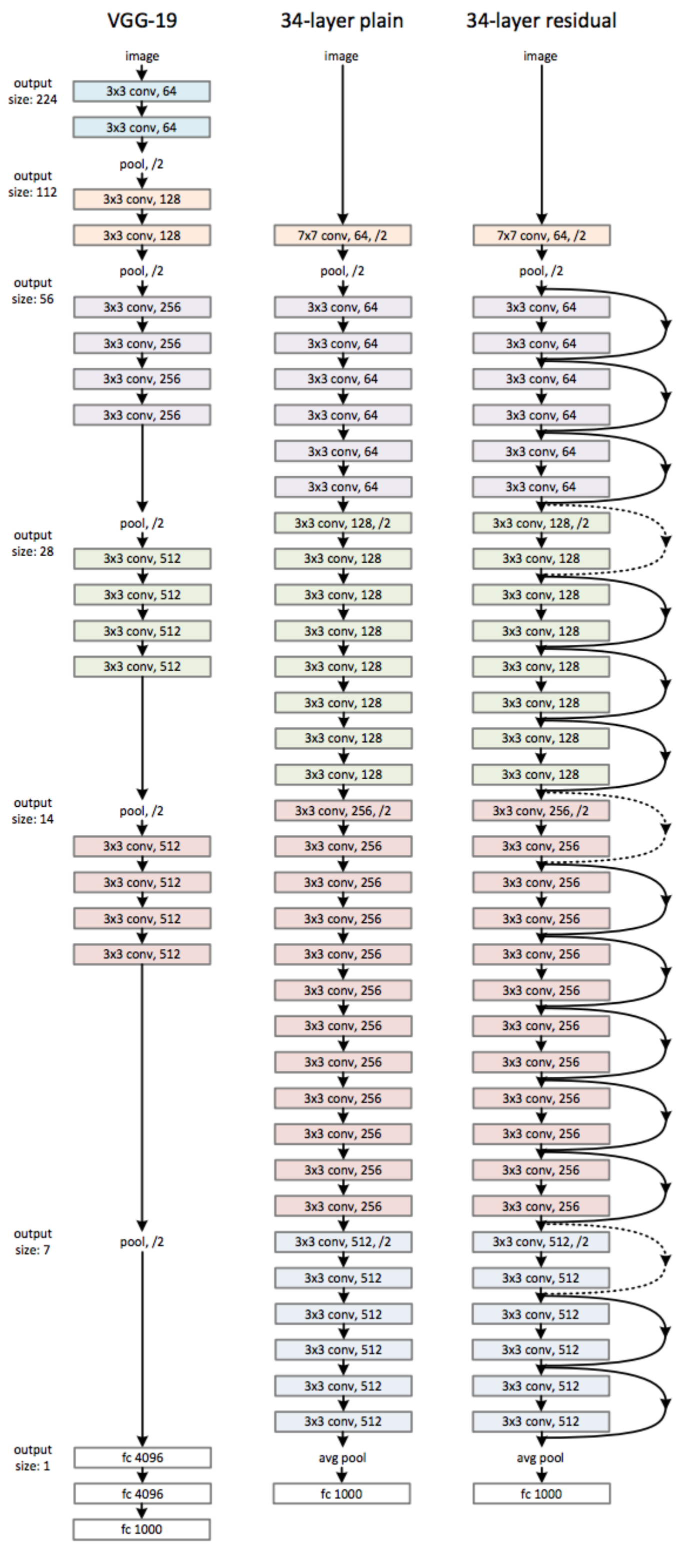


FIGURE 1.19 – The architecteur of ResNet.

Training a Convolutional Neural Network

Network training is the procedure of searching for the optimal set of weights for the network. Ideally, the network searches for the set of weights that leads to a correct prediction whenever a new data instance is passed through.
All FFNNs including CNNs follow the same training procedure. They start by initializing the weights of the network randomly. At first, the network starts with bad accuracy and it tries to enhance it. Data instances are forward propagated through the network one by one and predictions are made. Then, these predictions are compared to the labels to compute a loss. A higher loss indicates bad accuracy. Of course, our goal is to decrease the loss to a minimum value by the end of training. To do that, the weights get altered starting by the output layer and going back in the network. However, instead of editing the weights randomly, they are modified following a specific optimization algorithm, also known as an optimizer. This works because the problem of training a network is equivalent to the problem of minimizing the loss function with respect to the weights. The weights that minimize the loss function are the same as those that give the right predictions.

Recurrent Neural Networks (RNN)

Recurrent Neural Networks or RNNs were proposed by John [35]. They are a type of neural network specialized with sequential data [17]. In the context of sequential data, RNN can be very powerful for the prediction task compared to other networks because it uses feedback loops in processing data. In language processing, when we want to read a sentence, it must be read word by word. It is necessary to store information from the the first word to last, the next word takes into consideration the previous words to complete the meaning. RNN takes its formalism from this mechanism.
RNNs take as input a sequence $x = (x_1, x_2, x_3, \dots, x_T)$ and defines the sequence of hidden states $a = (a_1, a_2, a_3, \dots, a_T)$ to produce a sequences of outputs $y = (y_1, y_2, y_3, \dots, y_T)$ in time $t = T$, where T is total number of inputs, W is weights matrix, $\hat{y}$ is prediction output with an activation function g (usually the TanH)) as equation 1.15 and equation 1.16.

$$a^{<t>} = g(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a) \tag{1.15}$$

$$\hat{y}_{<t>} = g(W_{ya}a^{<t>} + b_y) \tag{1.16}$$

1. **Types of RNNs** We have seen that artificial neural networks, convolution neural networks, ect. are constrained and limited. They allow fixed size vectors in input and output and a fixed number of layers in the model. However, recurrent networks allow variable-length sequences of vectors either input, output, or both. that's what makes it better and more exciting.
   There are many types of RNNs (Figure 1.20), each one is used according to the type of the used data.

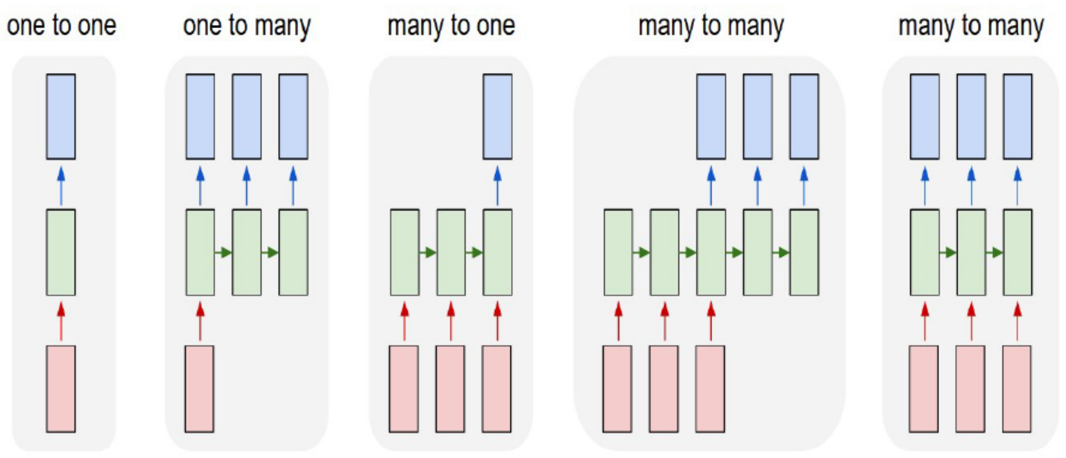


FIGURE 1.20 – Type of RNN.

One to One : One to One are used in the processing of non-sequential data because it loses the interest for recurrence, it does not need timestamps. In this type there is almost no difference between RNN and CNN. It accept a single input for designated to output. This type is applied in image classification application.

One to Many : One to Many type is used to process single input for generating sequence of outputs. For example, create the captions of an image where the input is a picture and the outputs are sequence of words as comments or annotations of the images. This is called image captioning.

Many to One : Many to One handles a sequence of inputs in order to output a single output that is usually used in sentiment classification. For example, such model can take a text as input and output a note or an observation about the sentiment expressed by that text.

Indirect Many to Many : Indirect Many to Many receives a sequences of inputs and produce sequences of outputs when the inputs is finished, which is the most common and recent kind. it is used in machine translation and speech recognition.

Direct Many to Many : Direct Many to Many has the same length of inputs and outputs, that result from each new input, where the current input is built based on the previous. It is used in video classification [32].

2. **Training A Recurrent Neural Network** Training an RNN properly requires data to be compatible with the task at hand. For example, if we are training a many to many model, the output for each time step must be predefined within the data because it is crucial

for computing the loss. The training takes place after unrolling the network through time. According to the back-propagation algorithm, we initialize the weights, apply for the forward-pass and get some predictions. Then, we compute the loss depending on the number of outputs we received. The next step is the backward pass in which we update the weights according to the gradients computed at each time step. The version of back propagation for RNN training is called Back Propagation Through Time.
RNNs gave some problems that we will learn about later, along with the proposed solutions.

3. **Problems of RNN**

   1. Exploding Gradient Problem : Exploding Gradient is the significant increase in the level of gradient during training. It leads to an unstable network in learning, especially in long sequences. This problem can be solved if the gradients are cut or canceled [30]

   2. Vanishing Gradients Problem : In the 1990s this was a big challenge and much more difficult than the exploding gradients to overcome. The gradient values get too small and the model stops learning or takes a long time to find a solution. The vanishing Gradients Problem happened in the case of long-term dependencies because RNN has difficulties in learning it.
   The solution for these two problems will be presented in the next section.

Long Short-Term Memory (LSTM)

Long Short Term Memory (LSTM) is an architecture for RNNs introduced by Sepp Hochreiter and Jürgen Schmidhuber in 1997[32]. it was developed to solve the RNN's training problems (Exploding Gradients and Vanishing Gradients). The short-term problem is known in RNNs as the inability to memorize in the long term. Therefore LSTMs were created for expanding memory and learning over a long time. LSTMs also tackled the exploding and vanishing gradient problems [63] and prevented gradients from excessive increase or decrease.

The basic units in LSTMs are called gates and they are used to control the information flowing over the network, they try to define which information should be updated in the memory cell. The gates in LSTM are the sigmoid activation functions. That means that their output is either 0 or 1. A Sigmoid function is used because we want to give only positive values, we need to keep a particular feature or we need to discard that feature, where 0 means the gate is blocking everything and 1 means the gate is allowing everything to pass through it. The Forgot gate decides which information can be kept for the next steps. The input gate or update gate decides which information must to added in the current step. The output gate decides which next hidden state by which will the information pass.
The math behind the LSTM is given as follows :

We use a cell memory $c^{<t>}$ instead of $a^{<t>}$ then we compute temporally $\tilde{c}^{<t>}$ as follows in equation 1.17 :

$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \tag{1.17}$$

The update gate can be computed as a sigmiod function as shown in the equation 1.18.

$$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u) \tag{1.18}$$

The forgot gate use also a sigmiod function as shown in Equation 1.19 :

$$\Gamma_f = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f) \tag{1.19}$$

The output gate, represented by the equation 1.20 :

$$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o) \tag{1.20}$$

The output of the LSTM unit can be obtained as follows in equation 1.21 :

$$c^{<t>} = \Gamma_u \times \tilde{c}^{<t>} + \Gamma_f \times c^{<t-1>} \tag{1.21}$$

Gated Recurrent Units (GRU)

Gated Recurrent Units (GRU) was proposed by Kyunghyun Cho et al. in 2014 [14],[15], It is considered a simplified version of the LSTM, and also used to solve RNNs problems. Like the LSTM, the GRU has gates, update gate and reset gate are two basic vectors in the GRU that decide which information can be passed to the output. The important thing which is special for them is that they are trained to keep information for a long time, without eliminating the information which is irrelevant to the prediction.
The update gate decides to copy all the information in the past or determine the amount of them and this can eliminate the possibility of happened the vanishing gradient problem.
The module uses the reset gate to determine which information to be forgotten. It's same of update gate but the difference is in the weights of the gates usage.
The math behind the GRU is given as follows : We use a cell memory $c^{<t>}$ instead of $a^{<t>}$ then we compute temporally$\tilde{c}^{<t>}$as follows in equation 1.22 :

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r \times a^{<t-1>}, x^{<t>}] + b_c) \tag{1.22}$$

The update gate can be computed as a sigmiod function as shown in equation 1.23

$$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u) \tag{1.23}$$

The reset uses also a sigmoid function :

$$\Gamma_r = \sigma(W_r[a^{<t-1>}, x^{<t>}] + b_r) \tag{1.24}$$

The output of the GRU unit can be obtained as equation 1.25 and equation 1.26 :

$$c^{<t>} = \Gamma_u \times \tilde{c}^{<t>} + (1 + \Gamma_f) \times c^{<t-1>} \tag{1.25}$$

$$a^{<t>} = c^{<t>} \tag{1.26}$$

1.4 Conclusion

In this chapter, we introduced, machine learning, deep learning and highlighted their relationship, In addition, we have explained deep learning in detail and tried to get acquainted with all its aspects. Our element of concern was Convolutional neural networks because they will be heavily used in our implementation chapter.

Next, we will discuss the main problem which is Object Detection along with it's side tasks, those being : Object detection and counting using the acquired information in this chapter. Moreover, we will summarize the different works and efforts made by the computer vision community to overcome this dilemma.

CHAPITRE 2

OBJECT DETECTION, TRACKING AND COUNTING SYSTEMS

2.1 Introduction

In this chapter, we will define the problem of Object Detection, Tracking and counting. Moreover, we will briefly explain the different approaches followed by researchers in the context of object counting systems.

2.2 Object detection, tracking and counting system pipeline process

This kind of system is processed on tree steps in pipeline, representing by fig 2.1.

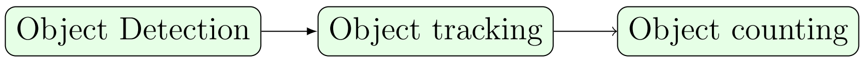


Figure 2.1 – Object detection,tracking and counting system pipeline.

Each step will be discussed in the next sections.

2.2.1 Object Detection

Object detection is the most active field in computer vision (CV) and image processing. It allows us to classify objects inside a given image or video by localizing each object them then drawing a bounding box around it.[90]. In addition, it forms the basis for many other computer vision tasks, for example, image captioning [39] [86], object tracking [38], Object counting [39], instance segmentation [25][26]...etc. In order to perform the task, a set of tools and techniques are relied on. Hence, object detection tools can be divided into methods based on hand-made features and methods based on Convolutional neural networks, and the performance of object detection has been significantly improved through deep learning technology [36].

We mention two approaches to object discovery and the most important works in this field. Object detection methods fall into two classes :

— Handcrafted feature-based methods (before 2014).

— Deep learning-based methods (after 2014).

2.2.2 Object Tracking

Object tracking is one of the most important computer vision techniques, It is typically used for computer vision tasks such as object counting in addition to many other practical applications application for examples robotics, traffic monitoring ,and autonomous vehicle tracking, etc.

Object tracking is divided into two parts : multi-object tracking(MOT) and single-object tracking(SOT), also known as visual object tracking.
multi-object tracking is to identify a group of targets (objects) automatically and then determine the exact path of each object. As for tracking a single object, sometimes also known as visual object tracking, it focuses on tracking only one object in a group of sequential images, even though there are many objects,Where the target to be tracked is determined in the first frame, and then the tracker is assigned to track the target in the rest of the frames.

A lot of work has been done in the field of object tracking and many algorithms have been created. They can be classified into :

— Machine learning –based object tracking Algorithms.

— Deep learning-based object tracking Algorithms.

2.2.3 Object Counting

Object counting [80] is an essential task in computer vision, due to its growing demand for several applications for example in surveillance or traffic monitoring. Object counting is the task of counting the number of object instances in a single image or video sequence. Existing approaches towards the counting problem can be roughly categorised as regression-based approach, density-based approach and detection-based approach.

2.3 State of the art

In this section we mention the most important methods related on our system of detecting, tracking and counting sheep, sorted by chronologically order.

Handcrafted Feature-Based Methods

1. **SIFT :** In 1999, a paper titled Object Recognition from Local Scale-Invariant Features was released by G. lowe[51]. In this work an object recognition system was developed. And it used as a new class of local image features. They are called SIFT features because they are highly constant to changes in size, brightness, and local distortions. Through it, features are detected with high accuracy by adopting a phased filtering approach that defines points Stable in the scale space where image switches are generated that allow local geometric distortions by representing blurred image gradients at multiple scales and at multiple orientation planes. The keys are utilized as inputs to the nearest-neighbor indexing method which determines matches for the candidate object. The final validation of every match is achieved through finding the remaining low least- squares solution to the unknown model parameters. This experiment achieved good results because of its good performance with crowded images and in less then two seconds.

2. **SVM :** In 1999, a paper titled Least Squares Support Vector Machine Classifiers a Large Scale Algorithm has been suggested by J.Suykens et al[73], The hand-made features of object classification and discovery are extracted through the inclusion of the Support Vector Machine (SVM), which has achieved good results, but suffers from the problem of high computational burden and its inability to address large learning problems with many training examples, which led to the suggestion of add-ons and improvements under the name of Least Squares Support Vector Machines (LS-SVM) which worked better for high-volume problems. SVMs and SIFT have been applied in another paper by W.Andrew et al in Sebtember 2016 under the title Automatic individual Holstein Friesian cattle identification via selective local coat pattern matching in RGB-D imagery [5],In This paper an average accuracy of 97% was obtained by the proposed system, Which aims for fully automatic optical identification of individual Holstein Friesian cattle through a dorsal RGB-D image dataset that was compiled by video capture of animals at Windhurst Farm in Langford (UK) using an upper sensor at a height of four meters from the surface of the earth. An image file was extracted and then went through the processing phase to obtain at the end a dataset containing 764 pairs of images from 92 individuals. The proposed system segments animals regions via fitting a depth model then extracts ASIFT descriptors over the detected area, finally SVM creates the predictions.

3. **Cascad Network :** a paper titled Rapid Object Detection using a Boosted Cascade of Simple Features was proposed by Viola and Jones in 2001[77]. In this paper, a new method for visional objects detection based on machine learning is presented under the title Rapid Object Detection using an enhanced series of simple features. It processes images quickly in addition to achieving practical rates in the detection process. The addition that was presented in this work is mentioned in succession, first, the introduction of a new image representation called Integral Image, as it allows features to be calculated very quickly, secondly, the AdaBoost-

based learning algorithm, which chooses a tiny number of visible features from an large set of features and thus results in very effective classifiers. Third, it is a way to combine more complex classifiers and allows you to quickly discard image backgrounds. The MIT + CMU data set was relied on for model training and testing, and an accuracy of 94% was obtained.

4. **HOG :**
   Dalal et al in 2005[16], published an article under the title Histograms of Oriented Gradients for Human Detection where The study of feature sets issued for strong apparent objects recognition was done by introducing a dependence on a new tool and comparing it with its predecessors. They experimentally show that HOG networks significantly out- perform feature sets. for human detection. The paper also highlights the impact of each phase of calculation upon performance, concluding that fine gradient, fine orientation binder, comparably rough spatial binning, and high-quality domestic disparity normalization are all for good results in overlapping descriptor blocks are studied.
   In order to carry out training and test the model, Two different databases were used to test and train the detector. The first dataset is the well-established MIT infantry da- tabase that contains 509 training and 200 test images of pedestrians in cityscapes. This group provides perfect results with the best detection devices, and for that reason, a new and more challenging data set was created under the name ”INRIA”, which includes 1805 images, which were collected by cropping them from a set of personal images,and the results obtained with HOG were 89 % at 10-4 FPPW(False Positives Per Window).

