2018

Beef Cattle Instance Segmentation Using Mask R-Convolutional Neural Network

Mohammad Danish
Technological University Dublin, Ireland

Beef Cattle Instance Segmentation Using Mask R-Convolutional Neural Network

Mohammad Danish

A dissertation submitted in the partial fulfilment of the requirements of Dublin Institute of Technology for the degree of M.Sc. in Computing (Data Analytics)

September 2018

Declaration

I certify that this dissertation which I now submit for examination for the award of MSc. in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institutes guidelines for ethics in research.

Signed: …….

Mohammad Danish

Date: 04th September 2018

Abstract

Maintaining the cattle farm along with the wellbeing of every heifer has been the major concern in dairy farm. A robust system is required which can tackle the problem of continuous monitoring of cows. the computer vision techniques provide a new way to understand the challenges related to the identification and welfare of the cows.

This paper presents a state-of-art instance segmentation mask RCNN algorithm to train and build a model on a very challenging cow dataset that is captured during the winter season. The dataset poses many challenges such as overlapping of cows, partial occlusion, similarity between cows and background, and bad lightening. An attempt is made to improve the accuracy of the segmenter and the performance is measured after fine tuning the baseline model. The experiment result shows that fine tuning the mask RCNN algorithm helps in significantly improving the accuracy of instance segmentation of cows. this work is a contribution towards the real time monitoring of cows in cattle farm environment with the purpose of behavioural analysis of the cattle.

Keywords: *Cattle behaviour analysis, Deep learning, Object detection, Instance segmentation, Mask RCNN Algorithm*

Acknowledgement

I would like to thank my supervisor **Dr. Robert John Ross** for his expert advice, valuable suggestion and constant encouragement and recommendation during the course of this research project.

I would also like to acknowledge all my **Dublin Institute of Technology academic staff** especially all my professors for their help, kindness, support and knowledge.

I would also like express my sincere gratitude towards **Aram Ter-Sarkisov** for his early support in implementing the technical stuff of thesis work.

Last and most importantly, I am eternally thankful to my brother **Mohammad Shadab** for his love, support and encouragement to pursue my masters in the area of my interest.

Table of Contents

List of Figures

List of Table

List of Acronyms

AP	Average Precision
AUC	Area Under Curve
CNN	Convolutional Neural Network
DBN	Deep Belief network
FP	False Positive
FN	False Negative
FCN	Fully Connected Network
FPN	Feature Pyramid Network
GLCOM	Gray Level Co-Occurrence Matrix
GPU	Graphical Processing Unit
IOU	Intersection Over Union
LF	Livestock Farming
MAP	Mean Average Precision
MLP	Multilayer perceptron
MS-COCO	Microsoft Common Object in Context
NMX	Non-Max Suppression
RBM	Restricted Boltzmann Machine
RCNN	Region Based Convolutional Neural Network
RELU	Rectified Linear Unit Layer
RNN	Recurrent Neural Network
ROI	Region of Interest

RPN	Region Proposal Network
SDAE	Stacked Denoising Auto Encoder
TP	True Positive
VGG	Visual Geometry Group
VIA	VGG Image Annotator
VOC	Visual Object Class

Chapter 1

Introduction

1.1 Background

In the recent years, the population explosion has leads to the increase demand in the consumption of milk and meat of beef cattle and therefore it has produced exponential impact on the development and maintenance of dairy sector. To address the issue, certain measures are required where the welfare and wellbeing of individual cattle could be supported along with the cost of maintenance and this could be achieved through intensive farming (Harmans et al., 2003). Maintaining such a large farm require tremendous management task including best strategy for health of cattle and welfare to assure high input-output ratio with low cost of maintenance (Miguel-Pacheco et al., 2014; Rutten et al., 2013).

With the price of diary product varying across the countries, the marginal difference between the farms which try to gain higher profit and economical successful farms is very small. To manage such ideal and sustainable farms without affecting the wellbeing of animals, the farmers required to pay much of their attention on continuous workflow planning as their workload increases (Gradin, 2015; Halachmi et al., 2000). As a result, the farmers (caretaker, veterinarians) have limited time for proper continuous monitoring for each cattle (von Keyserlingk et al., 2009).

Due to overall complexity of day to day farm activity, visual assessment of individual cows becomes inefficient and it requires large investment of time (Busse et al., 2015). This is a situation where application of computer vision problem can play a crucial role for tracking the activities of each cows by finding the spatial distribution of cattle. This could provide farmer with the valuable information which could help in detecting the social behaviour and health concern of individual.

These days Computer based community has witness a tremendous amount of advancement in making the machine to interpret and react to the environment. Some of

example of these areas are human pose detection (Ionescu, Ionescu, Gadea, & Islam, 2011), Face recognition on mobile phone (Freier, 2011) and self-driving car (Turk, Morgenthaler, Gremban, & Marra, 1987). Following these trends, one of the major challenge that yet to come is enable the machine to automatically detect each object in its surrounding accurately such as human beings.

Over the past few years computer vision community has achieved very good results on object detection and semantic segmentation. The convolutional neural network has played a vital role in these areas. The advancement in the object detection and semantic (class) segmentation problems is driven by the powerful algorithm such as Faster RCNN (Ren, He, Girshick, & Sun, 2015) and Convolutional neural network (Shelhamer, Long, & Darrell, 2016). Recently, in computer vision-based problem, Instance segmentation has gained much attention after the introduction of new algorithm called Mask RCNN. While the semantic segmentation algorithm outlines the class of object at pixelwise level. However, it does not differentiate between the objects that are related to the same class. On the contrary, Instance segmentation finds the mask representation of each object in the image. In this task, the output of the algorithm can find the precise location, spatial extent and classes of all objects in the image that are part of the scene. For example, shown in figure 1.1, if the original image has 8 chairs, it is not enough to

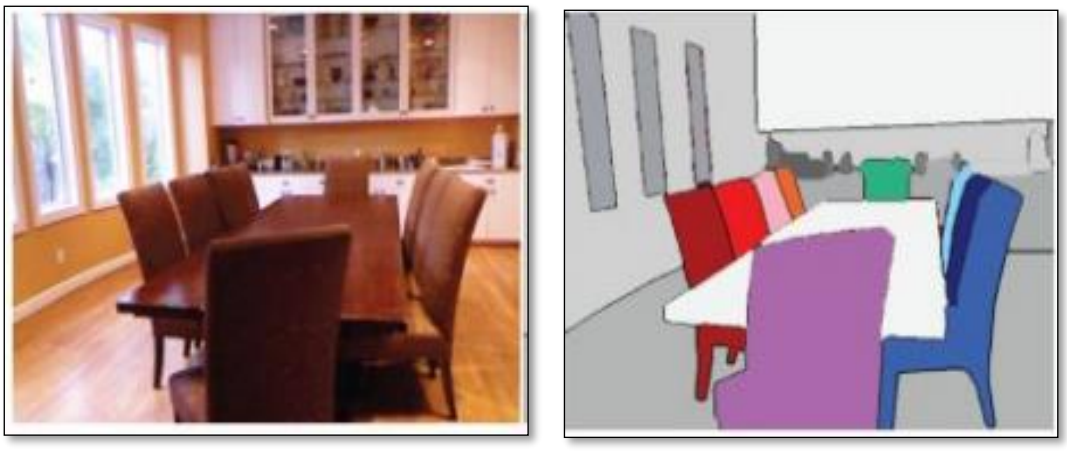

Figure 1.1: the output(right) of instance segmenter for given input(left) image.

(Silberman et al., 2014).

simply represents the pixels of the chair class. It is important to depict which pixels belong to each chair in the chair class that help in more understanding of the objects in the scene.

Traditionally, the use of different object detection systems for cattle tracking and monitoring has been applied in the field livestock farming (LF). These solutions have helped in segmenting the animals and learning the behaviour of animal (Porto et al., 2015). The algorithm such as CNN, semantic segmentation able to achieve good result in evaluating the complex scene evaluation and analysis of dairy barn environment. Applying Instance segmentation using Mask RCNN algorithm has potential to achieve good results in real time monitoring of the cattle farming that could help in early lameness detection and maintain the cost of the farm.

This thesis is a work for such challenging image analysis task. The work is a contribution towards real time monitoring of the cattle farming that help in detecting the early lameness in cows, maintaining the cost of cattle. It uses state of the arts pretrained CNNs model which attempt to provide the instance segmentation of the cows.