5. **SPM :** Lazebni et al in 2006 [44], published a paper titled ”Beyond Bags of Features : Spatial Pyramid Matching for Recognizing Natural Scene Categories”. Where it is a manner (way , method) for identifying categories based on approximate global geometric correspondence is presented in this work. This approach divides timage till ever finer sub-regions and calculates histograms of domestic characteristics discovered in everysubregion. The resulting ”spatial pyramid” is an easy and calculate-efficient adaptation of an orderless bag of feature representation of an image and demonstrates considerable performance improvements in demanding scene classification tasks. In particular, their suggested technique exceeds the cutting edge of the Caltech-101 database and provides a high level of precision on a huge database of 15 types of natural scenes. The data sets used for training and testing the detector are Consisting ofthree diverse datasets different :
   1. fifteen scene categories. The dataset consists of fifteen viewer categories each containing 200 to 400 images with an average image size of 300 x 250 pixels.
   2. Caltech-101. This data set consists of 31 to 800 images for each category. The majority of the images are medium-resolution around 300 x 300 pixels. It was considered the most diverse at the time.
   3. Graz. This data set consists of two categories of objects, we mention them as follows : the bicycle category with about 373 images, the people category with about 460 images, in addition to the background category, which consists of 270 images, where the resolution of

each image is about 640 x 480. This data set is characterized by high contrast within the category.

6. **DPM :** In 2008, researchers Felzenszwalb et al, proposed a part-based deformation model (DPM)[61]. This model provides a high degree of stability, and in order to increase the accuracy of this model, they increased the number of components and parts of the mixture to obtain a more realistic model, but because of the limited training data, the objective was not reached. This work inspired another model proposed by Patrick Ott et al [61] in 2011 in a paper titled Shared Parts for Deformable Part-based Models. It allowed the sharing of object-part models between multiple mixture components as well as object classes. This results in more compact models and allows sharing of training examples by multiple components, which mitigates the effect of a limited-sized training set. The model was trained on the PASCAL VOC2006/2010 data set, where they obtained the following results where AP = (62.7% to 67.0%) for a VOC2006 dataset.
7. **HOG-LPB :** Wang et al in 2009 [81],a paper titled An HOG LBP Human Detector with Partial Occlusion Handling where A human detection approach has been proposed capable of dealing with partial occlusion by combining Histograms of Oriented Gradients (HOG) and local binary patterns (LBP). It relied on two types of detectors, the first is a global detector for full scanning windows, and the second is a part detector for local areas of training data using a linear SVM. They create an occlusion probability map for each ambiguous scanning window by applying the answer of each HOG block to the global detector. The map of the probability of occlusion is then separated by the mean shift methodology. The segmented section of the window with the most negative answers can be described as an occluded area. When the partial occlusion in specific scanning windows is indicated with high probability, part detectors are used in uncontrolled zones for final classification on the present scanning window. Using the increased HOG-LBP feature, in addition to the global-part occlusion handling method. This work relied on the original INRIA dataset, in addition to a synthesized which occlusion data constructed from the INRIA and Pascal dataset and, The following results were obtained with a detection rate of 91.3% , 94.7% on the INRIA dataset and in order to validate the method for treating global segment blockage Using the composite occlusion data generated from the INRIA and Pascal dataset, it achieved good results but theirs detector cannot handle the articular deformity of people.
8. **Improved Ficher Kernels :** Perronnin, Sanchez and Mensink in 2010[64], Came up with some modifications on the original framework which were actually good. The main advantage is that these results are obtained using only SIFT descriptors and inexpensive linear classifiers. The results are : On PASCAL VOC 2007 the Average Precision (AP) from 47.9% to 58.3%. f- On CalTech 256dataset the Average Precision (ntrain=15, 34.7 (0.2))(ntrain=30, 40.8 (0.1))(ntrain=45, 45.0 (0.2))(ntrain=60, 47.9 (0.4)). In addition, an application was made to the data collection Flickr groups and ImageNet training the classifiers and comparing the results obtained. It showed that the classifiers learned on the Flickr groups perform

remarkably well.

9. **Selective search :** Van de Sande et al in 2011 [76], In this work, a selective search algorithm is presented to solve some problems that the ascending aggregation algorithm cannot solve. The idea is to use a variety of complementary and sequential aggregation strategies. This makes the selective search stable, robust and independent of the object class. This algorithm was applied to the Pascal VOC 2012 dataset with good results with Recall =99% Mean Average Best Overlap of 0.879 at 10,097 locations.

With the potential of Convolutional Neural Networks, object detection took a whole new turn and more powerful modern models were developed. We will learn about some of these models in the next section.

Deep learning-based methods

The modern methods of Deep Object detection can be split into two groups. Two-stage methods and one-stage methods see Figure 2.2[36].

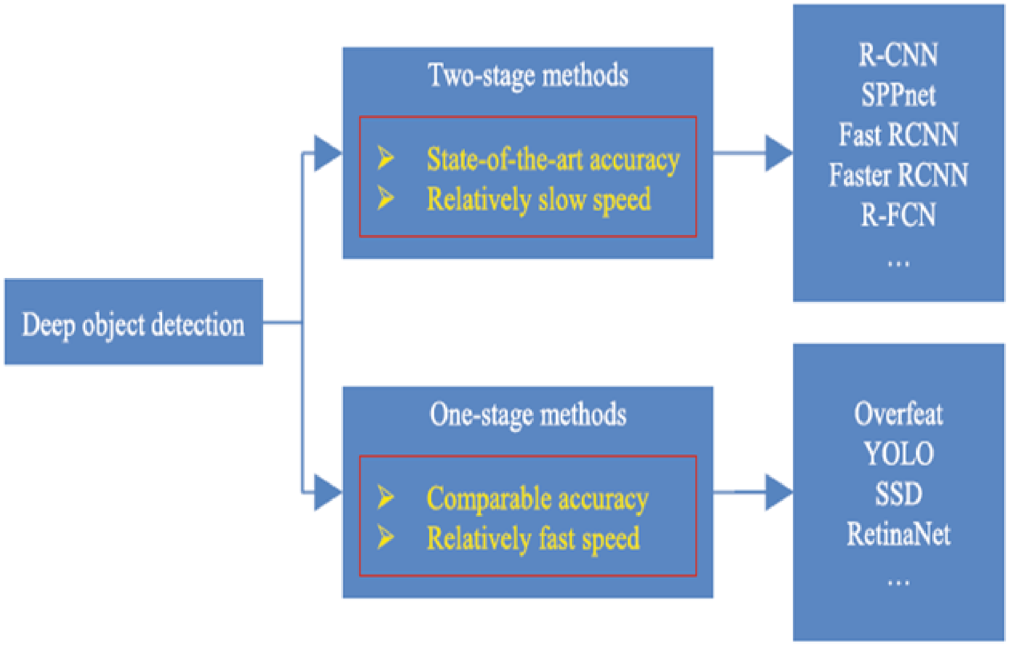


FIGURE 2.2 – Deep learning-basd methods.

CNN based One-stage Detectors One-stage approaches strive to predict object type and position at the same time. One-stage methods have a much higher detection speed and similar detection precision , Among these methods, we mention the following :

1. **YOLO** You Only Look Once (YOLO),It has several publications, we mention them and mention their most important applications, respectively :

   YOLOv1 :
   It is the first version in YOLO proposed in 2016 by J. Redmon et al[65], The model was trained and the test was conducted on the PASCAL VOC 2007 dataset and it was obtained that YOLO pushes mAP to 63.4% while still maintaining real-time performance. In addition, the basic YOLO model processes the image at a rate of 45 frames per second with real-time preservation, while Fast YOLO processes 145 frames per second.

   YOLOv2 :
   YOLO v2 It is an upgraded version of YOLOV1 released in 2017 by J.Redmon et al[66],The model was trained and the test was conducted on the COCO and PASCAL VOC data sets, Training the model they get 76.8 mAP at 67 frames per second and they get 78.6 mAP at 40 frames per second on VOC 2007.

   Wen Shao et al in 2019 introduced a paper titled Cattle detection and counting in drone images based on convolution neural networks [71] , where the YOLOv2 algorithm was used to detect and count livestock. Throughout this work, a livestock detecting and counting system that adopts drone images was suggested. Two databases captured by a drone were relied to form the first data set consisting of 656 images that were visualized at a height of 50 meters, while the second set consists of 14 images where the two databases were collected from different places in different weather conditions. The drone image is characterized by such a feature as allowing the targets to seem to have one size when the flight height is fixed. Therefore, they resized the training images to the resolution where the size of cattle is equal to the size of one cell in the output feature map. The input resolution optimization helped to better for the detection performance as they got a precision of 0,957, a recall of 0,946, and a 0,952 F- measure. Figure 2.3[71] represents the results obtained. Concerning counting cattle in a vast area of 1,5 to 5 hectares, when all cattle move less than their body length during the flight section, results close to the true value were obtained. However, they noticed a decrease in detection performance when applying the system to dataset 2.

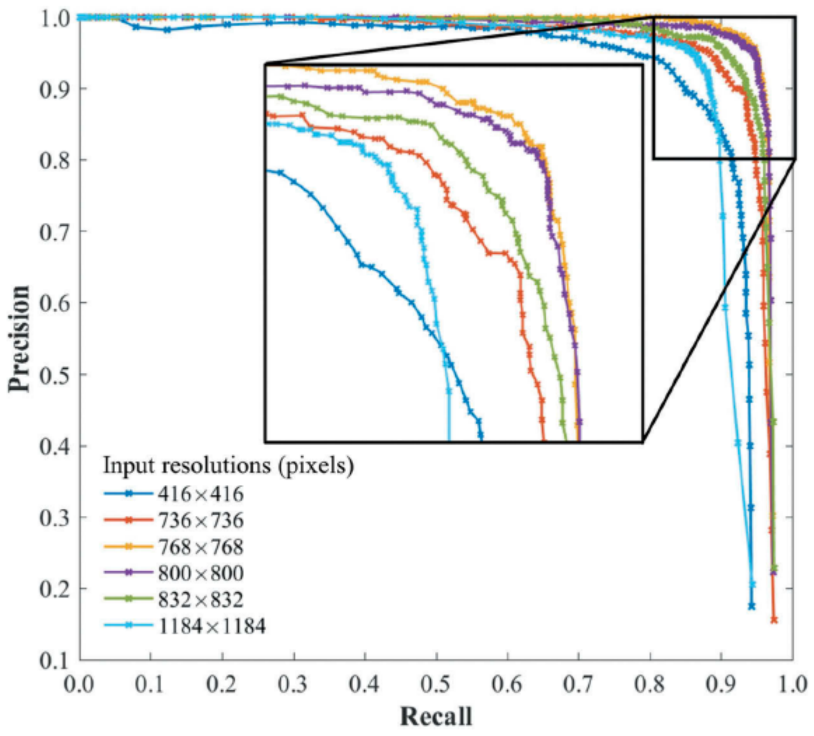


Figure 2.3 – Plot of the prcision-recall cattle dedection.

We briefly mention how the proposed system that was adopted works, and it is represented in Figure 2.4[71]. The system inputs is an image from the data set, where first the cattle is detected in each image, then they do the second step, during which the three-dimensional surface of the pastures is reconstructed by SfM from the same set of images, and the last step is done at its level by integrating the detection results for each frame, through This merger and with the passage of time the problem of double detection of cattle is eliminated.

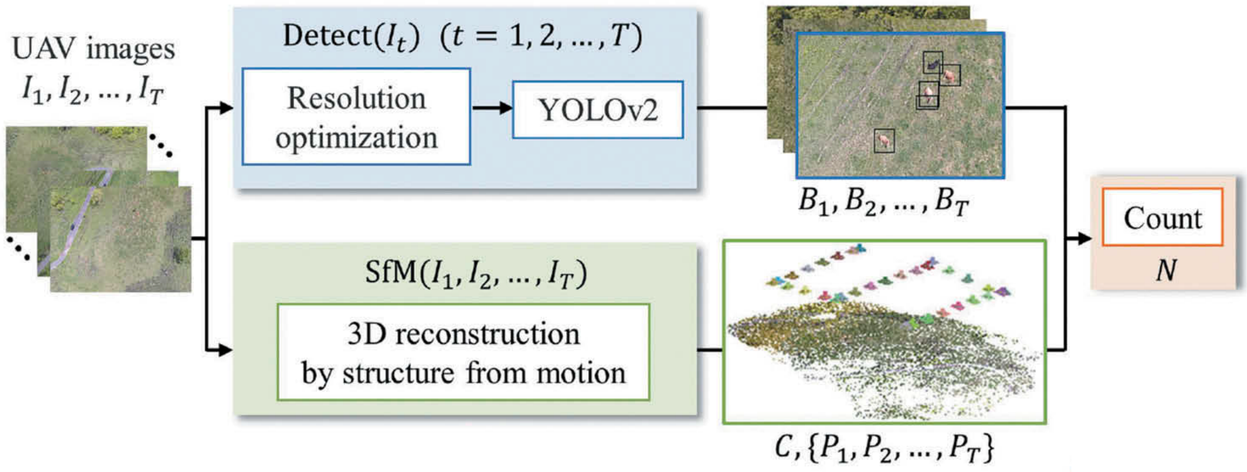

UAV images
$I_1, I_2, \ldots, I_T$
Detect(I_t) ($t = 1, 2, \ldots, T$)
Resolution optimization
YOLOv2
$B_1, B_2, \ldots, B_T$
SfM($I_1, I_2, \ldots, I_T$)
3D reconstruction by structure from motion
$C, \{P_1, P_2, \ldots, P_T\}$
Count
N

Figure 2.4 – System proposed to detect and count the cattle.

In the detection task, the YOLOv2 algorithm is used to improve accuracy, and the detection results are used in the cattle counting process. More details of the counting process can be seen represented in Figure 2.5 [71].

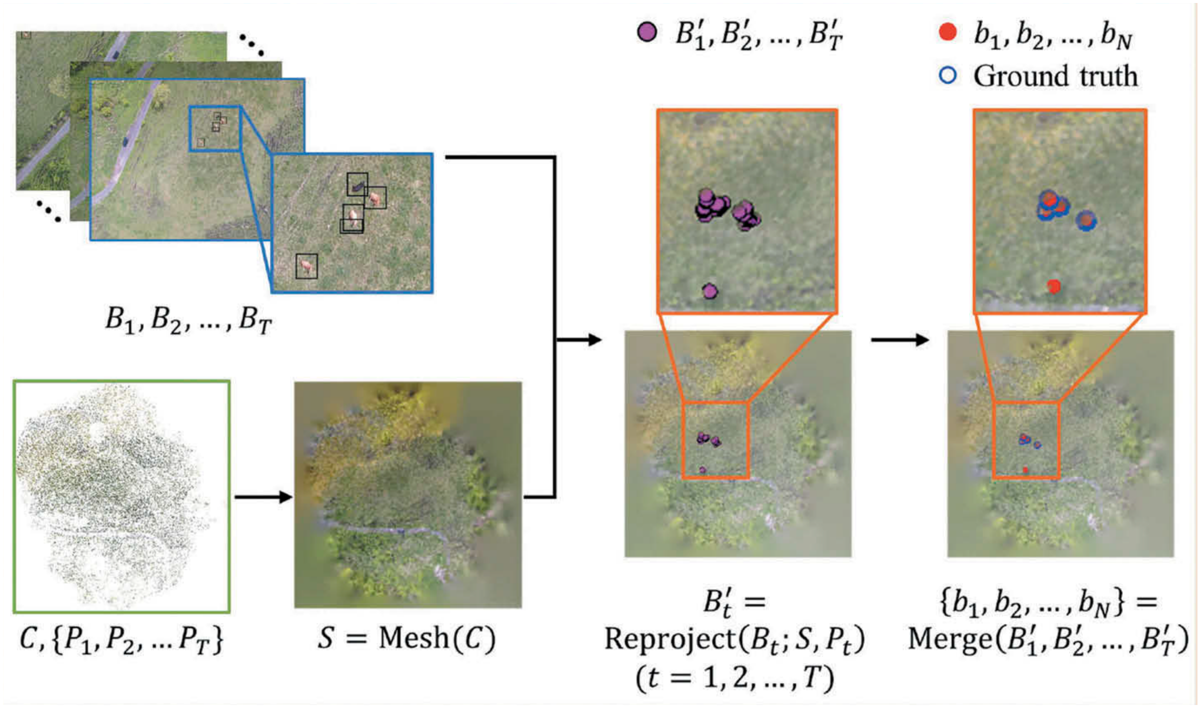

● $B'_1, B'_2, \ldots, B'_T$
● $b_1, b_2, \ldots, b_N$
○ Ground truth
$B_1, B_2, \ldots, B_T$
$C, \{P_1, P_2, \ldots P_T\}$
$S = \text{Mesh}(C)$
$B'_t = \text{Reproject}(B_t; S, P_t)$
$(t = 1, 2, \ldots, T)$
$\{b_1, b_2, \ldots, b_N\} = \text{Merge}(B'_1, B'_2, \ldots, B'_T)$

FIGURE 2.5 – Details of cattle counting system.

YOLOv3 :
YOLOV3 [67]is a new release in which updates are made to YOLOV2(YOLO9000) by J.Redmon et al in the paper entitled YOLOv3 An Incremental Improvement ,YOLOV3 can predict boxes for eighty different categories, At a rate of twenty-nine frames per second with 31.0 mAP.

YOLOV4 :
YOLOV4 It is an upgraded version of YOLOV3 presented by A. Bochkovskiy et al , in a paper titled YOLOv4 : Optimal Speed and Accuracy of Object Detection in 2020. YOLOV4 has achieved a great performance on the COCO data set where 43.5% AP at a real-time speed of 65 Frames Per Second (FPS). This method has been used in several modern related works. some of them are mentioned briefly in the next paragraphs.

Pooja Mahto et al published a paper in May 2020 under the title Refining YOLOV4 for vehicle detection [54], in which they talked about the actual vehicle detection technology that is widely used in the applications of traffic monitoring cameras or for self-driving cars. In this research, the team used a tuned in YOLOV4 algorithm which was suitable for vehicle detection. A publicly available academic dataset called (UA-DETRAC) was used in this work and it consisted of a hundred video sequences filmed with a resolution of 960 x 540 and at a speed of 25 frames per second, and it is divided into training and testing groups, and the pictures were taken in different places, but similar, in different conditions, day and night, and in sunny, cloudy and moderate weather. To implement the YOLOv4 algorithm on the data set, the hyper-parameters are first set with initial values and then they change with time until they are proven when they give them the desired results. Finally, the results were obtained as follows. This model presented results with an mAP of 67.7%, which exceeds the initial model by 10%. It also showed a tremendous speed in its detection ability, in addition to being able to detect 38 frames per second.

Xingda Sun et al in 2021,proposed a paper en titled Research on the Application of YOLOv4 in Power Inspection [72], where Due to the increasing demand for electric power in recent years, and as it constitutes of great importance for its multiple uses, smart systems must be used to contribute to facilitating some tasks such as inspections. In fact, unmanned aircraft were used to solve the problem of manual inspection. The researchers chose the YOLOv4 algorithm for target detection through images, where the training dataset is categorized and then the data is trained by the Darknet. The results obtained are somewhat acceptable as they obtained the Precision is 0.92 and the Recall is 0.904 after training they are on a special data set collected.

In light of the Corona pandemic, social distancing (leaving certain distances between people) is one of the most important prevention methods to avoid the spread of this virus, which has caused great material and human losses worldwide. In order to help calculate the distance that must be maintained between people in public places or work places, Researchers J. Li et al presented a paper titled The Application of YOLOv4 and A new pedestrian clustering algorithm to implement social distance monitoring during the COVID-19 pandemic [48]. In this paper, the latest YOLOv4 object detection technology was used, in addition to the Simple Online and Realtime Tracking with a Deep Association Metric (Deep SORT) multi-object tracking algorithm to detect and name pedestrians, as well as the new pedestrian grouping algorithm. Through the experiment, it was found that the social distance in the scene can be monitored in real-time. Strictly, there is a correlation between the average distance of pedestrian gatherings, the density of pedestrian gatherings, and the infection index, which can be used to assess the safety of places and prevent the spread of disease. In order to prove the effectiveness of YOLOv4, it was compared with the YOLOv3 algorithm, where mean average precision(mAP) was used as a measure to measure the accuracy of the algorithms, where YOLOv4 achieved mAP = 0.84, while the YOLOv3 algorithm achieved mAP = 0.77, addition, the larger the map size the smaller the frame rate per second in the model for YOLOv4 compared to YOLOv3. Through the comparison, it was found that YOLOv4 provides better results in real-time social monitoring tasks, and the pedestrian grouping algorithm contributes to the rationality of detecting distances.

2. **SSD**

   Single Shot MultiBox Detector (SSD)[50], Was been introduced by Wei Liu et al in 2016. It is a method of object detection based on the use of a single deep neural network,In order to train and test this model, three data sets were used : PASCAL VOC, COCO, and ILSVRC, where good results were achieved in all cases. SSD achieves 74.3% mAP1 in VOC2007 test at 59 FPS For 300×300 input and SSD achieves 76.9% mAP for 512×512 input.These results are good compared to other modern methods.Also, the detection accuracy obtained is good despite being simple in terms of composition.

CNN based two-stage detector Object detection is treated as a multistage procedure in two-stage methods. First, any potential object suggestions are derived from an input image, second The qualified classifier then further categorizes these proposals into the relevant object categories. This strategies advantages can be outlined as follows :
- It eliminates a significant number of proposals, which are then fed into the classifier that follows. As a consequence, it will speed up the detection process.

-The proposal generation stage may be thought of as a bootstrap technique. The classifier will concentrate on the classification task with little interference from context (or simple negatives) in the training stage, thanks to the suggestions of potential objects.
Among these methods, we mention the following :

1. **R-CNN**

   R-CNN has been used in many object detection applications and has achieved encouraging results. In October 2017 , A work entitled Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning was presented by W.Andrew et al.[4] This paper includes a first-of-its-kind work that applies deep learning to the automatic cattle recognition task. In order to do this work, two automatic databases known as FriesianCattle2017 were used, which were captured from inside the barn, and the second, AerialCattle2017, which is a set of images of cattle captured in the open air. It achieved good results with respect to the data set used, ie, in the case of cattle, their size is small and dispersed in the form, and this is due to the deep learning algorithms that have proven their efficiency. This operation was done manually as the client (drone) was transferred manually and only the data was captured. It was not notified through the identification process, and the video footage was analyzed by LRCN offline.
   The FriesianCattle2017 data set detection and localization performed robustly with an accuracy of 99.3%.
   In November 2018, A work entitled Detecting and Counting Sheep with a Convolution Neural Network was presented by F.Sarwar et al[70], Farmers face great difficulties in detecting and counting sheep, which causes them great losses, and due to the great development that

deep learning algorithms and drone are witnessing, the researchers in this article proposed a solution based Capturing a video with a drone and counting using the R-CNN model. To evaluate its efficiency and performance, it was compared with another completely different method called Expert System. The dataset used was extracted from Video filming from a drone from a sheep and beef farm in Perino, New Zealand, at an altitude of 80 meters, in different weather conditions(sunny, overcast) Both methods achieved fairly good results, and given these results, they have to work more to develop deep learning algorithms for effective detection of objects, especially if the objects are very small compared to the background or overlap between colors, as well as the use of Tracking method to eliminate errors in counting.

2. **Fast R-CNN**
   This work was presented in a paper by Ross Girshick in 2015[22], Through this work, a method of Convolutional neural network based on the fast region for object detection (Fast R-CNN) was proposed, as it improves the speed of training and testing in addition to increasing the quality of detection. This method was trained and tested on three data sets in addition to comparing its accuracy with other methods except They achieved good results outperforming them by a good margin. The PASCAL dataset was used and the following results were obtained VOC 2010 : mAP of 67.2%,Voc 2012 : mAP of 65.7%, VOC 2007 mAP (from 63.1% to 66.9%).

3. **Faster R-CNN**
   Presented by Girshick et all in 2015 in a paper titled Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks[68], In this work, Faster R-CNN is presented, which is a mixture of FastRcnn and RPN, where RPN is a network to generate efficient and accurate area proposals by sharing convolutional features with the final detection network and enables the operation of an object discovery system Unified based on deep learning to operate at near-real-time frame rates, Faster R-CNN has been trained and tested on the several datasets with good results.
   The Faster R-CNN system increases the mAP from 41.5%/21.2% (VGG- 16) to 48.4%/27.2% (ResNet-101) on the COCO dataset.

4. **Mask R-CNN**
   Mask R-CNN Architecture introduced in 2017 by He et al[26], is a Faster R-CNN Extension that provides a predictive model for a mask for each object detected and it is The most recent variant of the family models where It runs at 5 frames per second. trained on COCO dataset, The following results were obtained AP= 37.1, AP50= 60.0, AP75=39,4, APS=16,9,APM= 39,9 and APL=53.5.
   Many researchers have used Mask R-CNN in many applications to detect objects, including : themed paper for Automated cattle counting using Mask R-CNN in quad-copter vision system [87] In this paper, the researchers B. Xu et al,18 Feb 2019 used MaskR-CNN as the latest deep learning algorithm to detecting and counting cattle from quad-copter imagery.

Where the proposed system is represented in Figure2.6[87].

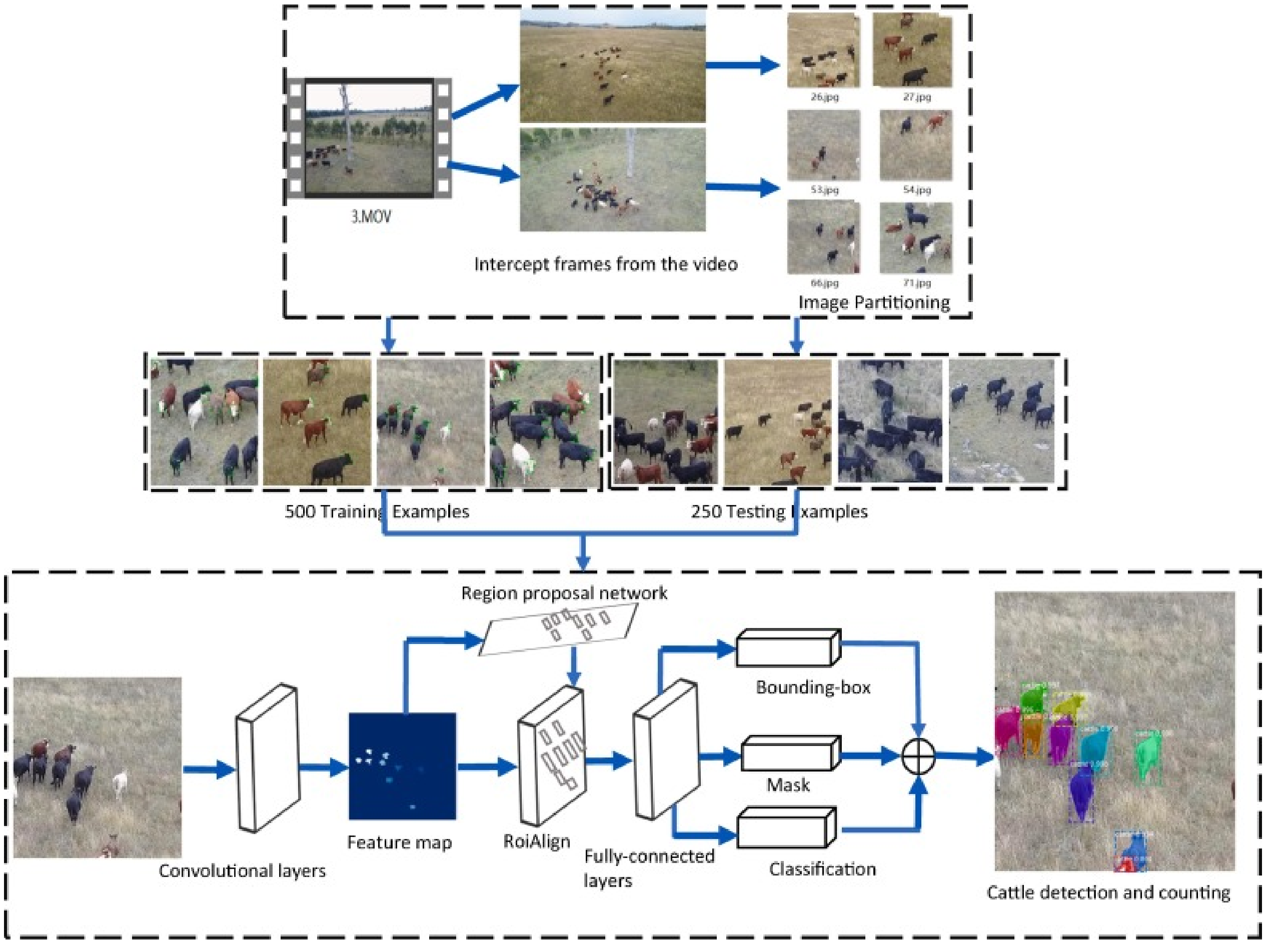


FIGURE 2.6 – The structure of cattle detection and counting algorithm.

The used dataset was compiled by shooting four videos by a drone during 15 rounds around three pastures and a feedlot in different weather conditions and different backgrounds, Intersection over Union was determined after several experiments with a value of IOU = 0.5 as the best value. We can see this in Figure 2.7(The curve on the left), which compares the accuracy and recall for different IoU thresholds for the three detection states. In addition to Figure 2.7[87](The curve on the right), which compares the three different detection states between F1 scores at different IOU thresholds.
To evaluate this method Mask R-CNN was compared with three other methods faster R-CNN , YOLOv3, and SSD. Using the same dataset in all methods : After making the comparisons, Mask R-CNN outperformed the other methods.

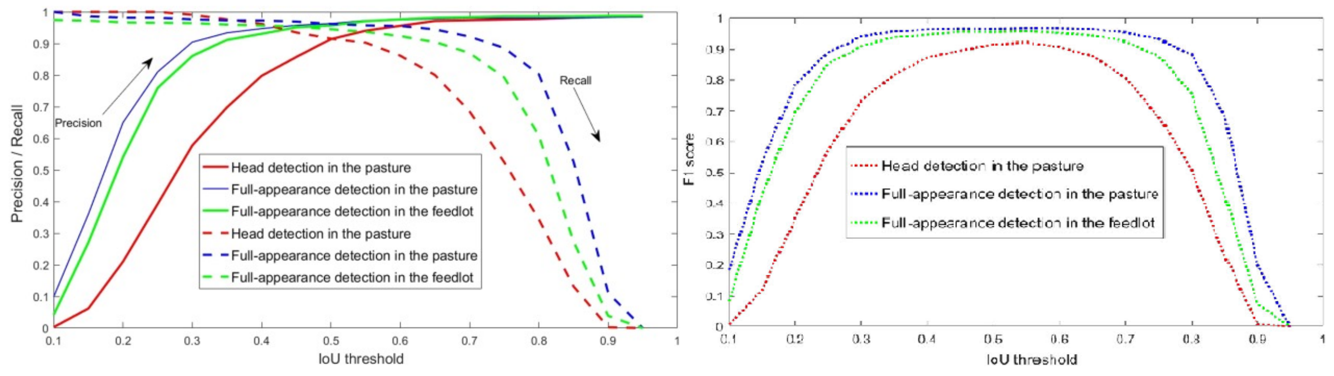


FIGURE 2.7 – The left curve represented comparisons of Precision and recall for different IOUs for the three detection cases. while the right curve represented comparisons of three different detection cases between F1 and different values of IOU.

It was also adopted in a paper entitled Livestock classification and counting in quad-copter aerial images using Mask R-CNN prepared by Beibei Xu et al in 2020[88], Mask R-CNN was used to train images and extract features from images captured by drones. The goal of this work is to build a more appropriate quad-copter vision system for the task. Therefore, the experimental results and comparisons over different IoU thresholds as well as the impact on the number of training examples demonstrate how the proposed system is able to effectively recognize two different categories of livestock. It has achieved an accuracy of 96% for classifying livestock and 92% for counting livestock These results offer great hopes for the development of this field and its use in the field of agriculture and livestock.

Y.Gan et al in 2021 presented an article entitled Object Detection in Remote Sensing Images with Mask RCNN where they used the Mask R-CNN detection algorithm [20]. In order to detect ships and aircrafts through a set of remote sensing images. Where the structure of the Mask R-CNN detector has been modified by adding an FPN module to improve the accuracy of detecting small objects. This is represented by Figure 2.8[20] and example results of detecting airplane and ship on different resolution to them model 2.9. In order to train, evaluate and test a data set consisting of selecting images from three previously available data sets, DOTA, OPTA2017, RSOD was used. After the process of training and testing the proposed model, the following results were obtained with respect to the results of the aerial detection evaluation : Recall : 100.0 ;Precision :100.0 ; F-measure :100.0 Or as for the results of an evaluation of detecting ship at different resolution ,In low resolution they gets :Recall : 56.7,Precision :71.8 ,F-measure :63.As for the high resolution, they get the following result : Recall :86.0,Precision :84.3,F-measure :85.1. Due to the results obtained, it has achieved highly accurate results.

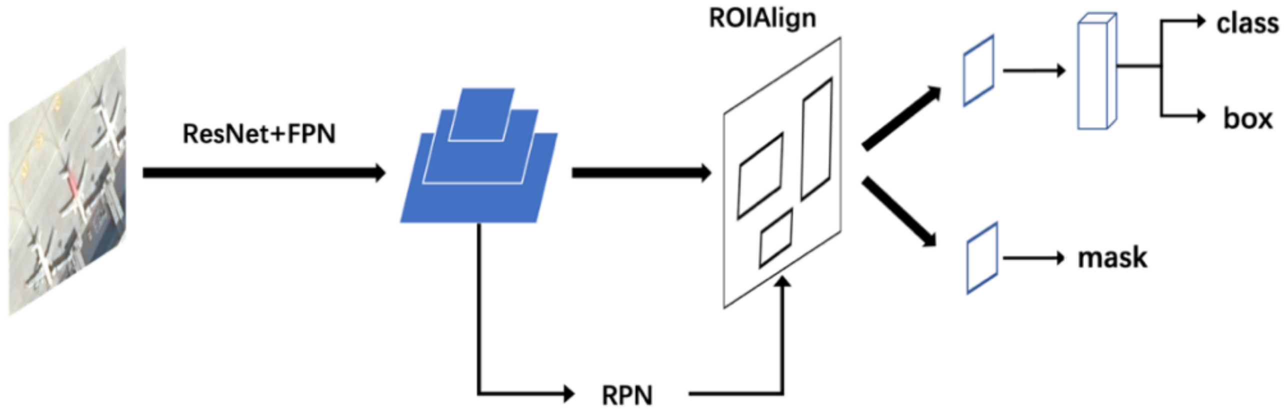


FIGURE 2.8 – The framework which that was used in a detection operation.

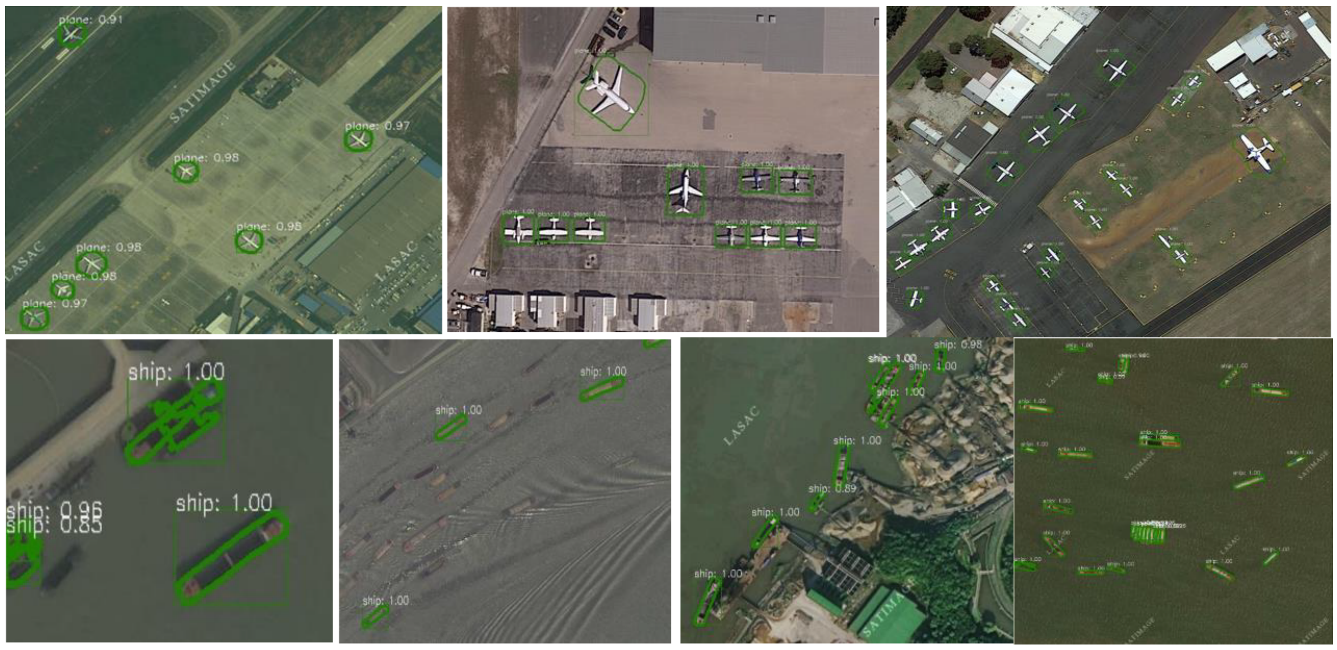


FIGURE 2.9 – Example results of detecting airplane and ship on different resolution.

The task of detection is followed by the task of tracking objects, which is considered among the most difficult tasks, as it takes a long time , since the objects to be tracked are subject to many changes in the shape and appearance in the images, due to distortion, complex background, random and sudden movements of the object.

In object tracking domains, many algorithms were presented, nd they were classified into two fields machine learning-based object tracking Algorithms and deep learning-base object tracking Algorithms.

Machine learning –based object tracking Algorithms

They can be classified into three main categories :

Point tracking Algorithms Point-based tracking involves object detection with several feature points in each frame. If the item is properly detected in an image, these methods have high precision rates. However, in cases of occlusions, they may lead to false object detections. Kalman filters[37] and particle filters are some of the key algorithms in this area.

Kernel Tracking Algorithms Using a kernel that records the movements of an item from one frame to another inside the region of the primary object. The algorithms under this category are classified in accordance with the approach used to track the object. The template matching [62], layering, mean shift[18], and Support Vector Machine (SVM) are some of the most used methods.

Silhouette Tracking Algorithms Tracking algorithms based on silhouette are effective for items that do not employ simple geometrical properties and have complicated forms like fingers and hands. The crude form models of these techniques are used for tracking objects in the previous frame. They are use preset information for each item and handling various complicated shaped objects, occlusion, fusion and issue splitting. Contour and form tracking are the major subcategories. The main advantage of silhouettes is its versatility to accommodate a wide range of object forms. they nonetheless require preliminary information on all items.

Deep learning-based object tracking Algorithm

1. **Detect and Track :**
   This paper[19], addresses one of the most important tasks applied to video, which became increasingly complex with time, the high-accuracy detection and tracking of categories of objects. This work was published under the title Detect to Track and Track to Detect by C. Feichtenhofer et al in 2017. It is based on creating a unified framework for detecting and tracking objects in a video at the same time. The ConvNet structure that was adopted in this work is aimed to carry out both detection and tracking tasks simultaneously. It structure first sets a multitasking target, that is detected by one of the detection methods, Next, correlation features that represent object iterations over time are introduced in order to assist the ConvNet with tracking, then in order to obtain high accuracy video-level target detection so they correlate frame-level detections based on paths across frames. For evaluation, the ImageNet VID dataset is used to obtain results and then compared to the winning method in the ImageNet Challenge. The resulting performance for single-frame testing is 75.8% Mean Average Precision (mAP). Video level methods. raises performance substantially to 79.8%. In Online capabilities and runtime. The performance is 78.7%mAP and Temporally strided testing. The architecture is only evaluated in each frame of the input sequence and the

detected pathways should connect the detections over larger time steps. The performance of time step = 10 is 78.6% mAP which is 1.2% lower than the full-frame evaluation. As for the comparison with the ImageNet challenge winner, the proposed structure outperformed it.