1.2 Research Problem

The aim of the project is to improve the accuracy of the state of the art Mask RCNN deep learning algorithm for the Instance segmentation of cows. The sample data contains frames of cow which was collected from the raw video that was recorded in the winter season with a camera that was installed at a fixed angle for monitoring purpose. The images of each frame are the feature which will be used for the instance segmentation of cows. The Mask RCNN model for instance segmentation will be built and later is will be used to compare the performance of pretrained model and the performance of our model with the same cow dataset. Mean Average precision (mAP) will be used for evaluation of the segmenter empirically. The research question that will answered from this thesis work can be stated as:

"*Are off the shelf semantic segmentation algorithms powerful and robust enough to be used on novel indoor data of cow with bad lightening and without face tracing?".*

The outcomes of this thesis will act as a proof of the concept for the use of Mask RCNN algorithm for detecting the instance of cows in the feedlot for behaviour analysis.

1.3 Research Objectives

The research objective of this study is to determine whether on fine tuning the off-the -shelf instance segmentation Mask RCNN algorithm on indoor cow dataset produces a significant improvement in the classification accuracy in terms of mean average precision. To achieve results, few experiment will be performed which is described below:

- Explore the existing works to analyse the computer vision problem and its application on the instance segmentation of the objects and perform a comprehensive analysis.

- Select the sample dataset from the feedlot containing the images of cows with challenges like bad lightening, occlusion and overlapping.

- Perform data preparation like object labelling, normalisation and image resizing.
- Build instance segmentation model on our prepared dataset using the state of art Mask RCNN algorithm.
- Compare our model mAP with the existing pretrained weights used on MS COCO dataset to analyse our results.

1.4 Research Methodologies

The study is focussed on the comparison of two models using the Mask RCNN algorithms on our challenging cow dataset and therefore it comes under the secondary research. The experiment will be carried out using the existing cow dataset and no new data is gathered for this purpose. As part of this existing research, literature review was carried out for the object detection algorithms, cow behaviour analysis, and image preparation to get the comprehensive idea of the project.

The research work follows the quantitative (Epidemiological) methodology and it is empirical in nature. The experiment is carried to get the results and then it will verify with given hypothesis. Experiment results are then evaluated to check the performance of the classifier by comparing the accuracy (mAP) of our model with the existing

pretrained model. On the basis of the result, whether are model accuracy is better than the existing one, hence it is an inductive experiment.

1.5 Scope and limitation

This project is focussed on the instance segmentation of cows using the Mask RCNN algorithm of the cow dataset prepared from the feedlot having bad lightening and partial occlusion among cows. the image dataset contains 25 frames and each frame has 10 cows to create flexible and state of art algorithms that can be used to detect multiple objects with real implementation of crowded area of varying illumination. Deep learning Mask RCNN will be used to build the models using the cow dataset to perform the instance segmentation of cows. the trained model will be then compared with existing pretrained model to discuss how they differ in terms of their performance and accuracy (mAP).

In terms of limitation, this works contains small sample size, only 25 images with 10 objects per image was used to perform the experiment. Also, the original dimension of image is not changed, therefore, the algorithm takes more time to train the model.

Furthermore, the result may be produced better if each object in the image is cropped and its corresponding mask is used for the training. This research only targeted one algorithm Mask Region based convolutional neural network for segmentation, other algorithm could be used that could provide the better result. Also, this instance segmentation is only for the cows, it could have been used for other animal's farms which are operated at one place such pig, horse etc.

1.6 Document outline

The outline of the project is given below, with total of six chapters which is further divided into subsection.

- Chapter 1 (Introduction) was devoted to the introduction of instance segmentation and background about cow's behaviour issue. It also covers the problem statements being solved along with overview of related work in the

computer vision problem. It describes the research methodology, scope and limitation of research.

- Chapter 2 (Literature Review) highlights the concept of state-of-the-art related computer vision problem for the purpose of filling gaps in the research to propose the research question for this thesis. In this chapter foundation of the computer vision problem called convolutional neural network, and detailed explanation of what are the object detection algorithms like semantic segmentation and instance segmentation are provided.

- Chapter 3 (Experiment design and methodology) will explain the design of the experiment with explanation of each steps performed. The complete detail of data preparation that is being used will be described in this chapter. Further, it also explains the list of techniques proposed to implement as part of the research.

- Chapter 4 (Implementation and results) gives the concrete explanation of the implementation of the experiment. The result of training network is also described here.

- Chapter 5 (Discussion and finding) will gives the detailed analysis and result of the experiment and based on the result a decision regarding the acceptance and rejection of proposed hypothesis will be made. This chapters also outlines the strength and weakness of the project.

- Chapter 6 (Conclusion) will summarize the working and finding of the research undertaken during this thesis work which includes problem definition, network design, experiment setup and evaluation of the finding and limitation for further work.

Chapter 2

Literature Review

The chapter provides a detailed review of relevant literature about the application of computer vision problem in the filed cattle farming. Various object detection techniques and their short coming is described here. The review is broadly classified into three sections, Cattle farming Analysis, Deep learning and Object detection as shown in figure 2.1.

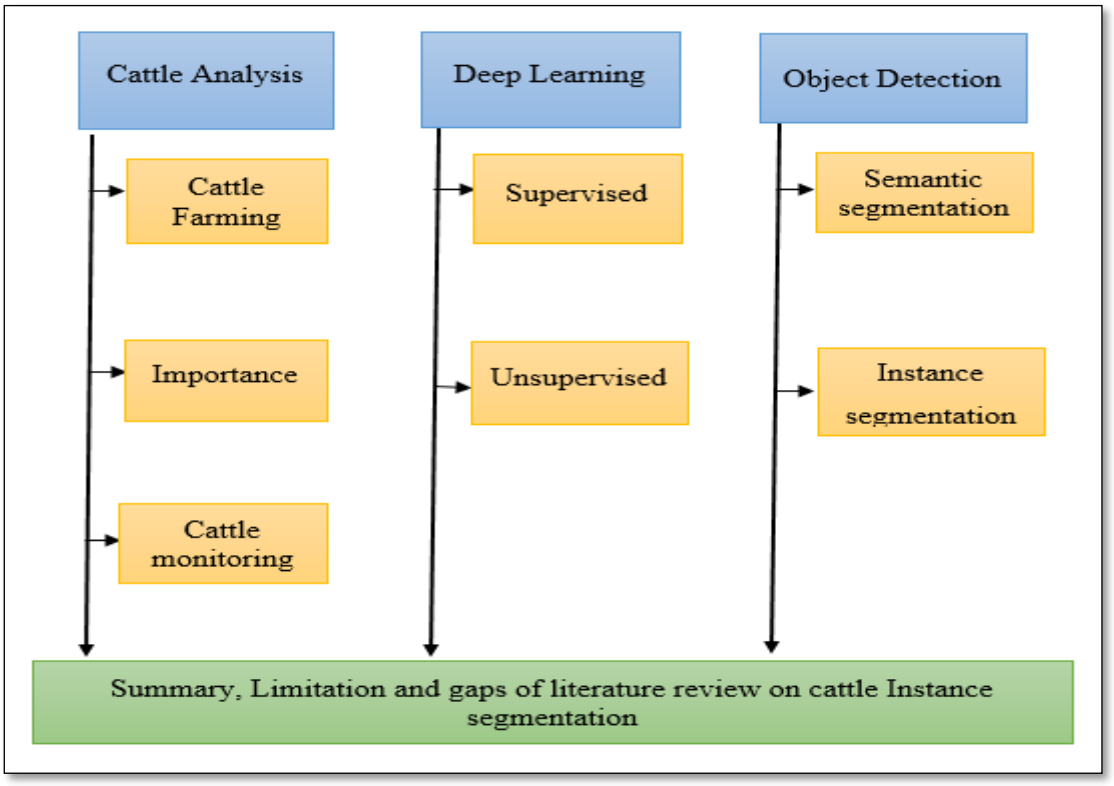

Figure 2.1 Literature Review Design

Section 2.1 describes state of the art computer vision algorithms for the cow behavioural analysis in cattle farming. This give general theory about the cattle farming, importance of monitoring, various computer vision techniques to detect the cattle and its behaviour. The next section 2.2 is all about the image analysis where baseline algorithm like convolutional neural network and its application is defined. In the next section 2.3 the state of art object detection algorithm is given which is further divided into semantic segmentation and Instance segmentation.

In the last section 2.4, reflection of detailed analysis of research paper around cow detection algorithm is provided. The analysis will help in finding the limitation and gaps in the existing works that will serve as motivation for the proposed research question.