One of the advantages of this method is that failed detections can be recovered using previous frame detections, but it is a computationally expensive algorithm.

Figure 2.10 [19] represents the detection and tracking approach.

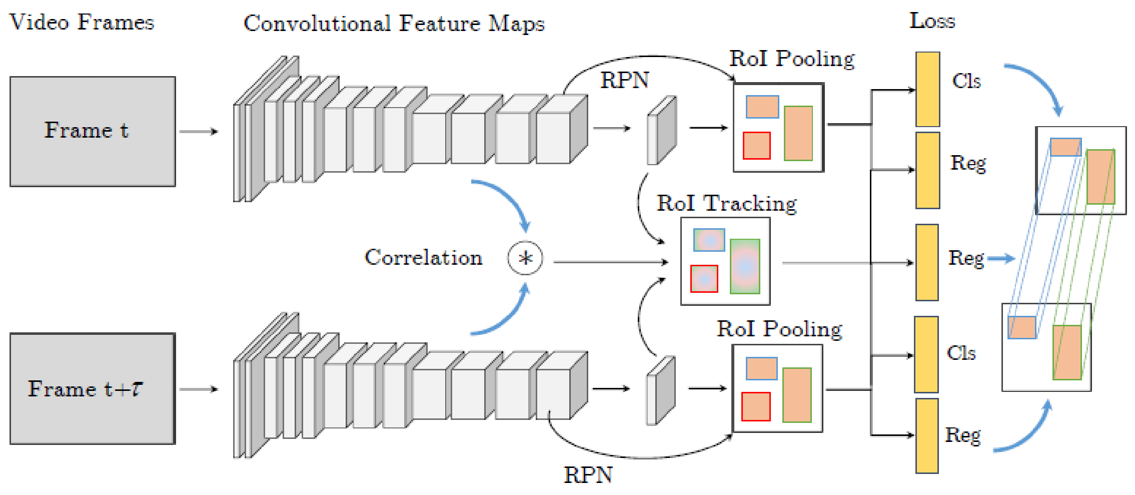


FIGURE 2.10 – Architecture of Detect and Trac approach.

2. **Deep SORT :**

The tracking algorithm Simple Online and Realtim (SORT) was proposed by Bewley et al in 2016, which performed well compared to the tracking algorithms at the time, and then an extension of it called Deep SORT was proposed by N.Wojke et al in 21 Mar 2017.

Deep SORT [83] is an algorithm for tracking-by-detection that incorporates both the detection results box parameters and the appearance information of monitored items for associating the detections with objects previously tracked in a new framework. It is an algorithm for online tracking. That's why it examines information on current and previous frames to anticipate the current frame without having to process the entire video at once.

To evaluate the performance of the Deep Tracking algorithm Deep SORT , the Multiple Object Tracking (MOT)16 has been relied on as the benchmark. The results were obtained : Multi-object tracking accuracy equal 61.4 and Multi-object tracking precision equal 79.1.

Kristina Host et al presented a paper [33]in which the Deep SORT's tracking algorithm was used in order to track the players inside the stadium after they were discovered by the You only look once (YOLO)v3 algorithm. Aiming to analyze a specific sports performance and adopt a specific method. This task is among the most difficult tasks due to the fact that the players move quickly and in different directions and wear the same clothes.

The dataset relied on was collected through a video taken of the men"s handball match. The results obtained were as follows multi-object tracking accuracy equals 99.3, multi-object tracking precision equals 99 ,and Multi-object tracking precision (MOTP)F1 equals 24.7%.

The obtained Multi-object tracking accuracy (MOTA)A and MOTP results are excellent, but they cannot be relied upon for evaluating the algorithm because the same detector was used to discover the ground truth and in the tracking. As for the MOTPF1 scale, a result of 24.7% percent was obtained due to the difficulty of the scenario in addition to a large number of players on the field and their movement quickly, which sometimes leads to their blocking.

3. **Tractor++ :**

   This work [8], Was created by Philipp Bergmann et al in on 17 August 2019, under the name Tractor, This algorithm relies on detection techniques for tracking. Where the detector has been converted to a tracer, which tracks multiple objects, in order to do the trace, the regression is exploited as it does not require specific

   training of the tracer in addition to that it performs well in complex tracking scenarios.

   In October 2019 [8] Improvements were made to Tractor where a new version appeared called Tractor++ which has some improvements Those are two extensions to track pedestrians, namely the movement model and reidentification.

   Tractor++ (- 2D MOT 2015 : MOTA=53.5,- MOT16 :MOTA=54.4 ,- MOT17 :MOTA=44.1).

4. **TrackRegion-based Convolutional Neural Network (R-CNN) :**

   In a paper proposed by Paul Voigtlaender et al.in April 2019[78],One of the popular tasks was addressed, which is multiple object tracking and segmentation (MOTS). In order to do this work,a neural network-based algorithm was proposed to jointly deal with discovery, tracking, and segmentation. Its effectiveness has been based on various measures.

   This work was evaluated by several scales multi-object tracking and segmentation accuracy, multi-object tracking and segmentation precision, Soft multi-object tracking and segmentation accuracy, as well as based on KITTI MOTS and MOTSChallenge databases, and the following results were obtained on KITTI MOTS, where sMOTSA is 76.2, MOTSA is 87.8, MOTSP is 87.2, and Results on MOTSChallenge where sMOTSA is 52.7, MOTSA is 66.9, and MOTSP is 80.2.

   results are considered good and open up prospects for research in this field.

In the previous parts, we discussed the most important methods and algorithms for the task of detecting and tracking objects , As they are mandatory for enumerating targets. Next, we will address counting objects and the most important papers and works presented in this field.

Regression-based Approach Regression-based approaches predict instances and count immediately based on regressors together with images features [11][41]. Such as example, in this paper

[11], The first thing to do is segmenting the video into different components of homogeneous motion, where a Gaussian process regression is learned to calculate the number of cases for each segmented area. Cumulative attribute representation [12] was proposed and used in order to teach the regressor to handle the imbalanced training data. Inter-dependent features are used to mine the spatial importance among different regions to learn the number of counts [13]. Feature normalization is taken into account to deal with perspective projections in [41]. By drawing on the advantage of deep learning [58] Through the use of a convolutional neural network (CNN), the number of cars is calculated and estimated. the Regression-based methods cannot provide clues of the individual location of each object, which makes its potential applications limited.

density-based approach Density-based Approach first maps the image to a density map, such that the integral over any sub-region gives the count of objects within that region [89] [1] [47].This article [47],Mentions that the pixel-level density is learned by minimizing a regularized risk quadratic cost function. In addition to that Based on the density map, an interactive counting system is introduced in this research paper [6] in order to incorporate the relevant feedback.

Instead of a hand-crafted features,[89] depended and concentrated on convolutional neural networks (CNNs) based density map and instance count estimation in the cross-scene scenario. While the density map provides a certain clue of the crowdedness, Despite all this, it still suffers from the exact positioning of objects in each case.

detection-based approach The detection-based Approach gives the total count by localizing each object instance. Such approaches can be considered as the application of object detection ,and thus benefit a lot from the improvement of the object detection [22][66].

Applying the object detection approach directly in order to count the number of objects (elements) in the scene requires a lot of effort and focus, especially in scenes or images that contain elements of very small size on the massive instance labeling. In this paper [49], Under the title Themed Shape-Based Human Detection and Segmentation via Hierarchical Part-Template Matching by Zhe Lin et al, where a hierarchical part-template matching approach is proposed to detect humans, which requires careful feature and template design, while in this paper [34], Titled Drone-based Object Counting by Spatially Regularized Regional Proposal Network by Dan Kong et al, neural network learning is applied to detecting and counting car instances through incorporating layout information. That goes back to its large potential in applications demanding location information.

2.4 Summary of related object detection and tracking

We have previously seen many works in the field of object detection and tracking, and we will present the summary of these works in the following tables :

Table 2.1 – Object detection methods.

Researcher	Algorithms	Dataset Used	Main Results			
			Accuracy	Precision	Recall	FPP
W.Andrew et al , 2016 [5]	**SVM + sheft**	SVM + sheft	Average accuracy of 97%	-	-	-
Viola et Jones, 2001 [77]	**Cascad network**	MIT + CMU	Accuracy of 94%	-	-	-
Dalal et al in 2005[16]	**HOG**	MIT + INRIA	Accuracy of 89%	-	-	10-4 FPPW
Lazebni et al, 2006 [44]	**SPM**	fifteen scene categories. Caltech-101 Graz .	-	-	-	-
lines Felzenszwalb [61] et al,2008	**DPM**	PASCAL	-	-	-	-
		VOC2006	-	62.7%	-	-
		VOC2010	-	67.0%	-	-
Wang et al , 2009[81]	**HOG - LPB**	INRIA+ Pascal	-	-	-	detection rate :91.3% FPPW=10-5 97.9% FPPW=10-4 97.9%
Sanchez, et Mensink in 2010[64]	**Imporovid Ficher Kernels**	PASCAL VOC 2007 CalTech 25	Average Precision (AP) from 47.9% to 58.3%. Average Precision (ntrain=15, 34.7(0.2)) (ntrain=30, 40.8(0.1)) (ntrain=45, 45.0(0.2)) (ntrain=60, 47.9(0.4))	-	-	-

Continued on next page

Table 2.1 – Object detection methods.

Researcher	Algorithms	Dataset Used	Main Results			
			Accuracy	**Precision**	**Recall**	**FPP**
Van de Sande et al ,2013 [76]	**Selective searche**	Pascal VOC 2012	Mean Average Best Overlap of 0.879 at 10,097 locations	-	99%	-
J. Redmon et al , 2016 [65]	**YOLOv1**	PASCAL VOC 2007	-	mAP = 63.4%	-	-
J.Redmon et al ,2017 [66]	**YOLOv2**	COCO PASCAL VOC	-	- COCO get 76.8 mAP at 67 frames per second - Voc2007 get 78.6 mAP at 40 frames per second	-	-
Wen Shao et al in 2019[71]		private data set	-	precision = 0,957	0,946	-
J.Redmon et al	**YOLOv3**	Ms COCO	-	mAP =31.0%	-	-
A. Bochkovskiy et al in 2020 [10]	**YOLOv4**	Ms COCO	-	43.5% AP	-	Realtime speed of 65 FPS
Pooja Mahto et al , 2020 [54]		UA-DETRAC Benchmark	-	mAP = 67.7%,	-	-
One		UA-DETRAC Benchmark	-	mAP = 67.7%,	-	-
Dihua Wu et al 2020 [85]		Private data set	-	mAP = 97.31% - inference time was decreased by 39.47%	-	-
Xingda Sun et al , 2021 [72]		Special data set collected	-	Precision = 0.92	0.904	-
Junxiao Li et al ,2021		special data set collected	-	mAP=0.82	-	-
Wei Liu et al , 2016 [50]	**SSD**	-PASCAL VOC -Ms COCO -ILSVRC	-	- mAP1= 74.3% in VOC2007 - mAP =76.9%	-	-59 FPS For 300×300 input -512 ×512 input
W.Andrew et al , 2017[19]	**R-CNN**	-Friesian Cattle 2017 -Aerial Cattle 2017	detection and localization accuracy = 99.3%	-	-	-

Continued on next page

Table 2.1 – Object detection methods.

Researcher	**Algorithms**	**Dataset Used**	Main Results			
			Accuracy	**Precision**	**Recall**	**FPP**
F.Sarwar et al ,2018 [70]		- special data set collected	-	-	-	-
Girshick et all ,2015[22]	**FastR- CNN**	-PASCAL	-	mAP of 67.2% for Voc 2010 -mAP =65.7% for VOC 2012 -mAP (from 63.1% to 66.9%) for voc 2007.	-	-
Girshick et all in 2015[68]	**FasterR-CNN**	-Ms COCO	-	-the mAP from 41.5%/21.2% (VGG- 16) -the mAP from 48.4%/27.2% (ResNet-101)	-	-
He et al,2017 [26]	**MaskR- CNN**	-Ms COCO	-	-AP=37.1 -AP50=60.0 -AP75=39,4 -APS=16,9 -APM=39,9 -APL=53.5	-	-
B. Xu et al,18 Feb 2020 [88]		-Special data set collected	-IOU=0.5 -Accuracy for pastures 94% -Accuracy for the feed-lot 92%	-	-	-
Beibei Xu et al , 2020 [87]		-Special data set collected	-Accuracy is 96% for classifying livestock -Accuracy is 92% for counting livestock	-	-	-

Continued on next page

Table 2.1 – Object detection methods.

Researcher	Algorithms	Dataset Used	Main Results			
			Accuracy	Precision	Recall	FPP
M.Machefer et al in 2020 [52]		-Ms COCO	-	In detection : -mAP = 0.418 for potato plant -mAP=0.660 for lettuces -MOTA : - 0.781 for potato plants -0.918 for lettuces	-	-

Table 2.2 – Object tracking algorithms

Researcher	Method	Datasts	Results			
			Accuracy	Precision	ID	FPP
C. Feichtenhofer et alin 2017 [19]	**Detect and Track**	ImageNet VID dataset	-	-The performance of time step=1 is 78.7%mAP. -The performance of time step=10 is 78.6% mAP.	-	-
Bergmann et al in 2019 [8]	**Tractor++**	MOT challenge datasets	2D MOT 2015 : MOTA=53.5 MOT16 : MOTA=54.4 MOT17 : MOTA=44.1	-	-	-
Voigtlaender et al.in April 2019 [78]	**TrackR-CNN**	-KITTI MOTS dataset. –MOTS Challenge dataset.	KITTIMOTS : -MOTSP is 87.2 -MOTS Challenge : -MOTSP is 80.2	-	-	-
Wojke et al in 21 Mar 2017.[83]	**Deep SORT**	MOT challenge datasets (MOT16)	MOTA : 61.4	MOTP :79.1	-	-
Kristina Host et al in 2020 [33]	**Deep SORT**	Special data set collected	MOTA :99.3%.	-	-IDF1 :24.7%. -IDP :24.7%. -IDR :24 .7%.	-

2.5 Conclusion

In this chapter, we study the different works and papers of object detection, tracking and counting tasks.

The next chapter will present our implementation of this kind of system.

CHAPITRE 3

OUR CONSTRUCTED SYSTEM

3.1 Introduction

In this chapter, we discuss the steps that we follow to build an integrated system for detection, tracking and counting sheep.

Our pipeline process Figure 3.1 present the pipeline process that we follow to construct our model. The first step is the dataset gathered. Secondly, we pre-processed it to be used in the training and testing step, then we reveal the experimental results for our model. Next, we move to the counting stage, in which we face some challenges, last but not least we overcome the difficulties we encountered in the previous stage by tracking.

Each stage will be discussed later in detail.

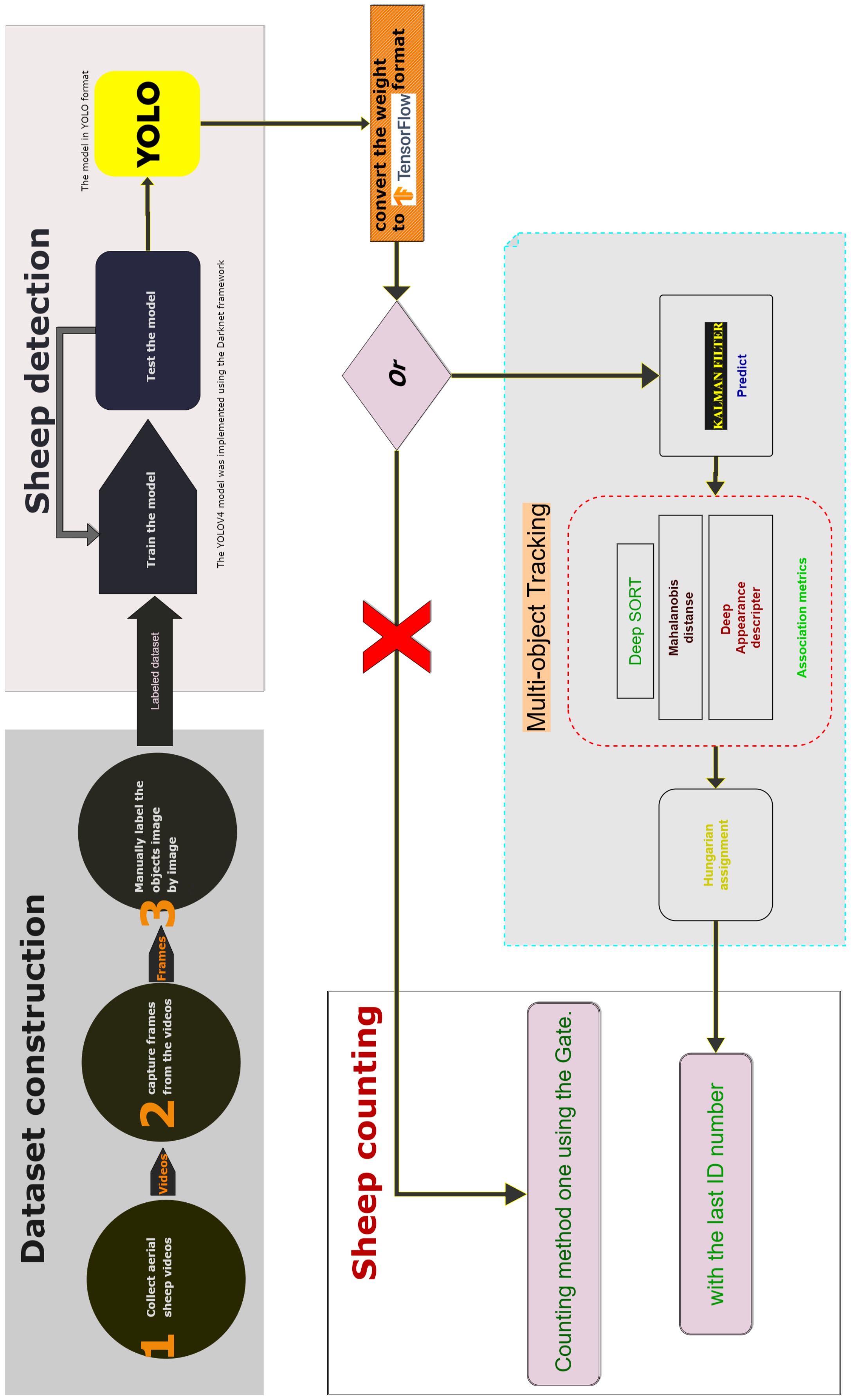


FIGURE 3.1 – Our pipeline process for this work.

3.2 Dataset construction

3.2.1 Dataset gathering

At first, our plan was to collect aerial photographs using our drone from a fixed height and horizontal angle (bird eyes view). Ensuring a uniform sheep size and to obtain high resolution and quality image data with different backgrounds, Figure 3.2.