2.1 Cattle farming Analysis

2.1.1 Cattle farming

Cows (also called dairy cattle) has been known and used for various purpose since the early human ages. Ever since the evolution of cattle farming, there is continuous changes in the behavioural pattern of cow, how they interact with the other animals, cow farmer, and environment such as building structure (McTavish et al., 2013).

Dairy animals are diurnal creatures, and the vision is their predominant sense, they are too able to recognize long and short wavelength colours and additionally depend more on an impression of moving instead of stationary objects (Adamczyk et al., 2015). One of other animal-typical qualities impacting the dairy animals' behaviour in various circumstances is the capacity to segregate between individual under various conditions (Coulon et al., 2011). It clarifies, mostly, the complexity of social collaborations and hierarchical structure inside groups of dairy cows (Kiley & Plain., 1983). There is logical proof (Taylor and Davis, 1998) recommending that cattle could utilize already stored mental pictures from prior social experiences and partner them with associate individual, influencing their goals and future interactions. These factors, and in addition some different parameters (e.g. breed, stage of lactation), require increased level of planning for maintaining the complex structure of farm for the wellbeing of the cattle functioning (Barkema et al., 2015).

2.1.2 Importance of automatic cattle monitoring

With the explosion in the population over last few decades, the need for the milk and meat has increased exponentially, which has led to the 10-folded increment and development of the diary sector. Also, the climate change which impacted the environmental transition need to be addressed, as in the cattle farming sustainability (Geers & Madec, 2006). To cater this issue, some means are required that could help in maintain the health and wellbeing of the cows as well cost of maintaining the farms

(Hermans et al., 2003). These big farms require the enormous attentions for administration routines and systems around cattle wellbeing and welfare to guarantee the most significant yield proportion at the least cost possible (Miguel-Pacheco et al., 2014). With the rapid development in the field of cow's dairy farm, free style (loose housing) farming areas has become the favourite choice for housing the cows along with routine management task, but it does not always look at the natural behaviour of animals. The critical analysis of grouping and interaction between the cattle is very important from sustainable production perspective as well as monitoring the wellbeing and the health of the cows (Phillips., 2002).

As per recent reports (Barkema et al., 2015), as the size of the dairy farm is increasing, the result is that large number of animals required daily caretaking. As day to day farm work incorporates a wide range of viewpoints, the ideal opportunity for observing the cattle and finding those of need with extra care has drastically decreased, which could result in diseases being unnoticed until next stages (Barkema et al., 2015). Therefore, continuous monitoring of animals in real time and assurance of early detection of disease could help in proving the better sustainable production without affecting the health of the cows.

Considering the complexity of daily work (advanced management such as feeding, caretaking of individual) involve in the of dairy farm, the need for visual assessment of individual animal by the farmer becomes inefficient which requires large amount of time. Therefore, the need for robust identification of animals became very big issue for the production and the performance of the farming. the first attempt for automatic monitoring was made with help of computer vision approach at livestock in late 1980 (Marchant, 1988). Ever since, with development of new hardware and technology in the field of visual world, this filed is getting more and more attention. However, there is still a very big challenge in identifying and differentiating between the animals with mixed and non-homogeneous background. This problem leads to the slow development of proper monitoring for the commercial farming. Therefore, we need a new object detection technology which could eliminate these problems and improve the quality of monitoring.

2.1.3 Computer vision techniques for animal identification.

In the recent years, the use of various computer vision-based approach for the welfare of animal, monitoring 24x7, and tracking the animals in dairy farm is developing within an area of precision livestock farming (PLF) research. There are solutions available which are suited for segmenting the animals in the barn from the background and monitoring the position (siting, lying, standing, eating) of each cattle (Porto et al., 2015). When these computer-based systems combine with the machine learning approach, it is possible to create a method which are capable of evaluating the complex scenario such as daily farm (Simonyan and Zisserman, 2015). These algorithms can help in multifactorial analysis of the dairy environment.

To monitor feedlot cattle behaviour and the interaction between them and farmers, one should able to judge and quantify the performed interaction in a continuous and reliable manner (Cangar et al., 2008). The manual analysis of the recorded video and focal observation are common methods which are now used for the analysis of cows. But these techniques are not very reliable as it very time consuming and requires utmost attention of the caretaker. Also, these systems hugely demand the skill set of the caretaker performing the annotation and interpretation of behaviour (Haidet et al., 2009). Another issue with system is that farmer or caretaker should be able to easily identify the cattle in the group of cattle with bad lightening and partial occlusion.

In the past decades, several techniques were proposed to identify and track the animals in the dairy farm. The WIFI, RFID, Bluetooth based system (Awad, 2016), GPS based product. Among all the product, the product which gained much popularity is RFID based product due to certain advantage over other product. The advantage includes huge storage capability, long battery life, affordability and scalability. However, RFID based system still requires lot of manual setting like marking animals with Tags, rules and infrastructure integration into one system (Busse et al., 2015). Therefore, considering all these problems and with increasing size of the farms, there is need for the flexible and robust system capable of performing individual tracking and identification (Banhazi and Tscharke, 2016).

Currently deep learning approach is emerging as the computer vision-based solution for identification and tracking of the animals. It has emerged as one of the most

powerful tool for feature extraction and representation of the individual animals. The different layer of the deep learning acts as the feature extraction which helps in recognising the complex data and varying animal features. The state of art deep learning approach and framework is serving as the effective solution in cattle recognition system by learning different feature.

2.2 Deep Learning

Deep learning algorithms are the part of Machine Learning that teaches computers to learn itself just as humans learns. It is inspired by the architecture and functioning of the visual vortex where each layer receives input signal from the layer below it, then it transforms the representation and propagates to the above layer. The general deep neural network consists of one input layer followed by more than 2 hidden layers and the output layer. Deep Neural network is called deep because the depth of the network (layer) is often more than that of conventional neural network which is sometime called as shallow network. The single layer network is sufficient to estimate any function. As the number of the layer is increased, the efficiency of task is also increased. It is because the first hidden layer extract low level feature and the second layer extract the high-level feature from previous layer. The same idea has been extended in the deep learning paper (Schmidhuber, 2015). The general deep learning structure is given in the figure 2.2

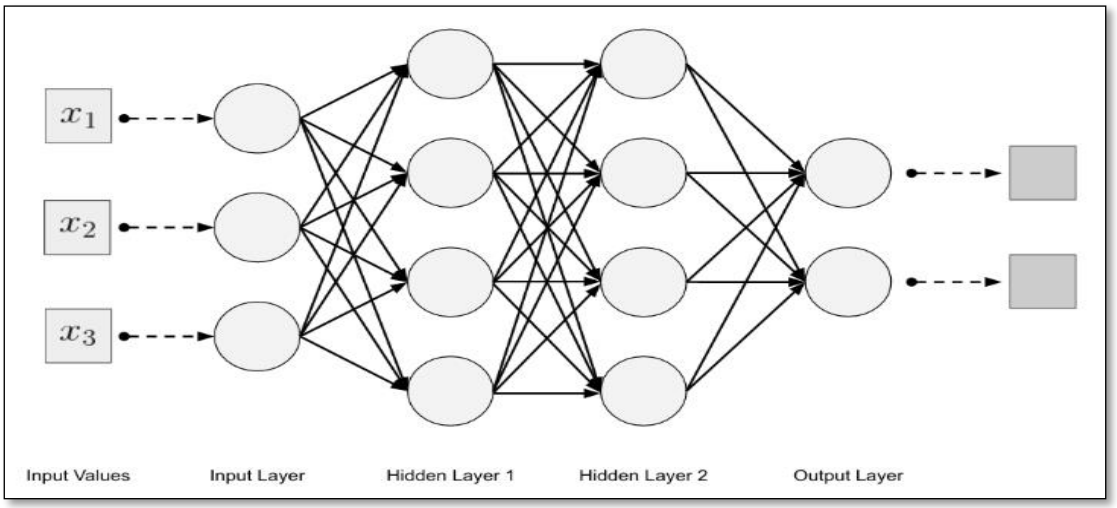

Figure 2.2 General architecture of deep neural network

(Soniya, Paul, & Singh, 2015)

Deep learning network has a gained its popularity in the many fields. Some of the application of deep learning are object recognition, speech recognition, text analysis, medical science, theoretical science. Also, it is used by the big companies like google, Facebook, Twitter and others to provide various services to their customers.

Just like machine learning, the learning in deep learning can be classified as supervised and unsupervised learning. The section discussed about both the technology of deep learning with its application.

2.2.1 Unsupervised learning

Unsupervised learning works towards the improvement of models that are capable of extracting significant and high-level representation from high-dimensional unlabelled data. This is inspired by the visual cortex which requires small volume of labelled data.

One of the popular unsupervised deep learning algorithm which allow the learning procedure for nonlinear features is the Deep Belief network (DBNs) (Salakhutdinov, 2015). It is built by stacking the several Restricted Boltzmann Machine (RBMs). Another unsupervised deep neural network is deep autoencoder. It is basically extract the complex feature with the help of multiple layer. This Algorithm is not straight forward, as deep neural network is dependent on the weight initialisation method and poor initialisation of weight can degrade the performance of network. In addition to this problem (Shin et al., 2013), auto encoder also suffers from propagating the gradient in the backward propagation algorithm. Therefore, to solve this problem of weight initialisation, (Adachi, 2014) introduced a new algorithm which learns feature of one layer at a time and this process is also called pretraining. This concept is based on RBM.

Deep neural network can also be utilised for dimensionality reduction of the given data. For this reason, auto encoders (Salakhutdinov & Hinton, 2009) have been proved to be successful in wide variety of applications such as image and document retrieval. An auto encoder as shown in figure 2.3 is deep unsupervised neural network in which the input data is same as the target data. Auto network is comprised of an “encoder” which transform the input data into low dimensional code and “decoder converts the low dimensional code back to data. This process of constructing data from code also requires minimizing the error between original data and its construction.

Additionality, the encoder part can serve as good feature extractor in an unsupervised learning. For example, Stacked Denoising autoencoder (SDAE) is used for feature extractor in various classification problem. The experiment performed in (Vincent et al., 2010) showed that SDAE with higher level of noise in data forced the model to extract high level feature.

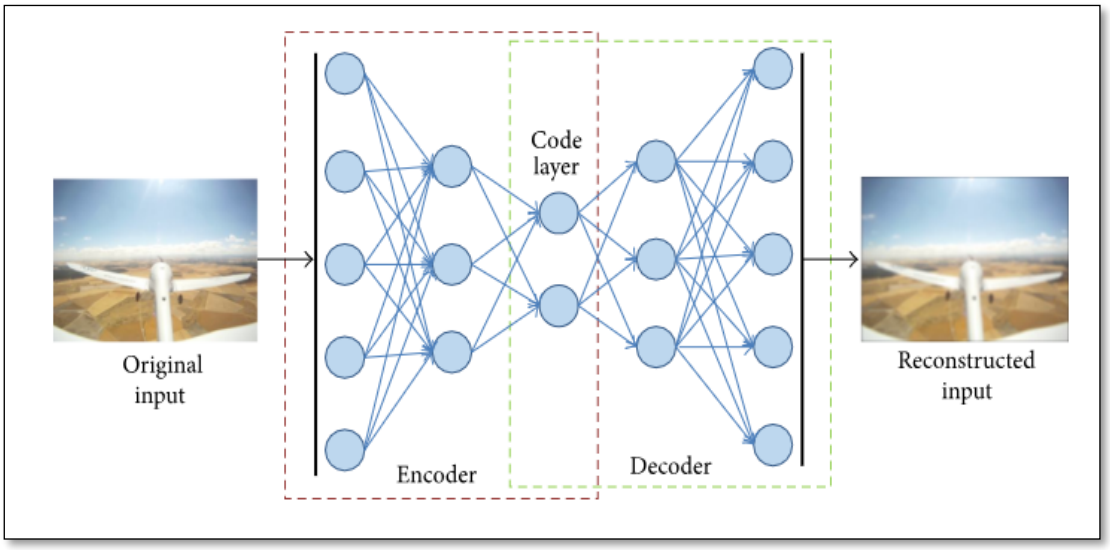

Figure 2.3 Deep auto encoder, "Encoder "layer converts the data to low dimensional code while "Decoder" reconstruct data from code

(Adrian, Carlos, Alejandro, & Pascual, 2017)

2.2.2 Supervised learning

Supervised learning algorithm functions by working on how to associate an input data with the output data given a training set of input and output. The most common type of supervised learning used in deep learning are feed forward network, Convolutional neural network (CNN) which is the slight variation of feed forward, Recurrent Neural network (RNN), Long short-term memory model which is variation of RNN.

The most common Feed Forward Neural Network is known as Multilayer Perceptron (MLPs). Its works as function approximator: for given value of sample x with n features, the trained model is expected to produce an output value or category y that shows consistency with input and output value on given training set. The hypothesis

function is usually made by stacking up several hidden layers and it is activated to get the desired result. These layers are made up of neurons whose activation at time for given input vector $x \in \mathbb{R}^n$ is given by the equation:

$$a_\theta(x) = g\left(\theta^T x\right) \tag{2.1}$$

Where g is the activation function that is nonlinear and θ is a vector of n weights.

One of major supervised learning algorithm is Convolutional Neural Network (CNN) when input data is image. Another machine learning algorithm does not scale well when it comes to the task of image. When input data is images or coloured images, the number of connection and learnable parameter increases exponentially. For example, if one trying to train a model in which input image is of size 800x600 coloured image and assuming the first layer has 1000 neurons, then the for the first layer only the number of parameter will be more than 1.3 billion. Therefore, it becomes very important to reduce the number of connections at each layer when the task is working with image. CNN helps by working on the image by reducing the number of neurons at each layer by only connecting to the subset image's pixel at each time. This is accomplished by using the convolutional neural network in which image is convolved with series of learnable filter.

For convolutional layers, all the neurons are forced to have the same weights and connects only to the small parts of the image (e.g. 3x3 pixel area). Also, they are organised in spatial grid throughout the image such that one connects slightly different location from the last one. At each layer of convolution, there are several learning filters which convolves around the image to produce the several output images which is also called "feature map". The CNN layer also incorporates Rectified linear unit layer (RELU) which operates at pixel-wise manner in their inputs. It calculates the output layer as

$$\mathrm{ReLU}(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} , \tag{2.2}$$

Where X is the value of the input pixel. Another important layer which operates with the Convolutional layer is max pooling. The max pooling layer works independently in

the feature map of the input by reducing the spatial size. This reduction in size is done by keeping only the maximum value found in the region. The filters are able to learn the low-level feature like edge, horizontal line, vertical line and high-level feature such as object depending on the architecture of the network. The generic structure of the convolutional neural network is shown in figure 2.4.

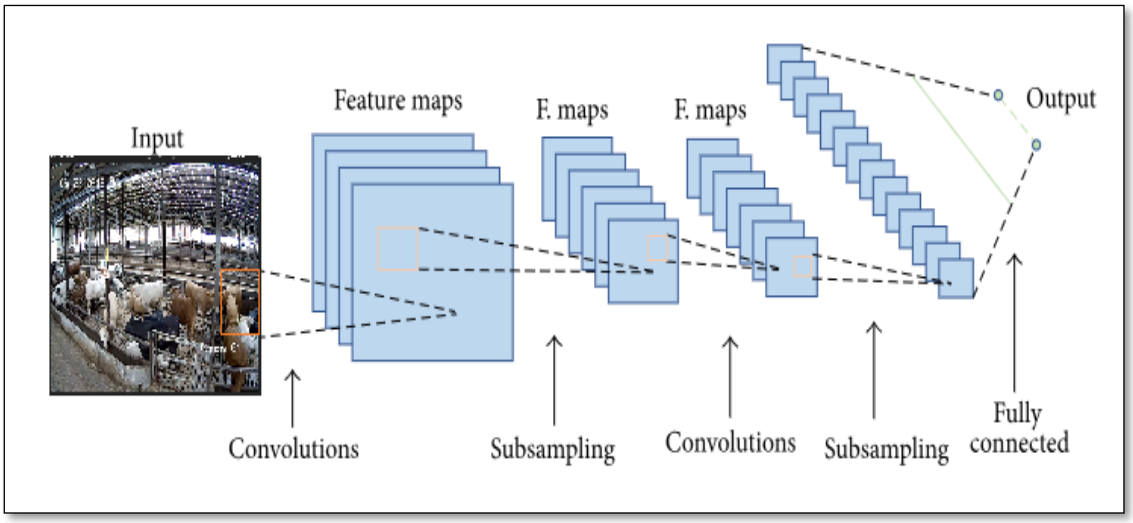

Figure 2.4: The generic structure convolutional neural network and sub sampling layer

Convolutional Neural network has played a vital role in cattle classification. Lot of work has been done in this field. In the paper by (Santoni, Sensuse, Arymurthy, & Fanany, 2015) proposed a Gray level Co-occurrence matrix (GLCOM) CNN for different cattle race classification. It used 5 fives types of different cow's race to train the model. the model with GLSM image produced the better result than CNN. But the model was not able to classify the cow efficiently when used with non-homogeneous background. Similarly, CNN was used as the watchdog for monitoring the interaction of cows (Ardo, Guzhva, Nilsson, & Herlin, 2018). The model was evaluated using the average precision.