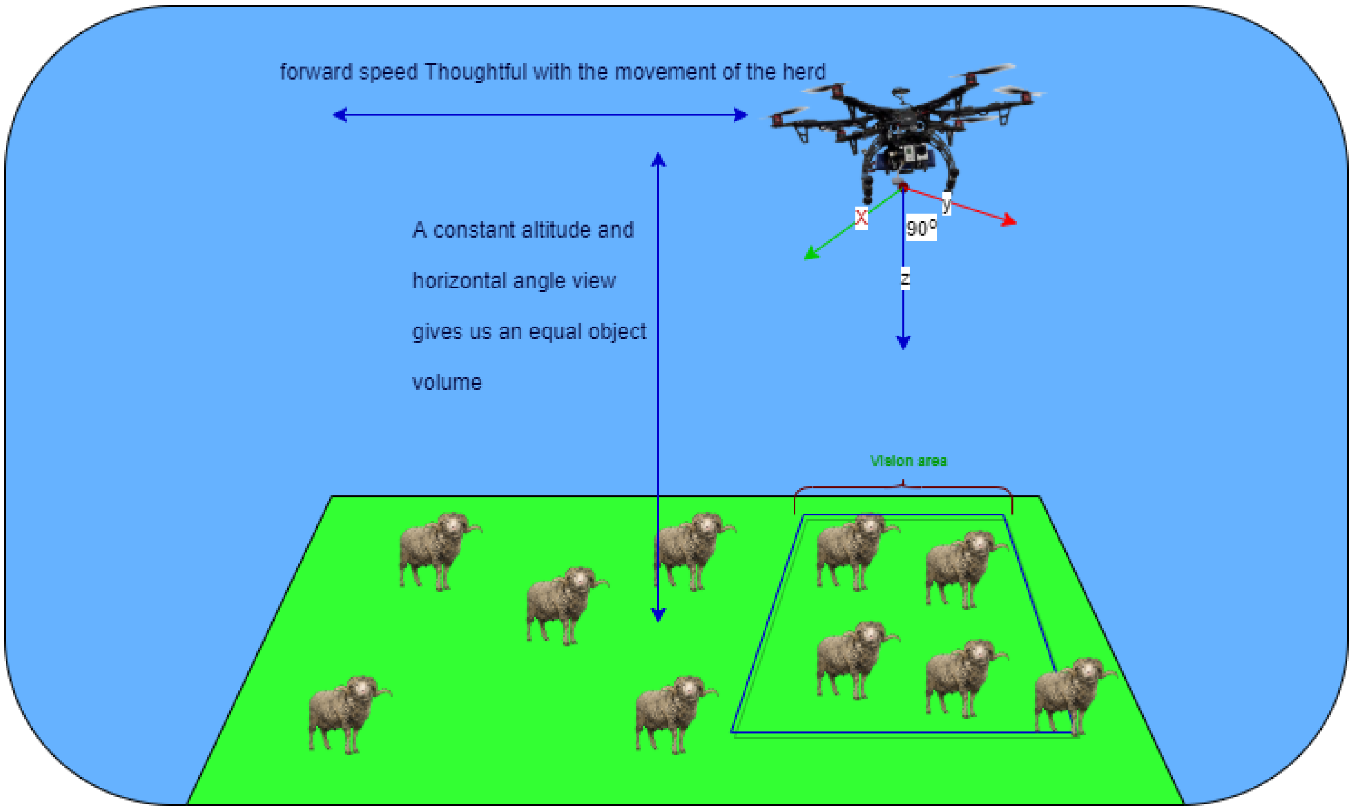


FIGURE 3.2 – Our future strategy for collecting dataset.

However, we fell short in equipment and we had to find an alternative. So we went back to the web and searched for footage using the search keyword *herding sheep with a drone.*

We downloaded from YouTube more than 79 videos ranging in length from 59 seconds to 10 minutes. We filtered away the irrelevant clips, and passed the rest through a frame extraction script. The resulting dataset consisted of 3679 carefully selected images in different weather conditions, different illuminations (at sunset, noon, etc.),different backgrounds (desert and grassy, full of trees), different angles (vertical from the side) and at variable heights.

The 80% of the extracted images were used for the training. The 20% thats left was used for validating.

3.2.2 Dataset labeling

After collecting the frames, we had to go through manually labeling the elements one by one. There are several tools available for the labeling task. but, we settled on a free tools called LabelIMG, for which the interface is illustrated by figure 3.3. This work required a continuous effort from a team of two people over a period of no less than 15 days

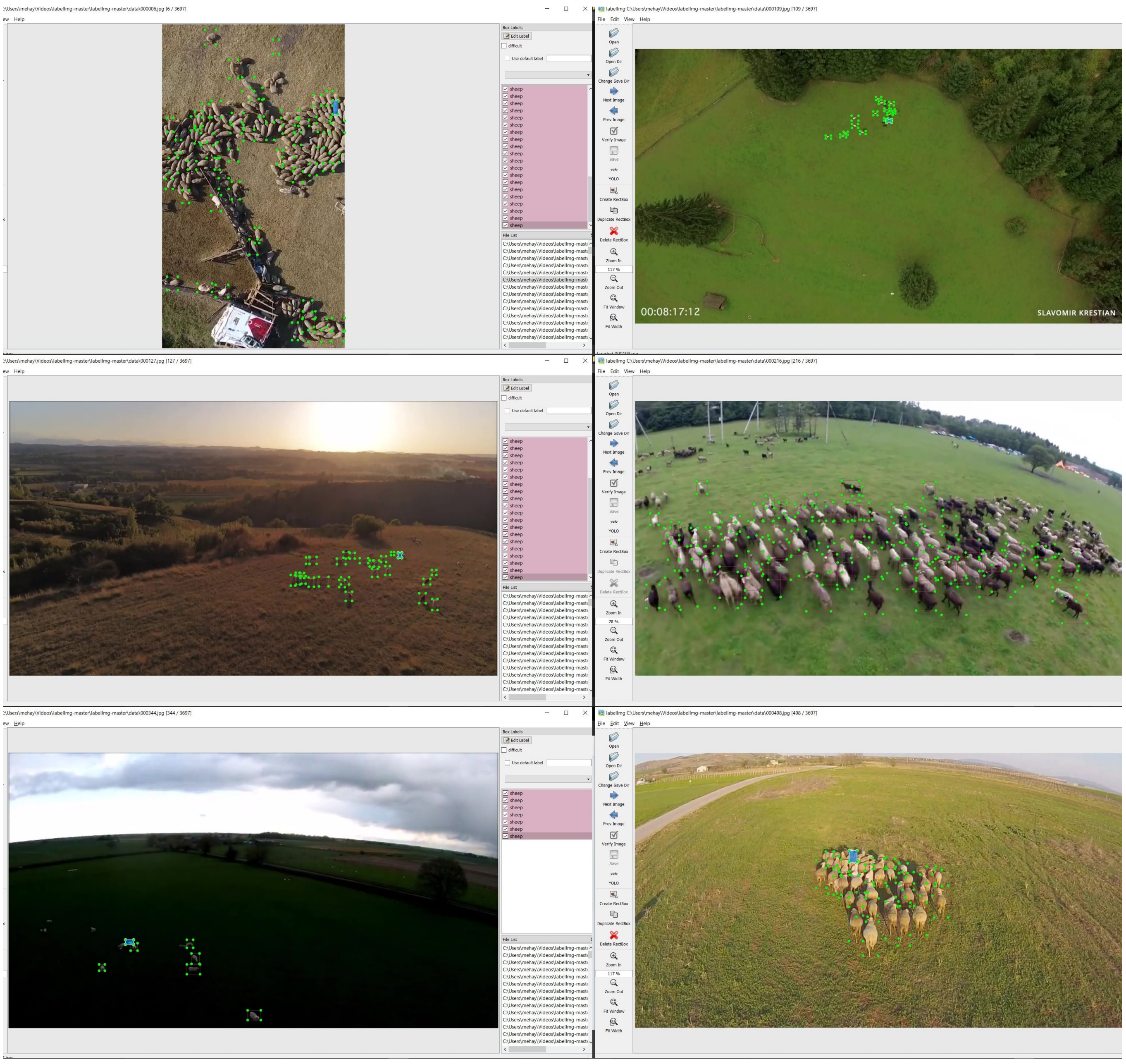


Figure 3.3 – Example of labeled images using a LabbelIMG tool

The goal of using this tool is to generate bounding boxes around the objects, and the class of these objects attached to them. Each bounding box has the following parameters (Fig.3.4).

(bx, by) : The center bounding box in the image.

bw : The width of the box.

bh : The height of the box.

c : The class of object.

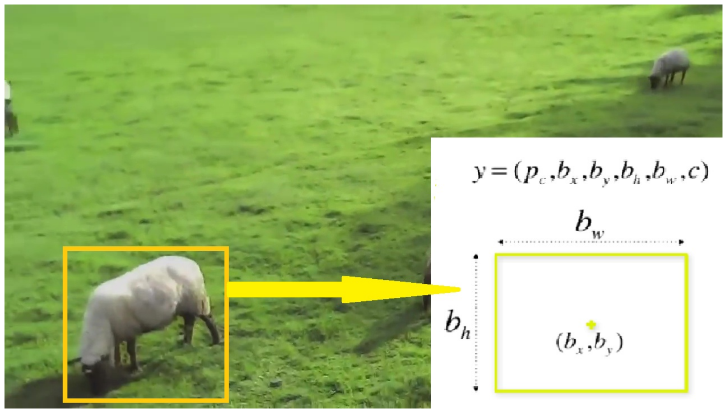


FIGURE 3.4 – Bounding box parameters

And since images ware randomly collected from the Internet, with different types of cameras The dimensions of these images will not be the same, and this was a problem knowing that training data must be unified in dimensions before being fed to the model. Treating this problem (resizing the training images) caused the boxes surrounding the objects to change because of the modification of the frame dimensions (ex. Fig 3.4(bw=21.2 %, bh=25.62%)), so the solution is to use percentages instead of the pixel numbers of the actual image.

The end result was a (*.txt) file containing information of boxes for each frame, then we create a file that links between each txt file and its frame to be used in the training process.

3.3 Detection stage using YOLOv4

The architecture of our a model consists of an fine tuned YOLOv4 network . Our problem is to detect specific objects on a generally uniform colored background, and the size of these objects from an upper angle is generally the same. That being said, a typical YOLOv4 network is more than enough for such a problem.

In order to optimize the network's performance and adapt it with the size of the objects in our dataset, We have frozen some layers from the original network, Figure 3.5. The reason is that the eliminated parts of the network are respectively responsible for detecting small and large objects in a given image.With this procedure, we can reduce the complexity of this model, increase the number of frames per second, and gain higher image resolution. Unfortunately, this decision would have been fruitful if we had full control of how the data was collected (fixed angle, fixed altitude, etc...).

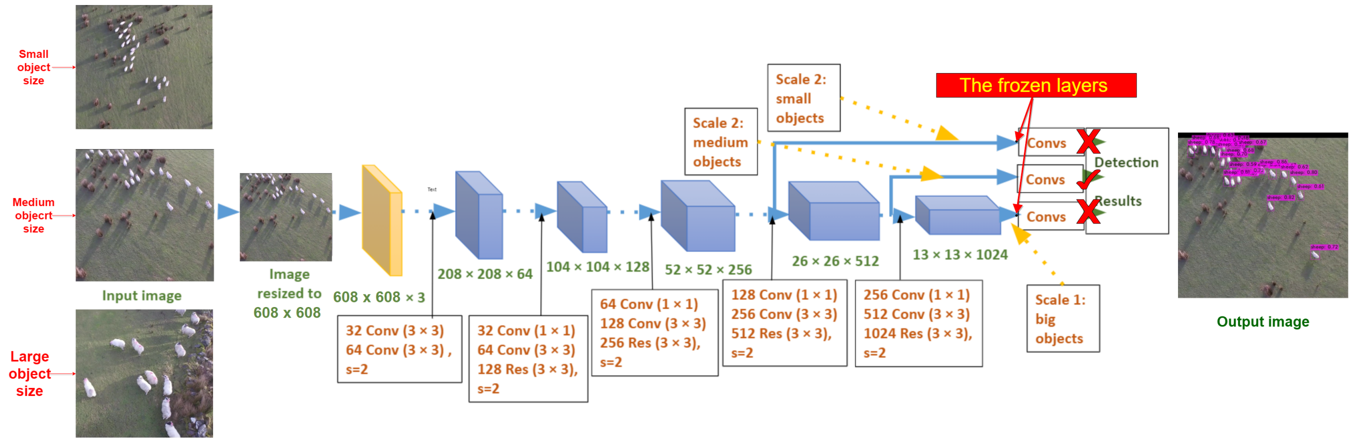

Small object size
Medium objecrt size
Large object size
Input image
Image resized to 608 x 608
608 x 608 × 3
208 × 208 × 64
104 × 104 × 128
52 × 52 × 256
26 × 26 × 512
13 × 13 × 1024
32 Conv (3 × 3) 64 Conv (3 × 3) , s=2
32 Conv (1 × 1) 64 Conv (3 × 3) 128 Res (3 × 3), s=2
64 Conv (1 × 1) 128 Conv (3 × 3) 256 Res (3 × 3), s=2
128 Conv (1 × 1) 256 Conv (3 × 3) 512 Res (3 × 3), s=2
256 Conv (1 × 1) 512 Conv (3 × 3) 1024 Res (3 × 3), s=2
Scale 2: medium objects
Scale 2: small objects
Scale 1: big objects
The frozen layers
Convs
Convs
Convs
Detection Results
Output image

FIGURE 3.5 – YOLOv4 architecture after modification

The choice of the YOLOv4 was made for a number of reasons, including :

- It is a object recognition system that can recognize multiple objects in a single frame.
- YOLO recognizes objects more precisely and faster (real-time) than other recognition systems.
- It can predict up to 9000 classes and even unseen classes, That is, we can easily add more objects in the future.
- Less complicated compared to its counterparts.

3.3.1 Training detection model

The model was implemented using the ***DarkNet*** framework, which is an open source neural network framework written in **C** and **CUDA**. It is fast, easy to install, and supports CPU and GPU computations. Its code is publicly available in Github [10] [79].

We reconstructed the YOLOv4 network with the modifications mentioned above and trained it with the following parameters While preserving the above-mentioned layers .

Batch size 64.

Network size 608 * 608.

Subdivisions 16.

MaxBatches 6000

Steps 4800, 5400.

Number of filters for the last conv layer ((classes=1) + 5)x3=18.

The training was not launched from scratch, instead we instantiated the network with the MS COCO dataset weights aiming to make use of transfer learning.

Throughout the training process, we used the callback function to save the model weights with the aim of providing progress in case of training failure or the internet lag. Google Collab gives us 12 hours of free use and checks if there is an interaction every 1 :30 hours. To complete the training, we need at least 60 hours of continuous connection on the cloud. Unfortunately, disconnection happens every 90 minutes as a security measure. Which prompted us to use some tricks like saving training weights every 1000 epochs, The saved training file we called (yolov4-custom-1000.weights, yolov4-custom-2000.weights, Etc...) And we also linked the Collab to the Google drive because the memory of Collab is temporary.

Software (Google Collab)

Google Colaboratory is a free online cloud-based Jupyter notebook environment that allows us to train our deep learning models on advanced virtual machines with pre-installed python packages. It provides decent hardware as well.

Hardware

The hardware provided by Google Collab is as follows.

GPU NVIDIA Tesla K80

RAM 12 GB

Storage 10 GB in Google drive cloud

3.3.2 Detection result

The performance of our model is determined by three parts :

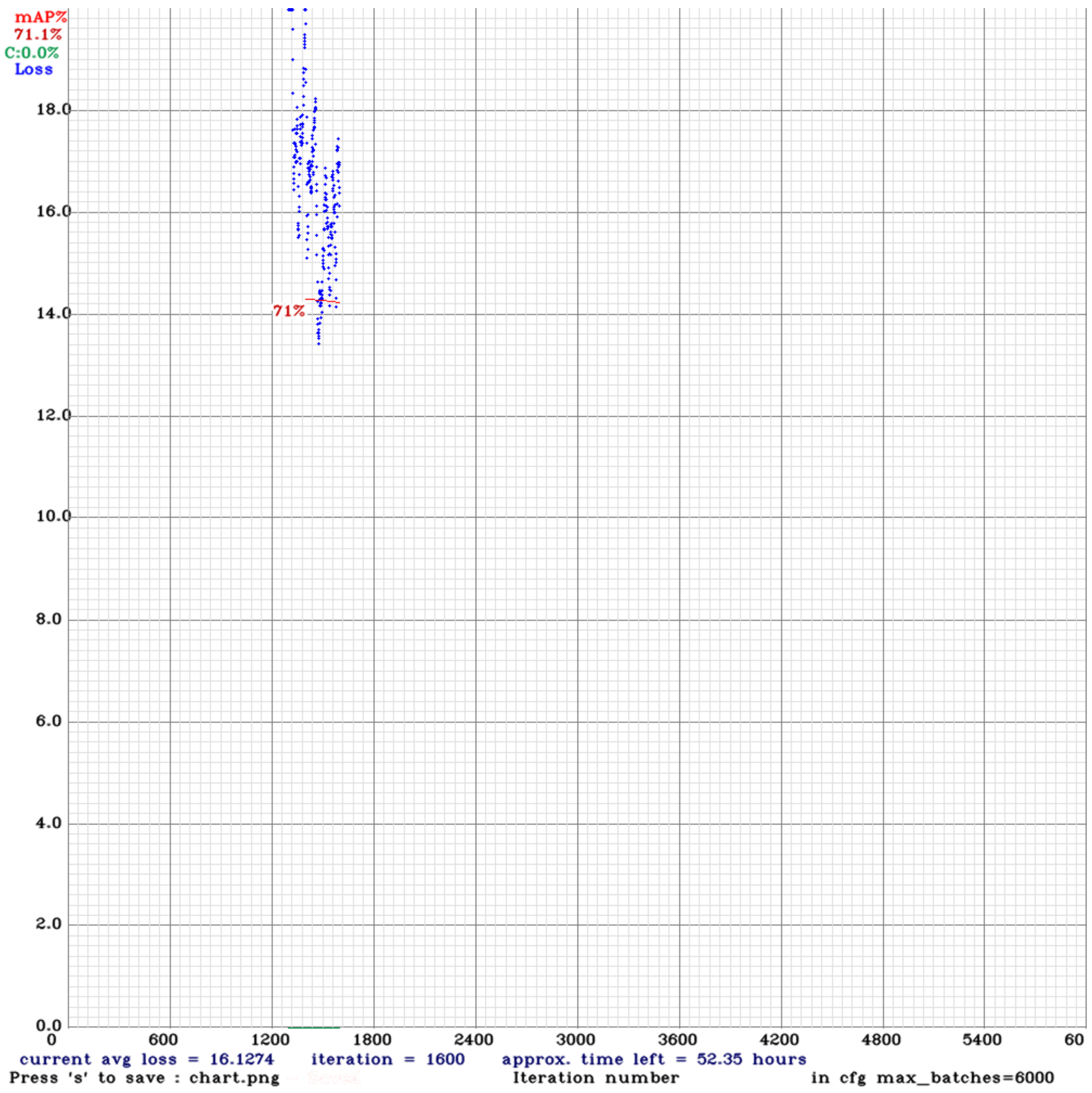
— Training mAP and loss.

— Testing mAP and loss.

— FPS.

Training of the detection result

After many training attempts, interruptions and start overs, we obtained the following results for which Figure 3.6 describes the improvements the maximum average precision mAP and the loss on the y-axis with respect to the number of iterations on the x-axis in the training process. We can observe that by the end of the training the model achieved a mAP equal to 71% with a loss of 16.1274 These values are considered good judging by the number of training images provided. In other words, the model is expected to perform better if there is more training data. Which is obtained through the use of our own drone rather than collected from the internet and if there is no interruption in the training, We get a continuous graph extending from top to bottom, but this graph represents the last training of the model after many interruptions



```
current avg loss = 16.1274    iteration = 1600    approx. time left = 52.35 hours
Press 's' to save : chart.png
```

FIGURE 3.6 – After completing the last training process

Testing of detection result

In the testing phase, we used 20% of the collected dataset. We randomly selected images and then passed them on to the trained model. The results obtained were analyzed manually or by observation. The selected set of images was varied and in different conditions (eg vertical viewing angle 3.8, low illumination 3.9, interference between animals, which gave us an occlusion state 3.10, side view angle 3.7).

Frames Per Second

As we saw in several previous papers, images of the objects were captured by drones, but such situations pose a problem as the number of studied object increases the area they cover increases accordingly. Intuitively, we can increase the height of the capturing device to cover all the elements risking the drop in image quality and the change in sheep sizes. Instead, we decided to process a video (frame per frame) From a constant height with the movement of the drone forward according to the motion of the sheep.