2.3 Object Detection

One of the major key problem in image analysis is the object detection problem. This problem is the task of predicting the location and spatial approximation of object of a set of predefined class, while also assigning the correct class to them. The location and spatial approximation of the object is encoded as the tightest rectangle that is also called bounding box. A sample output of an algorithm of object detection is shown in figure 2.5. The example is taken from the challenge PASCAL VOC paper.

Figure 2.5 Example of object detection Pascal VOC challenge

(Everingham et al., 2015)

Although many approaches were used to solve the object detection problem, but the most recent one is region based convolutional neural network. These network works differently, it first makes use of low level pixel information to propose a large number of area which have higher probability of containing the objects (bounding box for prediction). These proposals are then feed to the feature extractor for instance ResNet 50 which produces a feature vector of each region.

The resultant vector is then fed to the series of fully connected layer (Burges, 2010) to compute the classification among the predefined class. This also computes the confidence score, which ranks the proposals so that subset of them can be returned. In addition to it, feature vector is also fed into regressor to improve the quality of bounding box proposal area.

In the recent Years, the region CNN based approach for object detection has now become the standard approach because of improved performance. These approaches are significantly faster than other approach making them a favourite choice for research community (Ustyuzhaninov, Michaelis, Brendel, & Bethge, 2018). Fast R-CNN algorithm has become one of the famous algorithm for performing object detection task. It has been used in several object detection challenges and its performance was much faster and accurate than region-based CNN (Ustyuzhaninov et al., 2015).

2.3.1 Semantic segmentation

The next big challenge after object detection in the world of computer vision problem is semantic segmentation of object. Semantic segmentation is the ability to segment unknown image into different object (e.g. person, ocean, car, cycle, cow, dog). Specifically, object detection will only perform the classification task that classify object with specific label such as person, horse, cow, car. Looking at bigger picture, semantic segmentation is high level task that paves the way for complete scene understanding. The importance of scene understanding in the computer vision problem can be seen from the fact that large number of application is nourishing knowledge from the imagery. Some of the applications are autonomous driving (Geiger, Lenz, & Urtasun, 2012), image search engine (Wan et al., 2014). These kinds of problem have also been addressed earlier using various computer vision techniques and machine learning algorithms. The K -means and thresholding techniques was applied to perform the cow segmentation by (Nahari, Jauhari, Hidayat, & Alfita, 2017) but it was only doing the semantic segmentation of one cow in the image. Similarly, other techniques were applied to perform the image segmentation, but they were not able to get the greater accuracy.

2.3.2 Instance segmentation

The instance segmentation problem can be thought of performing object detection with the semantic segmentation. This is very new field in the field of computer

vision problem and its job is to locate each object in the class, predict its class and also provide binary mask for each object in that class. As this is very active field in the research community, various approaches have been taken to solve this problem, some makes CNN based approach, other makes combination CNN and region-based approach. One of the algorithm that has gained much popularity is mask RCNN algorithm in the recent days. The mask RCNN is the part facebook AI research (He, Gkioxari, Dollár, & Girshick, 2017). It is an extension of the existing faster RCNN algorithm (Ren, He, Girshick, & Sun, 2015) in which there is additional layer which is able to produce the mask in addition to the bounding object. The instance segment is able to detect the object at per pixel level and this is only the difference between semantic segmentation and instance segmentation. The difference between them in shown in the figure 2.6

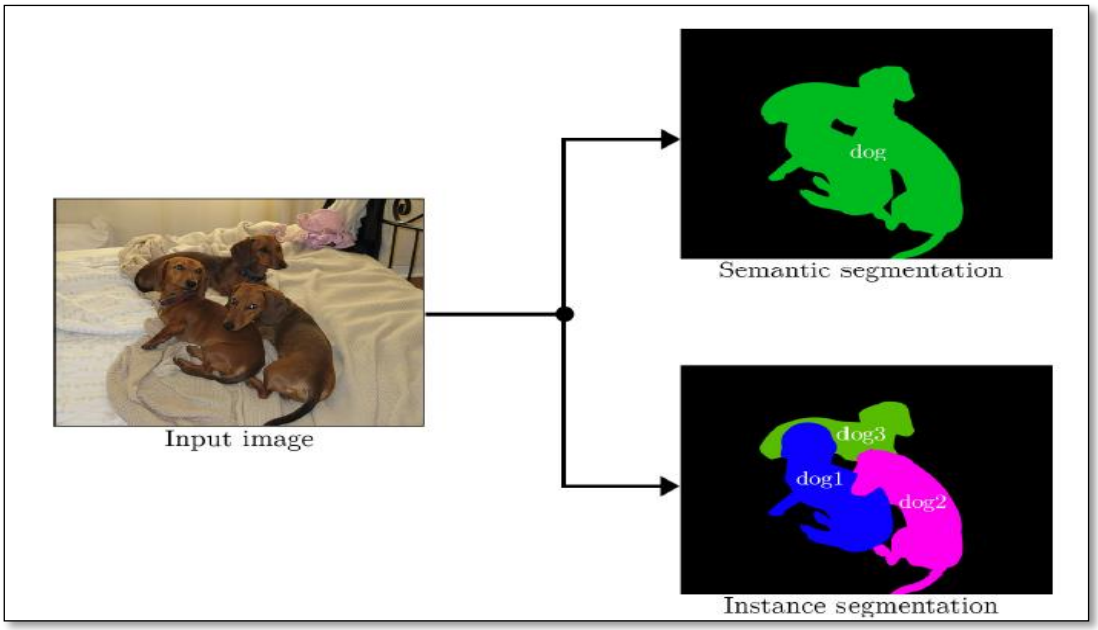

Figure 2.6 Illustration difference between the instance segmentation and semantic segmentation

2.4 Summary, limitation and gaps of literature

A detailed review of application of deep learning computer vision algorithm is studied as part of solving the problem of instance segmentation in the field of cattle farming. Initially, relevant literature about the cattle and its behaviour problem are

studied to gain knowledge about the problem of monitoring large number of cattle. Computer vision solution in the field of cattle automatic monitoring is also studied. In the deep learning section, emphasis is given to the image analysis, where object detection, semantic segmentation and instance segmentation are discussed to solve the object detection problem.

Semantic segmentation techniques such as instance segmentation need to be investigated in future in the field cattle monitoring by the researchers. Object detection algorithm such as Fully Connected layer need to be investigated further in the cattle detecting and monitoring (Busse et al., 2015). Also, Simonyan and Zisserman suggested that object detection algorithm such FCN can be preferred over the other techniques such as GPS based, RFID tags for monitoring and detecting the cattle. In further research by (Nagl et al., 2015), they implemented the deep neural network architecture in order to improve the accuracy of the model.

(Nahari et al., 2017) used k-means and thresholding technique to perform the semantic classification of cows. Region based CNN network is applied in the field of animal identification and achieved better accuracy by (Banhazi, & Tscharke, 2016). Animals identification such cows can be explored by the fully convolutional layer instance segmentation algorithm that uses concept of mask splitter by (Ter-Sarkisov, Ross, Kelleher, Earley, & Keane, 2018). The concept is based on the ground truth mask representation of cows for performing the instance segmentation. New algorithm Mask RCNN developed by Facebook AI research area (Ren, He, Girshick, & Sun, 2015) have been successfully applied in the filed various object detection problems.

The limitation and research gaps presented in this section can be addressed by the research question given as

"*Are off the shelf semantic segmentation algorithms powerful and robust enough to be used on novel indoor data of cow with bad lightening & without face tracing?"*.

The next sections will describe about the research design, implementation and evaluation of experiment to address the research question

Chapter 3

Experiment Design and methodology

This Chapter will provide elaborated plan and design for performing the instance segmentation task as proposed in the research question. The experiment will follow the standard approach for implementing the model. It has data preparation, Network architecture and evaluation metrics for performing the experiment. All the steps of implementation will be performed using python language using deep learning library such as TensorFlow and Keras. TensorFlow and Keras is open source machine learning, deep learning library for high performance numerical computation. Originally built by google research engineer is favourite choice for performing computer vision problem.