The model ability to process as many frames as possible in one second, is considered as important as classification accuracy because it reflects the model ability to run in real time with a video stream from a real acquisition device that can be linked to a processor for a specific task

Our use of the simplified YOLOv4 model worked in favor of the FPS value as our model was able to run at 271 FPS with images of resolution 608*608 Such FPS value indicates that we can still improve the frame resolution without affecting the processing speed of the model and this edit can improve the accuracy of the classification in an impressive way.

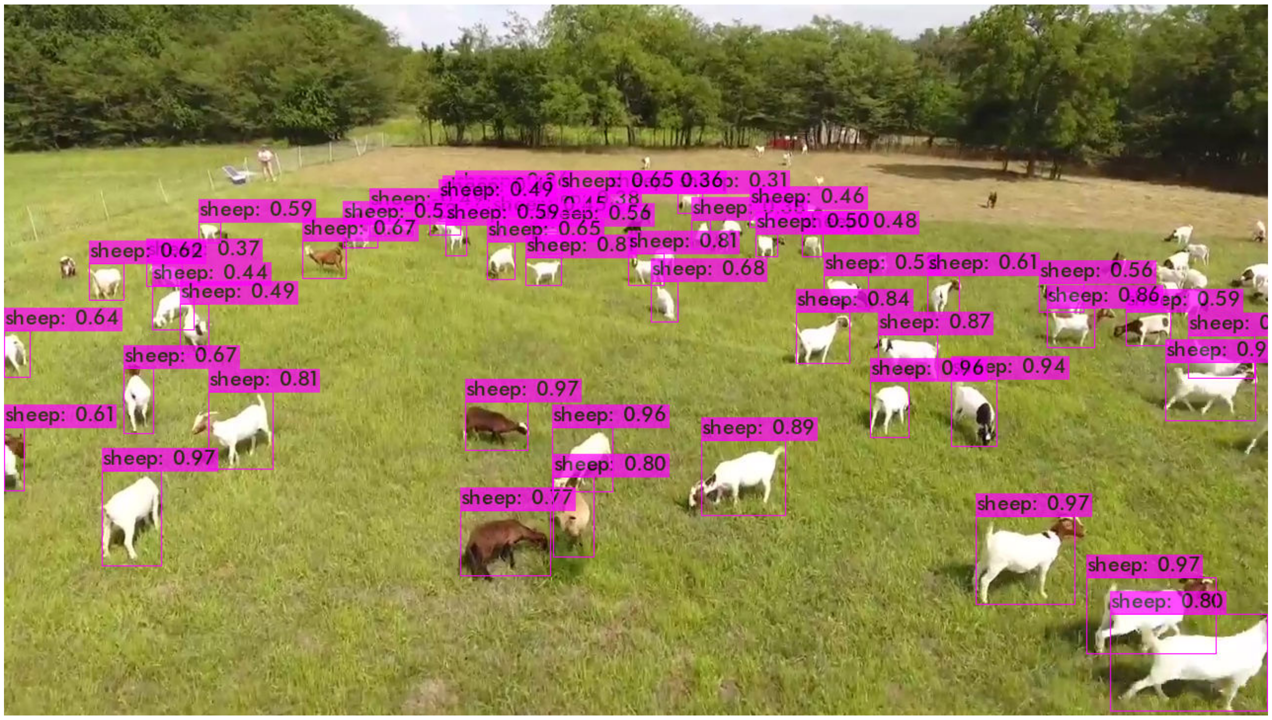


Figure 3.7 – Test example of sheep detection from a side view angle

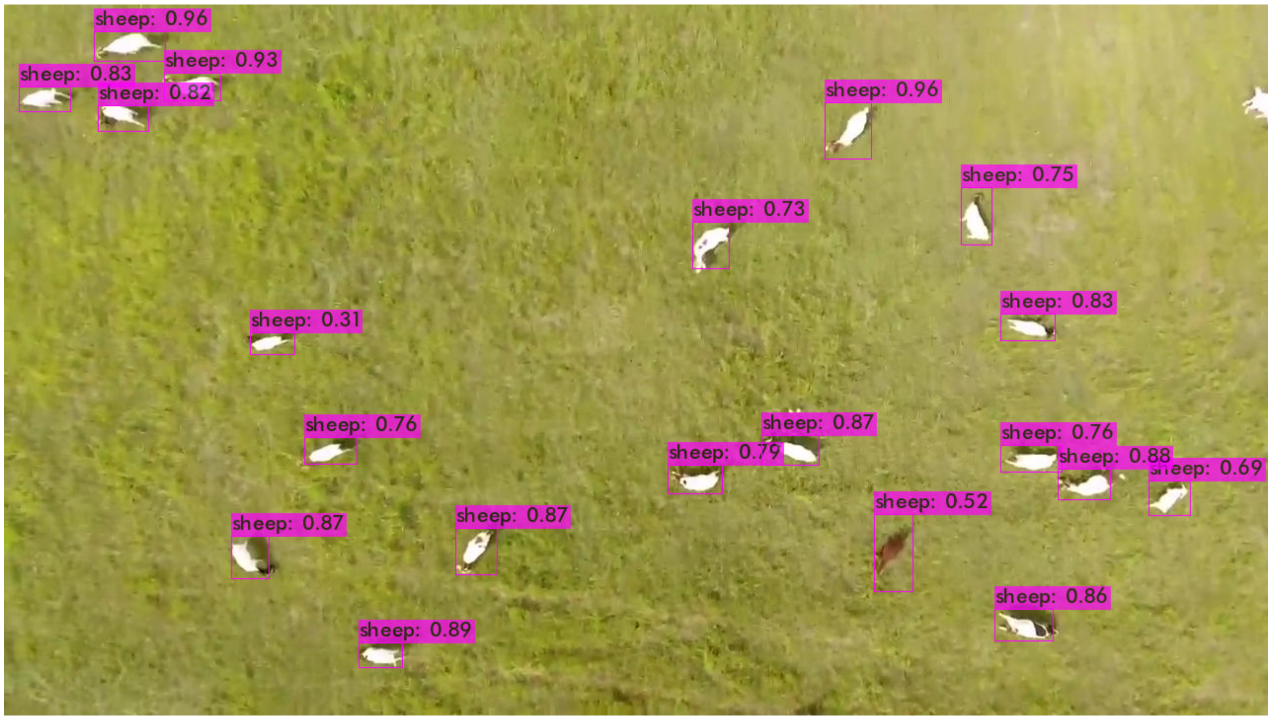


FIGURE 3.8 – Test example of sheep detection from a horizontal viewing angle at high altitude

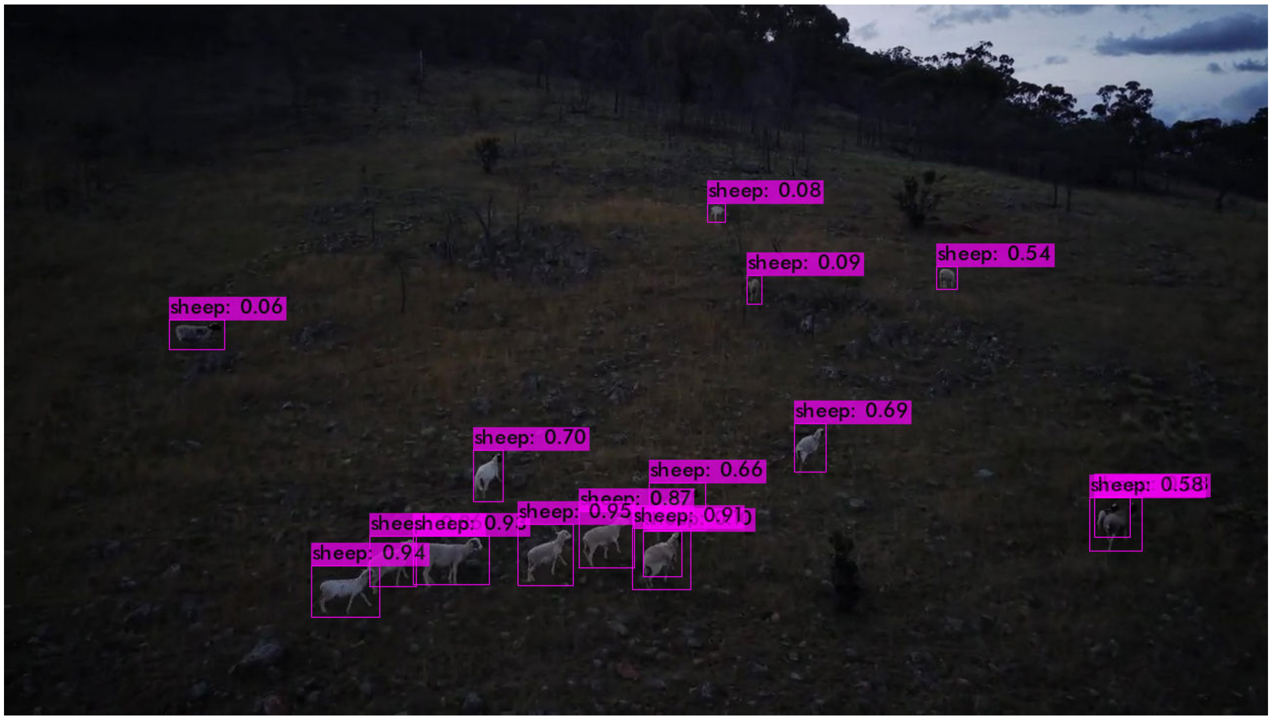


FIGURE 3.9 – Test example of sheep detection from a side view angle in low light

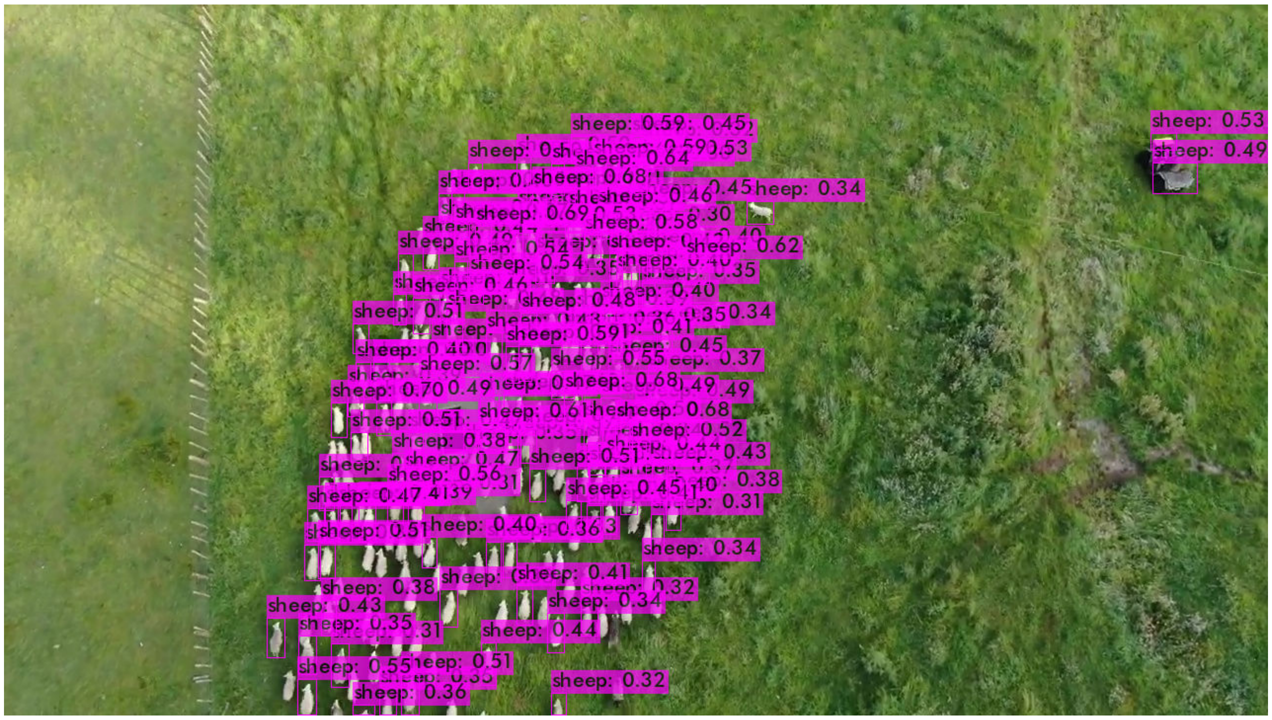


FIGURE 3.10 – Test example of sheep detection with occlusion .

3.4 Counting stage using the Gate method

After detecting objects in the video (frame by frame) using bounding boxes, we choose to make the counting method simple and low in complexity, to do this process we use a computer vision library (Opencv) where we draw an imaginary line as a gate that cuts the image in the middle and when the coordinates of the box intersect with the gate we add one number to the sum of sheep Fig 3.11.

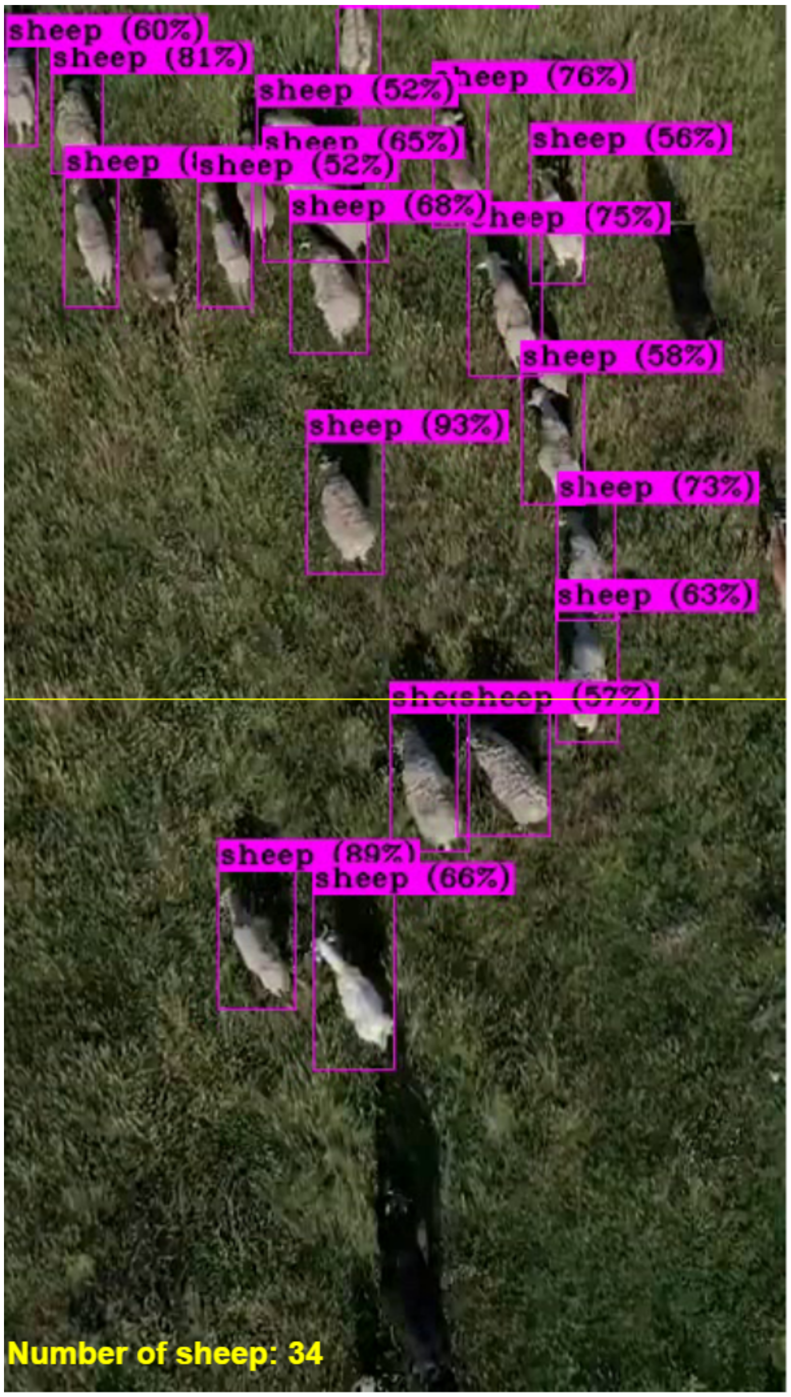


FIGURE 3.11 – Example results of a simple counting method using the Gate.

3.4.1 Counting result

As we were testing the model, new challenges surfaces such as :

— We know the frame has a large number of sheep then we don't have a link between the sheep in current frame and the previous frame, even though they are the same objects. in each frame there is a new detection and new bounding boxes.

— If the sheep gets out of the camera field and another one appears, we have no way of knowing if they are the same or not.

— We know that sheep move forward at similar speeds, but there are some cases where the movement is random For example, if a sheep cuts through the gate, the system counts it, then stops eating for example, and suddenly runs fast to highlight it again, the sheep is counted twice, even though it is one.

Conclusion on the use of this method : essentially with detection we work with one frame at a time and we have no ideas about the motions and past movement of also we don't know if they the same or not.
Therefore, it is necessary to move to the path Tracking stage.

3.5 Multiple Object Tracking stage using Deep SORT

After the catastrophic failure of the previously mentioned counting Gate method, we had to move to simple online live stream tracking in real-time [9, 84] with a deep metric association. Before developing the algorithm, we had to make some modifications to the previously trained model.
So, we converted the weights of the model that was trained using the DarkNet framework (in C language) to TensorFlow format, to reconstruct the YOLOv4 architecture using Keras framework with Python so that we associate the YOLO object detector with the Multiple Object Tracker algorithm according to procedures that consist of many complex concepts and calculations. All the processes involved are explained below in detail Fig3.12.

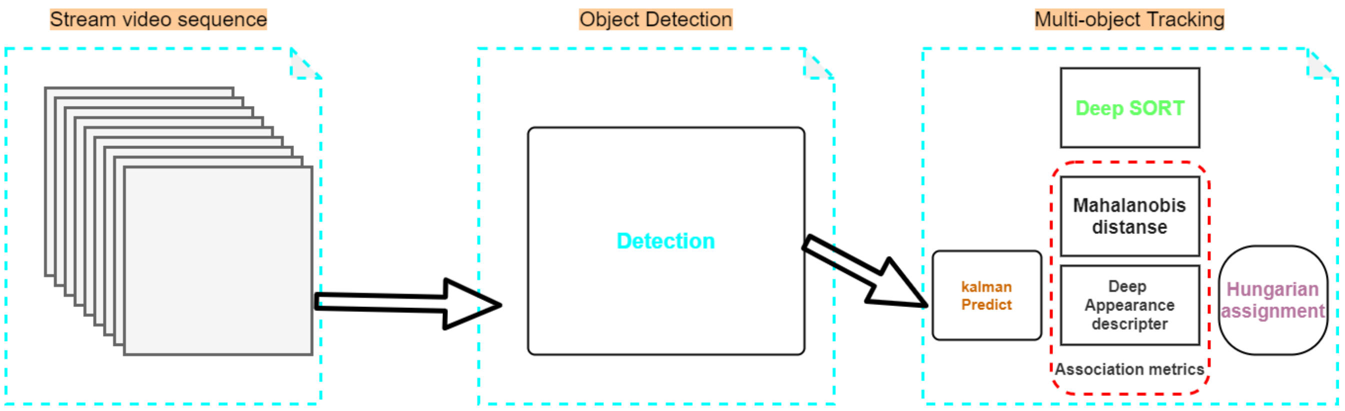


FIGURE 3.12 – From detection to the tracking system.

1. Detection :
   — First, all the sheep are detected in the frame using YOLOv4 framework.
2. Motion prediction :

— Once we have a detection (object position(x, y), bounding boxes ratio (γ), scale) for the frame, matching is performed for similar detections with respect to the previous frame.

3. Tracking :

   — The process of locating moving sheep over time in the video sequence (tracking the motions and appearances) involves.

   While continuous video {

   (a) Tacking in an initial set of sheep detections (box location).

   (b) Creating a unique ID for each of the sheep.

   (c) Tracking the sheep over time.

   (d) Maintaining the ID assignment with deep association Matrix.

   }

3.5.1 Tracking tools

Kalman Filter

There are a variety of engineering problems that require time series prediction, such as navigation systems, brake control mechanics, missile launch systems, computer vision challenges, and even economics. One of the best solutions to these problems is the Kalman filter developed by researcher Rudolf Kalman, who published his paper in 1960 [37] containing a solution to the problem of linear filtering for recursively discrete data.

Kalman Filter (KF) estimates the state of a system by processing all available measurements regardless of their accuracy and works well with Gaussian distribution and linear models.

The filter is very powerful in many ways : it supports estimates of past, present, and future states, and can do so even if the actual structure of the model system is unknown.

The Discrete Kalman Filter Tackles the broad problem of trying to estimate how the linear stochastic difference equation is regulated by a discrete time controlled process.

Note that states $x \in R^n$.

$$x_{k+1} = A_k x_k + B u_k + w_k \tag{3.1}$$

with a measurement $z \in R^m$ that is represented in the equation3.2.

$$z_k = H_k x_k + v_k \tag{3.2}$$

knowing that :
X_k : The state vector containing the terms of interest for the system at time k.
U_k : The vector containing any control inputs.
Z_k : The vector of measurements.

$\mathrm{A_k}$: The state transition matrix applies the effect of each system state parameter at time k on the system stat rat time $k+1$ without controls.
B : The control input matrix which applies the effect of each control input parameter in the vector $\mathrm{U_k}$ on the state vector $\mathrm{X_{k+1}}$.
$\mathrm{W_k}$ and $\mathrm{V_k}$: Random variables representing the process and measurement noise that is assumed to be independent and normally distributed with covariance $\mathrm{R_k}$ and $\mathrm{Q_k}$ respectively.

The Discrete Kalman Filter Algorithm Estimates the process by means of some kind of feedback check when the filter assesses the process status at a certain point in time and then receives feedback as measurements.
The Kalman filter equations are divided into two groups as follows : time update equations, which are also known as prediction equations, and measurement update equations, which are known as corrected equations.
The time update equations are responsible for the projection of the present status and error covariance estimates for obtaining previous estimates to the next time phase.
The measurement update equations are responsible for the feedback to be added in the previous estimate to produce an improved following estimate.

1. The time update equations for Kalman filter Discrete (predict) :
   - The first step in the prediction process, which we call the state head, is represented by the following equation 3.3.
   $$\hat{x}_{k+1}^{-} = A_k \hat{x}_k + B u_k \tag{3.3}$$
   - The second step in the prediction process that predicts the common variance of error is represented by the following equation 3.4.
   $$P_{k+1}^{-} = A_k P_k A_k^T + Q_k \tag{3.4}$$
2. The measurement equations for Kalman Filter Discrete(correct) :
   - The first step in the correction process, During which computing Kalman filter gain, is represented by the following equation 3.5.
   $$K_k = P_k^{-} H_k^T \left(H_k P_k^{-} H_k^T + R_k \right)^{-1} \tag{3.5}$$
   - The second step, During which an Update estimate using measurement $\mathrm{z_k}$, is represented by the following equation 3.6.
   $$\hat{x}_k = \hat{x}_k^{-} + K \left(z_k - H_k \hat{x}_k^{-} \right) \tag{3.6}$$

— The third step, During which the error variance is updated, is represented by the following equation 3.7.

$$P_k = (I - K_k H_k)\, P_k^- \tag{3.7}$$

The most realistic problem involves non-linear functions, the non-linear function load to non-gaussian distributions. so the KF is not applicable anymore.
To solve this problem, a solution has been proposed, which is an extension of the Kalman filter is called **the Extended Kalman Filter (EKF),** where the idea of local linearization is adopted, It has achieved good results in many applications, but it is not ideal in all cases, because it can diverge if the non-linear lines are large. Therefore, a solution was proposed based on choosing more than one point and doing the mapping, and this solution is called the **unscented transform (UT)** .

Mahalanobis distance

In 1930 a research paper was published by the Indian scientist P. C. Mahalanobis [53], that talks about basic statistics in multivariate analysis.

MD has been used in many different fields where its use is by calculating the distance between vectors concerning different practical uses.

The distance of the Mahalanobis is calculated by the following equation (traditional definition) :

$$\Delta^2 = (\boldsymbol{x} - \overline{\boldsymbol{x}})\boldsymbol{S}^{-1}(\boldsymbol{x} - \overline{\boldsymbol{x}})^T \tag{3.8}$$

Where :
$\boldsymbol{x}$: $(\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots \boldsymbol{x_n})$ and $\overline{\boldsymbol{x}}$: $(\overline{\boldsymbol{x}}_1, \overline{\boldsymbol{x}}_2, \ldots \overline{\boldsymbol{x}}_n)$
$\overline{\boldsymbol{x}}$:Matrix of distances from the mean.
$\mathbf{S}^{-1}$: Inverse of the covariance matrix.
$\mathbf{T}$: transpose of matrix.

Hungarain assignment

method of Hungarian it was presented and published in 1955 by researcher Harold Kuhn [43], and it was given this name in relation to Hungarian scientists and researchers in mathematics because it is based on previous research by them.
Moreover, many researchers have noticed the extent of its strength and effectiveness, and for this reason, some additions were made to it, which increased its effectiveness. where it was called the Munkres assignment algorithm.
The Hungarian algorithm addresses the assignment problem, which appears when the values are equal, for example, we want to provide n machines to n workers. Where you minimize the cost or maximize the cost. This process is carried out through a series of sequential steps in special conditions

3.5.2 Motion prediction

An estimating model is a step between data association and the detection paradigm. To model a dynamic system, we apply Kalman filter for tracking element is considered out of frame or lost if it is not detected in a number of consecutive frames. The new track is initiated for a newly detected box. In addition, the bounding box descriptors are computed using a pre-trained CNN.

State estimation

The tracking scenario is specified in an eight-dimensional state space (u, v,γ, h,x',y',γ',h'),where h is the height, (u, v) is the bounding box position, and γ is the aspect ratio, as well as their associated velocities in image coordinates.
A typical Kalman Filter is utilized with constant velocity motion, and the velocity components are solved using a linear observation model.

Trajectory processing

This tells as to when the trajectory ends and when a new one begins. Each track has its own threshold of 30 frames for tracking the duration between a successful time from the last successful match to the current time. The track is removed if the value exceeds the previously determined threshold. For detectors that have no matching success, a new trajectory may be generated.

3.5.3 Association

In the last phase, a link is created between the newly detected box in the current frame and old object tracks in the previous frame, based on the estimated states using Kalman filtering using the previous information and the newly discovered box in the early frame. This is calculated using the squared Mahalanobis distance between the detection and the track position predicted by the Kalman filter and the Cosine distance that extracted from the appearance feature set.

Motion matching

The match between the currently active track and the current detection is referred to as matching. The term "effective trajectory" refers to trajectories that are still alive, that is, trajectories that have both tentative and verified states. The Mahalanobis distance between the detection and the track position predicted by the Kalman filter is used to plot the degree of motion matching. By calculating how many standard deviations the detection is away from the mean track location, the Mahalanobis distance accounts for state estimate uncertainty 3.9.

$$d^{(1)}(i,j) = (d_j - y_i)^T S_i^{-1} (d_j - y_i) \tag{3.9}$$

represents the degree of motion matching between the jth detection and the ith trajectory, where S_i is the observation space's covariance matrix at the current time expected by the Kalman filter. The ith track distribution's projection into measurement space is $(y_i,\, S_i)$.

Appearance matching

The use of the Mahalanobis distance matching metric alone can lead to major problems like ID Switch, especially when the drone moves, making the Mahalanobis distance metric incorrect, hence apparent matching should be used instead. The deep network is used to extract the feature vector for each detection, including those in the track. Each tracking target has its own gallery, which stores the feature vectors of the past 100 frames that it successfully associates with.The second metric is the minimal cosine distance between the feature set of the ith track's past 100 successful associations and the feature vector of the current frame's jth detection result.

Deep appearance descriptor

The purpose of the re-identification model is to build a feature representation from different training identities that can be used to execute nearest neighbor searches on images and identities provided at test time. The basic idea behind a simple re-identification model is that the output of a CNN model without the last fully linked layer may be used to generate a sheep image. Then we can calculate the cosine similarity of two animal traits to see if they are the same animal or not.

We modified the network architecture Fig.3.13from the original model used in the Deep SORT framework to get more features and to increase the accuracy of the trained model which in turn would improve the tracking efficiency. The modified CNN architecture of the re-identification model is as follows.

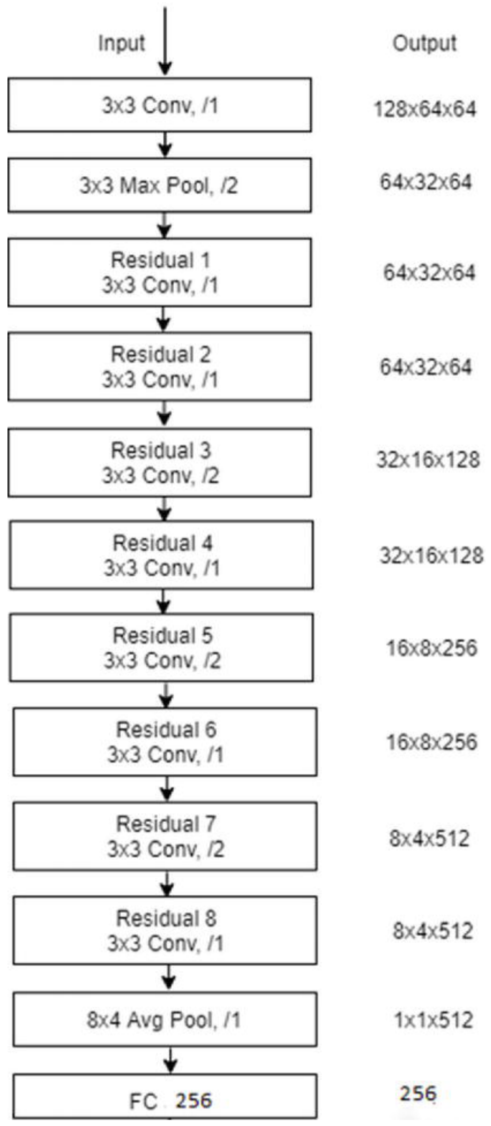


FIGURE 3.13 – The modified CNN architecture of re-identification model.

Various architectures were tested, but this one provided the highest level of accuracy. To obtain 512 features, one of the Convolutional layers prior to Max Pooling is eliminated, and eight residual blocks are used before Average Pooling. Before applying l2 normalization, the dense layer produces 256 feature vectors. Additional layers also reduced performance since the feature maps couldn't provide enough spatial resolution as they became more detailed. This model employed the ReLu activation function instead of ELU.

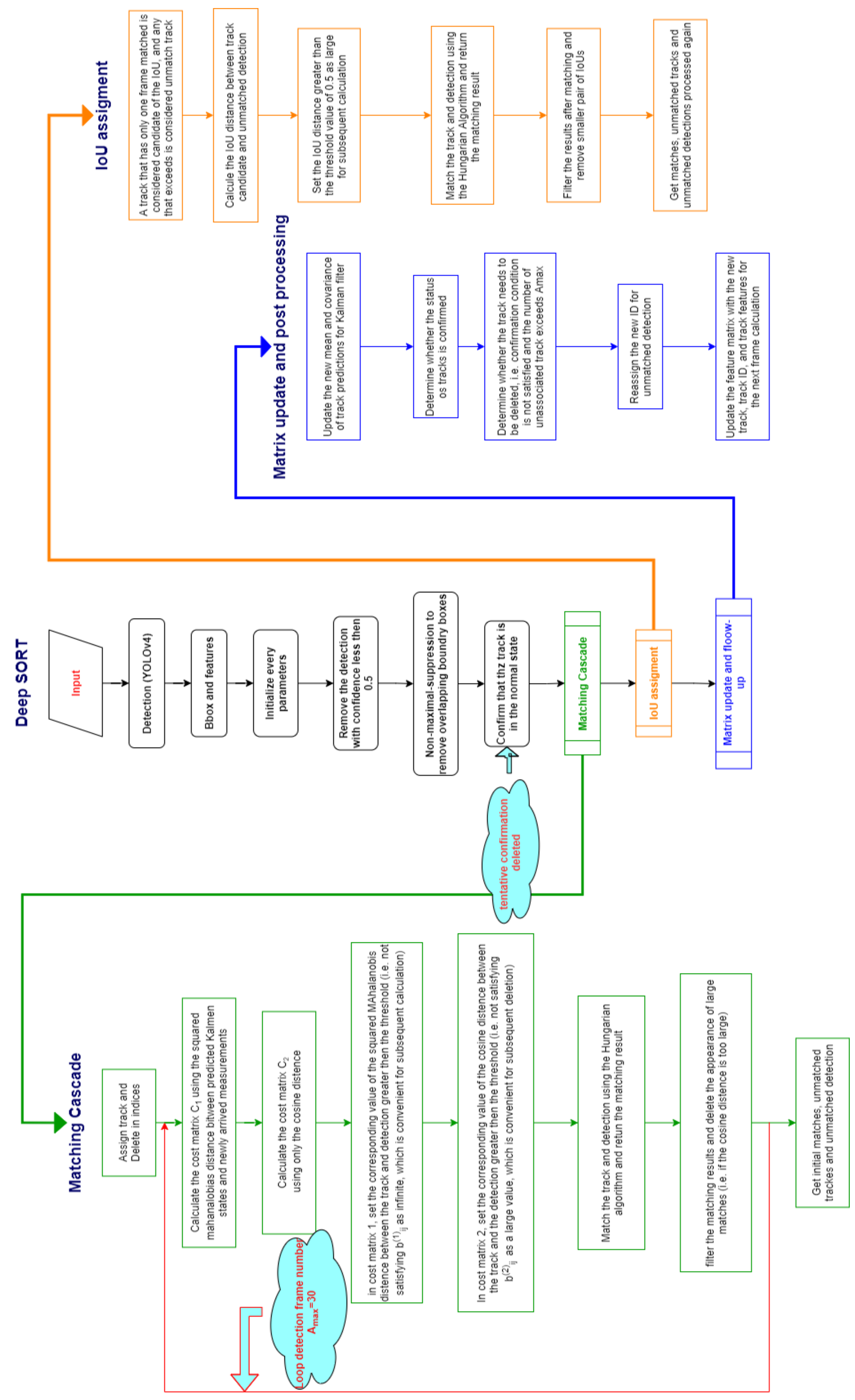

Deep SORT
Input
Detection (YOLOv4)
Bbox and features
Initialize every parameters
Remove the detection with confidence less then 0.5
Non-maximal-suppression to remove overlapping boundry boxes
Confirm that thz track is in the normal state
Matching Cascade
IoU assigment
Matrix update and floow-up
tentative confirmation deleted
IoU assigment
A track that has only one frame matched is considered candidate of the IoU, and any that exceeds is considered unmatch track
Calcule the IoU distance between track candidate and unmatched detection
Set the IoU distance greater than the threshold value of 0.5 as large for subsequent calculation
Match the track and detection using the Hungarian Algorithm and return the matching result
Filter the results after matching and remove smaller pair of IoUs
Get matches, unmatched tracks and unmatched detections processed again
Matrix update and post processing
Update the new mean and covariance of track predictions for Kalman filter
Determine whether the status os tracks is confirmed
Determine whether the track needs to be deleted, i.e. confirmation condition is not satisfied and the number of unassociated track exceeds Amax
Reassign the new ID for unmatched detection
Update the feature matrix with the new track, track ID, and track features for the next frame calculation
Matching Cascade
Assign track and Delete in indices
Calculate the cost matrix C_1 using the squared mahanalobias distance bitween predicted Kalmen states and newly arrived measurements
Calculate the cost matrix C_2 using only the cosine distence
in cost matrix 1, set the corresponding value of the squared MAhalanobis distence between the track and detection greater then the threshold (i.e. not satisfying $b^{(1)}_{ij}$ as infinite, which is convenient for subsequent calculation)
In cost matrix 2, set the corresponding value of the cosine distence between the track and the detection greater then the threshold (i.e. not satisfying $b^{(2)}_{ij}$ as a large value, which is convenient for subsequent deletion)
Match the track and detection using the Hungarian algorithm and retun the matching result
filter the matching results and delete the appearance of large matches (i.e. if the cosine distence is too large)
Get initial matches, unmatched trackes and unmatched detection
Loop detection frame number A_{max}=30

FIGURE 3.14 – The tracking procedure pipelines.

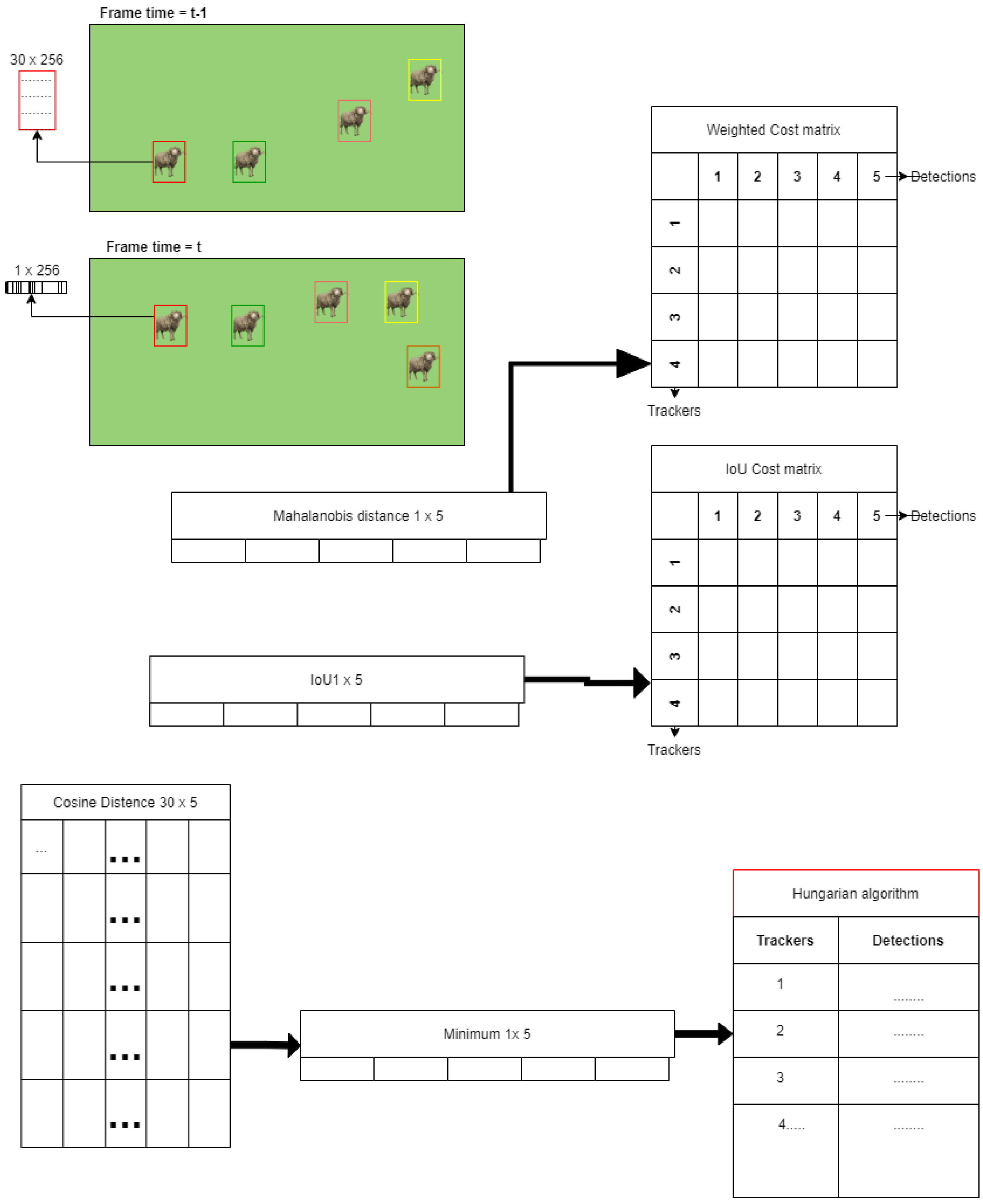


FIGURE 3.15 – The tracking procedure involves flowcharts.