The aim of the thesis research is to build an instance segmentation model for cow's segmentation by fine tuning the state of the art Mask RCNN algorithm. The model is performed using cow dataset which is prepared and extracted from the video of cattle farm collected over the winter. Figure 3.1 gives the high-level design of the experiment which is followed by detailed information of the experiment.

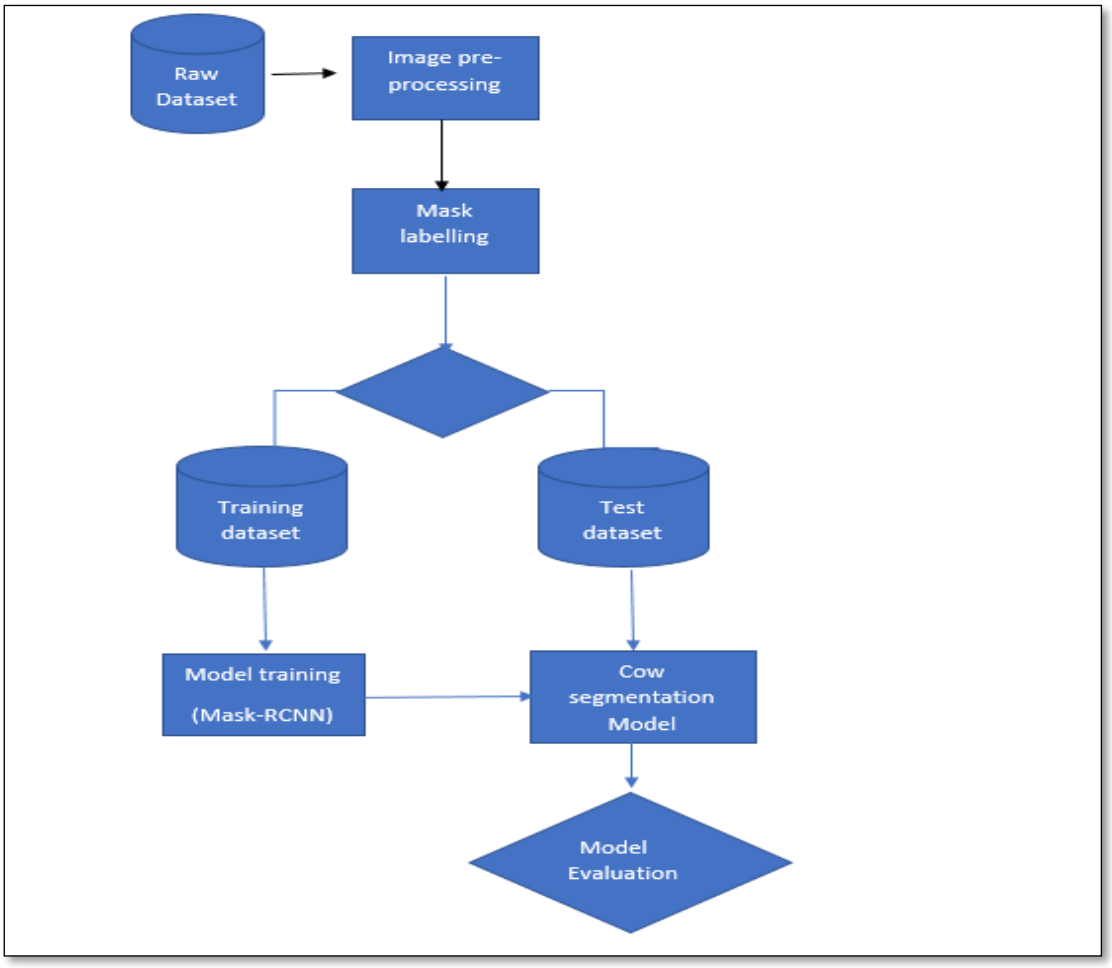

Figure 3.1 High level design of the Instance segmentation model

3.1 Business Understanding

The focus of the research is to improve the model accuracy of state of art mask RCNN deep learning model for the instance segmentation of the cows. In order to achieve this goal, baseline model is fine tuned to achieve greater mean average precision (mAP). Later the performance is evaluated and compared with baseline model to address the research questions. The given hypothesis is taken into consideration to address the research question.

"H0 Fine tuning the off-the-shelf instance segmentation Mask RCNN algorithm on indoor cow data produces a significant improvement in the classification accuracy in terms of mean average precision."

3.2 Data Understanding

3.2.1 Our Dataset

The dataset contains the information about the cow at the finishing feedlot for cattle over a period of two weeks. The dataset was prepared from the video which was recorded with CCTV camera fixed at angle towards the enclosures containing the heifers. Each enclosure contains 10 cows. for this research one of the enclosures is selected for the dataset construction. The dataset collected from the video is challenging than other state of art dataset such as Microsoft Common Object in Common (MS-COCO) and Pascal VOC dataset and therefore

1. The Object: the cows in the feedlot changes its position frequently and assume different pose. The shape of animals changes frequently when they move, lay down, walks, eat and groom other mates. Hence, the model needs much higher generalisation capability.
2. Similarity: It is nearly impossible to distinguish between two cows because of same colour, patch and other physical markings. Hence, it even very challenging for humans to distinguish between the cows.
3. Occlusion: the view of the cattle suffers from the partial occlusion due to positioning of camera.
4. Lightening: the construction of the farm facilitates both the artificial light and natural light. The natural light comes through the ventilation in the roof, therefore it had quite poor lightening. There is gap in the ventilation on the roof, light from these vents enters in the farm area which produces rectangular patches over the enclosures, along with the cows. this poses a challenge to the segmenter to generalise because they could treat shadow as the object.
5. Background: It is even very difficult for the advanced algorithm to detect cows with such background because the background is dark, noisy and poses similarity with different cows in terms of colour and patches such as grey black and white. In many cases it become very difficult to distinguish between cows and background.

The dataset constructed consists of 27 frames extracted from the one enclosure, making it total of 270 cattle of size 720x1280 size. The dataset is then randomly into training and validation datasets of size 190 and 80. Therefore the ration of training and validation is 70% and 30% respectively.

Example of one frame of dataset is shown in below figure 3.2.

Figure 3.2 Illustration of frame from our dataset for building the model

3.2.2 Benchmark dataset

The two-stocked (benchmark) dataset that most commonly used in object detection competition and research areas are Microsoft Common Objects in Context dataset, in short MS COCO (Lin et al., 2014) and Pascal Visual Object Classes, in short Pascal VOC (Everingham, Van, Williams, Winn, & Zisserman, 2010). The MS COCO 2017 is the latest version. It contains more than 250 thousand data of different settings, spatial location along with the ground truth mask generated by humans that is available for training and validation dataset. It covers more than 80 different classes. For the current thesis work, it only requires cow dataset which have total of 2071 images. The training dataset have 1986 images of cow while there are 87 images available for validation and testing. Similarly, Pascal VOC dataset have 64 images available for training and 71 images of cows are reserved for validation.

In this work, the model implementation and validation will mostly focus on self-prepared cow dataset, for testing purpose it will use both MS COCO Cow dataset and self-prepared dataset. Illustration of MS-CCOCO cow dataset is shown in figure 3.3

Figure 3.3 Example of cow image from the MS-COCO dataset (Lin et al., 2014)

3.3 Data Processing

Once the dataset is collected from the video, the next will be to annotate each object in the image. Segmenting each object in the image is time consuming and hard part. For this stage, Graphic annotation tool will be used to segment each cow in the image. The mask representation is then labelled for cows and background for training purpose. The mask representation of the image is shown in figure 3.4

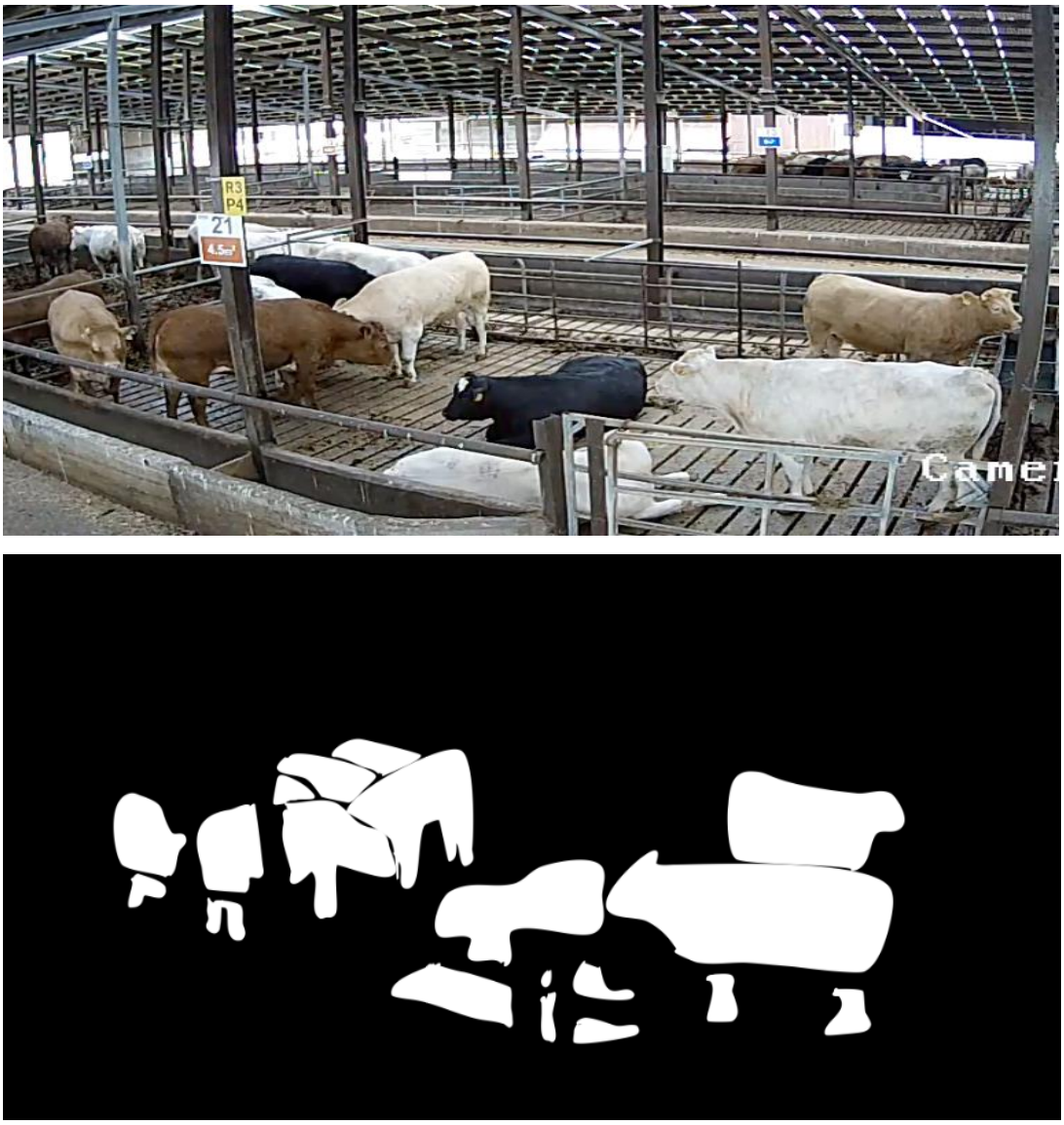

Figure 3.4 Illustration of Mask representation for the image

3.4 Network Architecture

This section describes detailed architecture of Mask RCNN architecture which is used for the segmentation of the cows. The current work is replication of the off the shelf mask RCNN algorithm. The current work implemented the same network architecture as defined in the Mask RCNN paper (Kaiming, Georgia, Piotr, & Ross, 2017). The Mask RCNN algorithm is simple and flexible framework for performing object instance segmentation. This algorithm effectively generates high quality mask for the input image. The abstract architecture of the network is shown in figure 3.1. Mask RCNN is a two-stage framework, the first stage scans the image and generates proposal that is also called region of interest (ROI), these are areas where objects are likely to be

present. The second stage classify the region and generates the bounding box and masks. At the high level the Mask RCNN consist of Feature pyramid network+ Backbone layer, followed by Region proposal network which generates positive region (object) and bounding box refinement. The mask is a convolutional network that takes positive regions from ROI and generates mask for them.

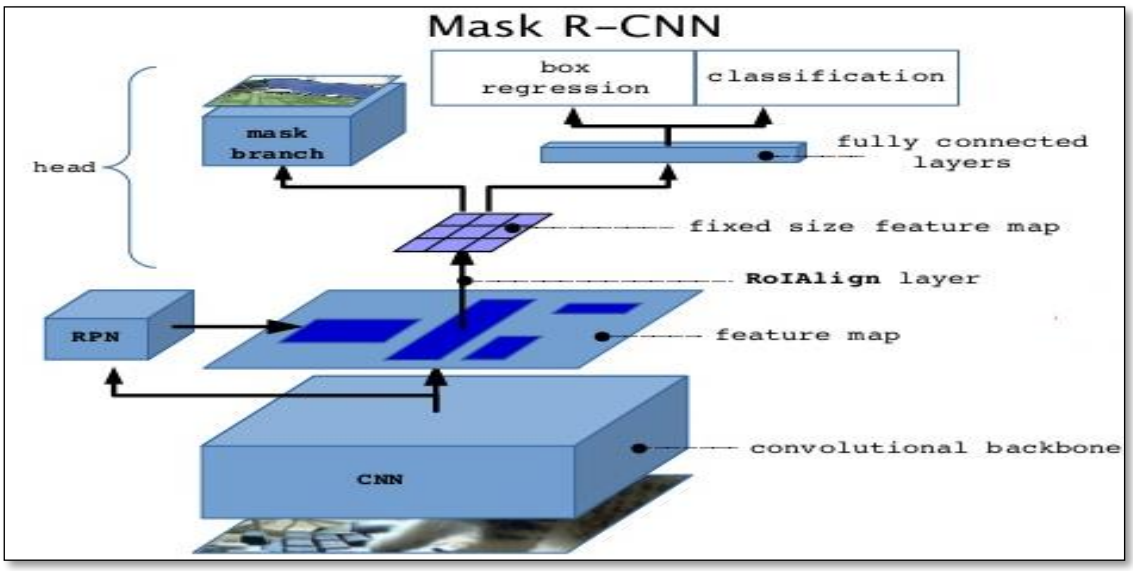

Figure 3.5: High-level network architecture of Mask RCNN algorithm (Kaiming, Georgia, Piotr, & Ross, 2017).

The various component of our Mask RCNN algorithm design and their functionality is given below.

1. Convolutional ResNet backbone and FPN: Residual Network (ResNet) (He, Gkioxari, Dollar, & Garshik, 2017) was introduced initially as CNN to perform image classification task. But it became popular choice for other deep learning task.

   Figure 3.6 gives the fundamental building block of ResNet architecture. In our model, ResNet101 (variant of ResNet) will serve as a basic CNN network of our network. It serves as a feature extractor. The early layer will detect the low-level feature such as edge and corners and deep layer will detect higher feature like cow in our case. This backbone layer processes the input image and outputs them into the feature map in the last layer. In the original paper RestNet101 (He et al.,

2017) is used in combination with Feature pyramid network (FPN) to serve as backbone layer.

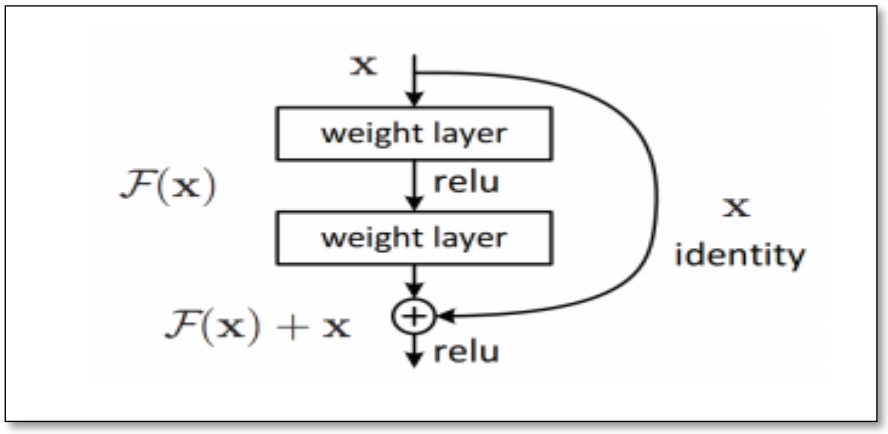

Figure 3.6: ResNet backbone schema adapted from (He et al., 2017)

2. Region proposal Network: the next RPN layer in Mask RCNN architecture is light weight neural network that scans the image in sliding windows fashion and finds the area which contains object. RPN scans over the extracted feature of backbone and it increases the efficiency and performance of the network. The RPN scan over the region are called anchors. There are more than 200k anchors for an image. The RPN generates two class from each anchor – anchor class and bounding box refinement. Anchor class have one of two class- foregrounds (containing cows) and background. While bounding box refines the anchor perfectly to fit the object better.