As shown by this system flowchart ans visual format Fig.3.14 and 3.15 :

a Determine the position of each detected sheep in each frame and its feature.

b We use the confidence level to filter the detection frame and features, for example if the frame is below the confidence level we delete it.

c We predict the sheep position by using Kalman filtering.

d In addition to updating the Kalman tracker parameters and feature set, evaluate the target's disappearance and the new target's reappearance if necessary.

e Match tracking prediction results with the results of the prediction.

It's important for the tracker to differentiate between confirmed and unconfirmed states.

Cascade matching is enabled if the tracker has a confirmed status.

Many trackers with the same disappearance time, compute a cosine distance matrix between each newly detected object and the feature set that was previously stored by each one. For example, if there are five detection targets in the current frame, there are four trackers, and each tracker has kept the depth information (256 features) of the past 30 frames. 4 x 5 cost matrices are calculated for each tracker. then we Calculate the (1-cosine distance) between the five new detection target features of the current frame for 30 features. then get the least cosine distance for each detection block, then get a 1 x 5 matrix, and put it in the cost matrix corresponding row, showing the minimum cosine distance between current tracker and current detection block.

Using a Kalman filter prediction and a detection frame to calculate the Mahalanobis distance between them. As a result of this, each detection frame is converted from the original format of [x, y, w, h] to [center x, center y, aspect ratio, height]. There is a tracker for every type of sheep. in a cost matrix for each tracker (i.e. each row), we calculate the Mahalanobis distance between predictions and detections.There is a matrix with a distance of 1×5 for the current row in the cost matrix, assuming that there are 5 detection results in the frame.

To make the element greater than the maximum distance in the cost matrix, we adjust the cost element Matrix $>$ Maximum distance in cost matrix properties.

Assign the track to the correct detection by utilizing the Hungarian algorithm to find the cost matrix with the lowest minimum cost.

Unessential detections and traces are discarded after the assignment is complete. It is also transmitted to the unmatched detections and unmatched tracks when the cost matrix value of the matching is larger than the maximum distance (threshold).

Based on the Intersection over Union (IoU), further matching of unconfirmed and unmatched tracks is performed. Calculating the cost matrix is the specific implementation of this. As a result, the cost matrix is set as input using the Hungarian algorithm. When the job is done, Mismatched outputs, unmatched detections, and unmatched tracks.

f To updating the parameters :

Calculating the Kalman filter formulae is the process of parameter update.

$$K(k) = p(k)H^T \left(Hp(k)H^T - R\right) - 1 \tag{3.10}$$

$$x(k) = x(k-1) + K(k)(z(k) - Hx(k-1)) \tag{3.11}$$

$$p(k) = (1 - K(k)H)p(k-1) \tag{3.12}$$

R : the measurement noise.
K(k) : the Kalman gain.
z : the measurement vector.
H : the measurement matrix.

As soon as the parameter update is complete, the feature is added to the tracker's feature set, and the parameter values are re-initialized.

i Removing unrelated trackers : Even though the new position is predicted, the detection box does not match.

ii Start unmatched detection to a fresh tracker : Assume that there is no match detection, indicates the appearance of a new target to be tracked, since the this time, new tracker is started as well as Kalman filter.

iii Deleting the trackers that need to be removed.

iv Update the remaining trackers feature set.

3.5.4 Tracking result

Image Caption As a commentary on Images 3.16, we note a chronology of a set of frames, processed by the system where the system gave an identification number for each sheep. At $t = n$, if we focus on (sheep -8) to the left of the pictures. it is walk until it get out of the frame, besides at $t = n + 1$ it doesn't show almost. Importantly, The system is still able to track it.
And at $t = n + 2$, it goes out of view and back again without losing its trajectory, and that's the challenge we managed to complete.
In addition, at $t = n + k$ even if there is a cutout in the video, while that the sheep still carries the same ID without losing it.

The counting According to the type of data that we have, we found that the best way to carry out the counting is by obtaining the last ID for the sheep herd since the numbers are given in a sequential manner.

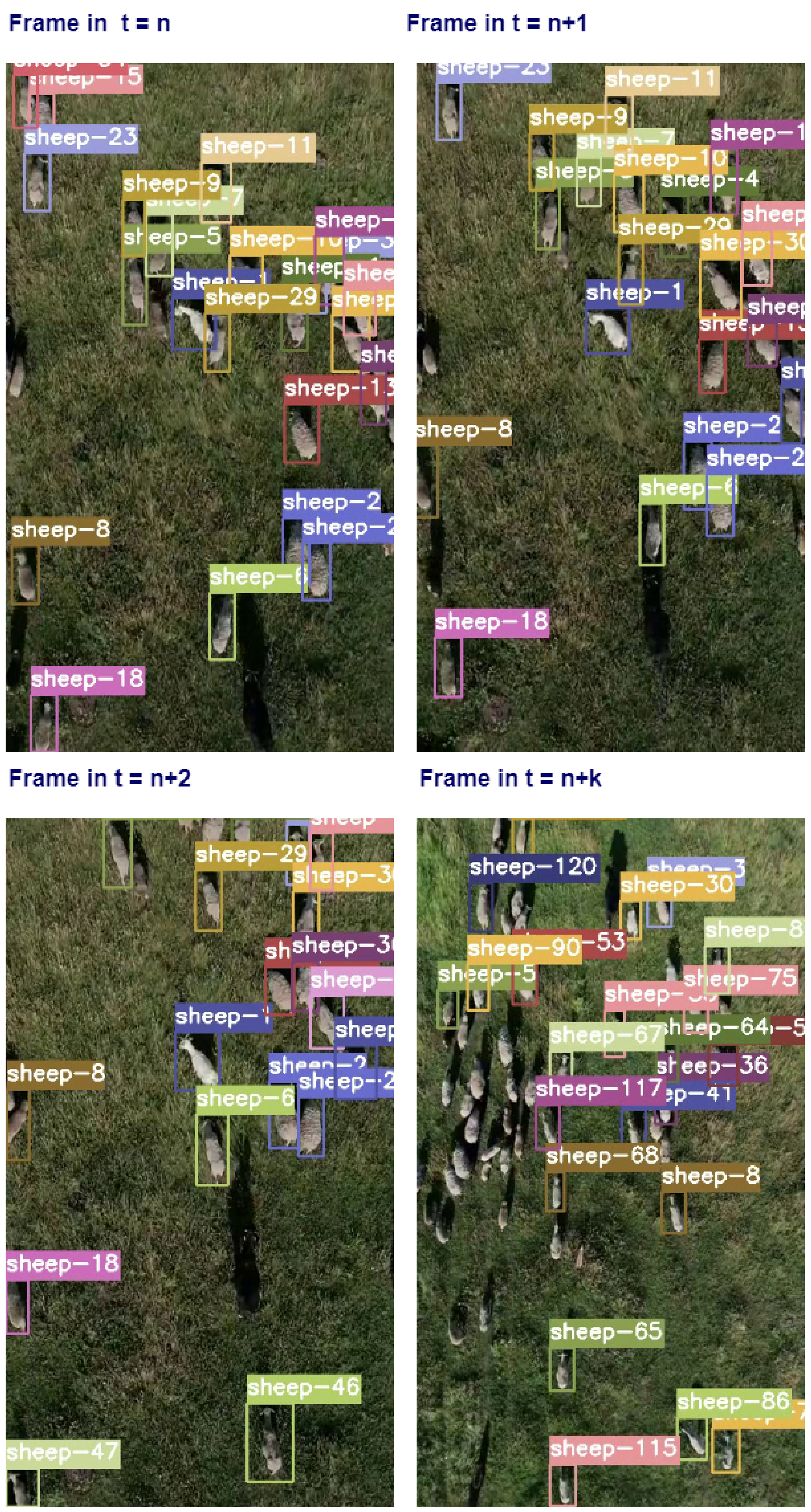


FIGURE 3.16 – The final result of tracking sheep in a group of frames in a chronological sequence.

3.6 Conclusion

This chapter presented the application of a sheep detection, tracking and counting system using deep learning. We explained the structure of the model and the modifications that we proposed to increase the efficiency of the architecture, but the idea was not completed, perhaps in the future, the data set (the way to collect it from the Internet, then purify it and extract images from the videos, then we labeled it).
The training and the difficulties we faced with the continuous interruption and finding a solution to the latter and the result was 71% in the mAP and 16.1274% in the avg loss function, and the testing process. Other than that we mentioned the results of the gate counting method that prompted us to introduce an intermediate stage of tracking and the obvious improvements that were made to the results of the experiment.

CONCLUSION

The goal of this work was to study the problem of Object Detection, Tracking and Counting and implement an intelligent system whose job is to count sheep in images captured from the sky, It can be expanded to other classes in the future. In this context, we presented the different state of the art tools for such a task. We explained deep learning and its use for our problem.

Based on the experiments presented in the third chapter, we concluded that using the state of the art YOLOv4 network was the best approach smarter.

We also concluded that the training data for Object detection must be prepossessed and labeled carefully because it plays a big role in the performance of the model.

Another conclusion is that we can definitely inspire from the one stage detectors if we want to improve the model's inference speed. We achieved a satisfactory result. But it is not enough to go to the counting stage directly, after research, we found that the best intermediate stage is tracking using Deep SORT algorithm carefully selected. As numbers we get 71% mAP on the train with 16.1274% as loss, this is in detection. By transgression, we will have the same results in tracking.

Our main plan was to work with real-life local data from local farms using a real drone of our own design, however, that became a challenge for us because of the lack of resources. Eventually, we relied on the web as an alternative to gather video data and extract frames to use for training our detector.

As perspectives, we plan to create the system that we intended to create in the first place. With an included drone that captures a stream of images that get sent to a processing center to do the inference. All while projecting the process on an interface for the user.

BIBLIOGRAPHIE

[1] Détection et suivi de la densité des personnes perçues dans les foules. In *Conférence internationale 2011 sur la vision par ordinateur*, pages 2423–2430.

[2] Yoshua Bengio Alain Tapp. Introduction a la apprentissage profond. 2019.

[3] Md Zahangir Alom, Tarek M Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C Van Essen, Abdul AS Awwal, and Vijayan K Asari. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3) :292, 2019.

[4] William Andrew, Colin Greatwood, and Tilo Burghardt. Visual localisation and individual identification of holstein friesian cattle via deep learning. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2850–2859, 2017.

[5] William Andrew, Sion Hannuna, Neill Campbell, and Tilo Burghardt. Automatic individual holstein friesian cattle identification via selective local coat pattern matching in rgb-d imagery. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 484–488. IEEE, 2016.

[6] Carlos Arteta, Victor Lempitsky, J Alison Noble, and Andrew Zisserman. Interactive object counting. In *European conference on computer vision*, pages 504–518. Springer, 2014.

[7] Mu Li Aston Zhang, Zachary C. Lipton and Alexander J. Smola. *Dive into Deep Learning Release 0.16.3*. Springer, Apr 23, 2021.

[8] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019.

[9] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.

[10] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4 : Optimal speed and accuracy of object detection. `https://github.com/AlexeyAB/darknet`, 2020.

[11] Antoni B Chan, Zhang-Sheng John Liang, and Nuno Vasconcelos. Privacy preserving crowd monitoring : Counting people without people models or tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2008.

[12] Ke Chen, gang Gong, Tao Xiang, and Chen Change Loy. Cumulative attribute space for age and crowd density estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2467–2474, 2013.

[13] Ke Chen, Chen Change Loy, Shaogang Gong, and Tony Xiang. Feature mining for localised crowd counting. In *Bmvc*, volume 1, page 3, 2012.

[14] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation : Encoder-decoder approaches. *arXiv preprint arXiv :1409.1259*, 2014.

[15] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv :1406.1078*, 2014.

[16] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[17] Li Deng. Three classes of deep learning architectures and their applications : a tutorial survey. *APSIPA transactions on signal and information processing*, 2012.

[18] David Exner, Erich Bruns, Daniel Kurz, Anselm Grundhöfer, and Oliver Bimber. Fast and robust camshift tracking. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 9–16. IEEE, 2010.

[19] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3038–3046, 2017.

[20] Yuhang Gan, Shucheng You, Zhengyu Luo, Ke Liu, Tao Zhang, and Lei Du. Object detection in remote sensing images with mask r-cnn. In *Journal of Physics : Conference Series*, volume 1673, page 012040. IOP Publishing, 2020.

[21] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[22] RB Girshick. Fast r-cnn. corr, abs/1504.08083, 2015.

[23] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[24] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[25] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *European conference on computer vision*, pages 297–312. Springer, 2014.

[26] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. Deep residual learning. *Image Recognition*, 2015.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[29] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7) :1527–1554, 2006.

[30] Sepp Hochreiter. Ja1 4 rgen schmidhuber (1997). "long short-term memory". *Neural Computation*, 9(8).

[31] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8) :1735–1780, 1997.

[32] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8) :1735–1780, 11 1997.

[33] Kristina Host, Marina Ivasic-Kos, and Miran Pobar. Tracking handball players with the deepsort algorithm. In *ICPRAM*, pages 593–599, 2020.

[34] Meng-Ru Hsieh, Yen-Liang Lin, and Winston H Hsu. Drone-based object counting by spatially regularized regional proposal network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4145–4153, 2017.

[35] JJ Ilopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA*, 79 :2554, 1982.

[36] Xiaoyue Jiang, Abdenour Hadid, Yanwei Pang, Eric Granger, and Xiaoyi Feng. *Deep Learning in object detection and recognition*. Springer, 2019.

[37] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[38] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. T-cnn : Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10) :2896–2907, 2017.

[39] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[40] Henry J Kelley. Gradient theory of optimal flight paths. *Ars Journal*, 30(10) :947–954, 1960.

[41] Dan Kong, Douglas Gray, and Hai Tao. A viewpoint invariant approach for crowd counting. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 1187–1190. IEEE, 2006.

[42] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25 :1097–1105, 2012.

[43] Harold W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2) :83–97, March 1955.

[44] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features : Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE, 2006.

[45] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10) :1995, 1995.

[46] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324, 1998.

[47] Victor Lempitsky and Andrew Zisserman. Learning to count objects in images. *Advances in neural information processing systems*, 23 :1324–1332, 2010.

[48] Junxiao Li and Ziao Wu. The application of yolov4 and a new pedestrian clustering algorithm to implement social distance monitoring during the covid-19 pandemic. In *Journal of Physics : Conference Series*, volume 1865, page 042019. IOP Publishing, 2021.

[49] Zhe Lin and Larry S Davis. Shape-based human detection and segmentation via hierarchical part-template matching. *IEEE transactions on pattern analysis and machine intelligence*, 32(4) :604–618, 2010.

[50] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd : Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[51] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[52] Mélissande Machefer, François Lemarchand, Virginie Bonnefond, Alasdair Hitchins, and Panagiotis Sidiropoulos. Refitting strategy for plant counting and sizing in uav imagery. *Remote Sensing*, 12(18) :3015, 2020.

[53] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2 :49–55, 1936.

[54] Pooja Mahto, Priyamm Garg, Pranav Seth, and J Panda. Refining yolov4 for vehicle detection. *International Journal of Advanced Research in Engineering and Technology (IJARET)*, 11(5), 2020.

[55] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4) :115–133, 1943.

[56] Matiur Rahman Minar and Jibon Naher. Recent advances in deep learning : An overview. *arXiv preprint arXiv :1807.08169*, 2018.

[57] Dimitris Mourtzis and John Angelopoulos. An intelligent framework for modelling and simulation of artificial neural networks (anns) based on augmented reality. *International Journal of Advanced Manufacturing Technology*, 111, 11 2020.

[58] T Nathan Mundhenk, Goran Konjevod, Wesam A Sakla, and Kofi Boakye. A large contextual dataset for classification, detection and counting of cars with deep learning. In *European Conference on Computer Vision*, pages 785–800. Springer, 2016.

[59] Giang Nguyen, Stefan Dlugolinsky, Martin Bobák, Viet Tran, Álvaro López García, Ignacio Heredia, Peter Malík, and Ladislav Hluchỳ. Machine learning and deep learning frameworks and libraries for large-scale data mining : a survey. *Artificial Intelligence Review*, 52(1) :77–124, 2019.

[60] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions : Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv :1811.03378*, 2018.

[61] Patrick Ott and Mark Everingham. Shared parts for deformable part-based models. In *CVPR 2011*, pages 1513–1520. IEEE, 2011.

[62] Mansi Parikh, Miral Patel, and Dulari Bhatt. Animal detection using template matching algorithm. *International Journal of Research in Modern Engineering and Emerging Technology*, 1(3) :26–32, 2013.

[63] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

[64] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010.

[65] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once : Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[66] Joseph Redmon and Ali Farhadi. Yolo9000 : better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[67] Joseph Redmon and Ali Farhadi. Yolov3 : An incremental improvement. *arXiv preprint arXiv :1804.02767*, 2018.

[68] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn : Towards real-time object detection with region proposal networks. *arXiv preprint arXiv :1506.01497*, 2015.

[69] Frank Rosenblatt. The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6) :386, 1958.

[70] Farah Sarwar, Anthony Griffin, Priyadharsini Periasamy, Kurt Portas, and Jim Law. Detecting and counting sheep with a convolutional neural network. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[71] Wen Shao, Rei Kawakami, Ryota Yoshihashi, Shaodi You, Hidemichi Kawase, and Takeshi Naemura. Cattle detection and counting in uav images based on convolutional neural networks. *International Journal of Remote Sensing*, 41(1) :31–52, 2020.

[72] Xingda Sun, He Hao, Yuan Liu, Yuanyuan Zhao, Yimeng Wang, and Ye Du. Research on the application of yolov4 in power inspection. In *IOP Conference Series : Earth and Environmental Science*, volume 693, page 012038. IOP Publishing, 2021.

[73] JAK Suykens, Lukas Lukas, Paul Van Dooren, Bart De Moor, Joos Vandewalle, et al. Least squares support vector machine classifiers : a large scale algorithm. In *European Conference on Circuit Theory and Design, ECCTD*, volume 99, pages 839–842. Citeseer, 1999.

[74] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[75] Andrew W Trask. *Grokking deep learning.* Manning Publications, 2019.

[76] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2) :154–171, 2013.

[77] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[78] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. Mots : Multi-object tracking and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7942–7951, 2019.

[79] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-YOLOv4 : Scaling cross stage partial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13029–13038, June 2021.

[80] Qi Wang, Junyu Gao, Wei Lin, and Yuan Yuan. Learning from synthetic data for crowd counting in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[81] Xiaoyu Wang, Tony X Han, and Shuicheng Yan. An hog-lbp human detector with partial occlusion handling. In *2009 IEEE 12th international conference on computer vision*, pages 32–39. IEEE, 2009.

[82] Bernard Widrow and Marcian E Hoff. Adaptive switching circuits. Technical report, Stanford Univ Ca Stanford Electsronics Labs, 1960.

[83] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[84] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649, 2017.

[85] Dihua Wu, Shuaichao Lv, Mei Jiang, and Huaibo Song. Using channel pruning-based yolo v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Computers and Electronics in Agriculture*, 178 :105742, 2020.

[86] Qi Wu, Chunhua Shen, Peng Wang, Anthony Dick, and Anton Van Den Hengel. Image captioning and visual question answering based on attributes and external knowledge. *IEEE transactions on pattern analysis and machine intelligence*, 40(6) :1367–1381, 2017.

[87] Beibei Xu, Wensheng Wang, Greg Falzon, Paul Kwan, Leifeng Guo, Guipeng Chen, Amy Tait, and Derek Schneider. Automated cattle counting using mask r-cnn in quadcopter vision system. *Computers and Electronics in Agriculture*, 171 :105300, 2020.

[88] 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.

[89] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 833–841, 2015.

[90] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years : A survey. *arXiv preprint arXiv :1905.05055*, 2019.