3. Non-max Suppression (NMS): Using RPN, the algorithm picks the top anchor that are likely to contain object (cows) and refine their location. If there are several anchors overlapping, the one with the highest foreground score will be selected and this process is known as NMS. After this the final Region of interest (ROI) is passed to the next stage.

4. ROI Pooling: Due to NMS, the size of the anchor has different size of feature maps, and this creates the problem for classifier. Therefore, ROI pooling helps in resizing the size of feature map into fixed size by cropping the part of the image.

5. Segmentation mask: the mask branch is convolutional network that takes the fixed size feature map and generates the mask for them.

During the inference mode, predicted mask is resized to the original size of ROI bounding box and provides the final mask for each object.

3.5 Performance Evaluation

Optimising the training and validation error of the mask RCNN subsystem can only quantify how well this subsystem is performing. To evaluate the performance of the whole model, a specific metric is used which is called Mean Average Precision (mAP).

It was first introduced in (Lin et al., 2014) which is an attempt to quantify how well our system is performing the task of instance segmentation based on the precision-recall curve of each class (only one class in our case i.e . Cow). First precision- recall curve is generated by performing the system evaluation and then Area under curve (AUC) is computed as average precision for that specific class. Finally, the Metric mAP is defined as the Average of average precision of all the predefined class which is called as Mean Average Precision (mAP^r).

It is mandatory to match the ground truth annotated object in an image with the predicted instances in order to generate the precision-recall curves for the algorithm. The generated instance from the model matches with the ground truth instance, if both has the same class and the metric Intersection over Union (IoU) is greater than the predefined value. In object detection challenge, metric IoU (He et al., 2017) is used to measure how much the overlap between the predicted value and ground truth value. The equation for IoU is given in below

$$IoU = \frac{\text{area of overlap}}{\text{area of union}} \qquad 3.1$$

If the instance generated by model matches with several ground truth values, then one with the highest score of IoU is considered. for this model evaluation, the threshold IoU selected for performance evaluation is taken from the MS-COCO (Lin et al., 2014) challenge and IoU value is from 0.5 to 0.95. metric mAP notation with

different threshold is given as mAP@X, where X is the threshold value used for computing the metric.

The ground truth instance which matches with generated instance is removed from the consideration so that no other generated instance can be matched with that object. Therefore, it penalises the repeated instance (it is considered as false positive). the precision-recall are calculated after all the matches are found for the image.

Precision is defined as how many instances generated by the model are correct and computed as

$$P = \frac{tp}{tp + fp} \tag{3.2}$$

On the other hand, recall measures how many instances, among all the ground truth are correct and computed as

$$R = \frac{tp}{tp + fn} \ , \tag{3.3}$$

Where, tp (True Positive) is the instances with the matching ground truth object.

fp (False Positive) is the instance with no matching ground truth object.

fn (False Negative) is ground truth object having no matching instances.

Finally, the average precision will be calculated, once the precision-recall points are generated using the various threshold IoU value. The formula for calculating the Average precision is given below.

$$AP = \sum_{n=1}^{N} [R(n) - R(n-1)] \cdot \max_{\tilde{n} \geq n} P(\tilde{n}) \tag{3.4}$$

Where, N is the number of precision-recall point generated, P(n) and R(n) are the precision and recall with lowest n'th recall respectively.

Once our model is ready, metric mAP for baseline model i.e. using the pretrained weights of MS COCO dataset and our trained model with our dataset will be evaluated

and compared to see the performance of each. The final step determines whether our trained model perform better than state of art baseline model and based on that we will accept or reject our hypothesis.

3.6 Strength and limitation

This part describes the strength and gaps of our research design. Firstly, the state of art mask RCNN algorithm is used to perform the instance segmentation task. The design of mask RCNN algorithm is based on region-based CNN and it has proved to very effective in the field of instance segmentation. Unlike other algorithm such as Fully connected network (FCN), the mask algorithm has different layer such as backbone layer, FPN and fully connected layer which is able to predict the instances more efficiently. The architecture of mask RCNN has been used in many object detection challenges and it has achieved better results than other CNN algorithm. Therefore, this algorithm has higher power of predicting the instances effectively. The other benefits of using this design algorithm that it is modular in nature and it will create a little overhead in terms of system performance. In terms of evaluation, Average precision will be used with different threshold value ranging from 50% to 95% IoU and mean of all average precision will be used to evaluate the segmenter accuracy.

In terms of limitation, the model will be trained using the fewer number of images. The dataset contains 27 images making it not perfect for training, but due time constraint in annotating the image less number of image has been used. For computer vision problem, higher the number of image used for training the model, better is the chances of getting the results. the other major limitation is use of backbone layer, this research only uses ResNet layer as backbone. There are also proven backbone network architecture such as Alexnet, VGG-16, google net could have been used to see the performance, but it was not included in the mask RCNN design due to time constraint.

3.7 Summary of design

This section outlines the summary about the design and methodology for this research. It also describes about the strength and limitation of design.

The topic starts with the brief description about the dataset in which, it describes about our dataset and benchmark dataset. then it also describes about the challenge in

our dataset such as bad lightening, partial occlusion, similarity between objects and background. It then discussed how we going to process the data to make it suitable for the modelling. It also discusses about the tool graphic which will be used to label the cows in the image. For creating model our dataset will be divided into training and validation set of 19 and 8 images respectively. Then test data from our dataset and benchmark dataset is used to evaluate the performance.

Further, Mask RCNN model architecture is defined having the description of different layer which has main component such as ResNet and FPN backbone layer, RPN layer and the segmenter of fully connected layer for mask representation. Two models will be trained, the baseline model will be trained using the existing pretrained weights from the MS-COCO dataset and other will be finetuned to with our dataset to get better accuracy. In the end, model will be evaluated using the mAP metric at different threshold IoU and then it will be compared to with baselined to select the best model.

The next chapter gives the detailed of the practical implementation of the proposed method and design.

Chapter 4

Implementation and results

The main aim of this chapter is to give the complete implementation details of network architecture that was described in the last section. The layout of this chapter is similar to the last chapter. In addition, it also discusses about the result obtained.

4.1 Business Understanding

The aim of this chapter is to give the practical implementation of the design that was discussed in the last chapter along with result produced from the model. The section will cover the following topic:

1. Data pre-processing
2. Network Architecture
3. Hyper parameter setting
4. Deep learning software (TensorFlow, Keras & GPU)
5. Evaluation of the result
6. Comparing the result to answer the research question

4.2 Data Pre- processing

The dataset used for this thesis consist of 27 images of 720 x1280 pixels that contain 10 cows in each frame. The dataset is imbalanced with regard to the number of pixel representing cows and background. Adjustment in the raw dataset is made to increase the performance of the training and inference pipeline. The datasets are normalised between 0 and 255 and it is then converted into 8-bit format. The training validation and test sets are stored in the array and dumped into pickle format and save into the disk. The normalised illustration of the data from our dataset is shown in figure 4.1

Figure 4.1 Illustration of data normalisation between 0 and 255 pixels

of the cow

To evaluate the object detection and image computer vision problem, image annotation is necessary (Lin et al., 2014). Basically, the image annotation consists of following attributes for every object in the image.

Class- class represent one of following for image collection like background, cycle, car, horse, cow, motorcycle, balloon, sofa, chair. But for our dataset we have only two classes Background and Cow

Ground truth Mask – the mask represents the axis and spatial position of each segment in the image.

Once the dataset has been collected, the next part is to annotate them, which is off course time consuming and hard part. There are lot publicly available tools to annotate the image. The popular tools which are used for annotation are

VIA- VGG image Annotator which stores the file in the JSON format,

label Me – One of the most known tool. The problem is that it bit slow, when performing zooming large image.

RectLabel – This tool is only compatible with MAC user.

COCO UI- This tool is used for the annotating the COCO dataset.

Graphic – This very simple and fast tool available on Apple iPad.

The tool which is used for annotating the mask is graphic tool due to its user-friendly interface and fast processing. This is the default tool for the iPad user. The total of 27 images were annotated which was divided into training and validation and testing set for implementing the model. Initially, it was bit slow to annotate each heifer in the image but the process of annotating the image manually became faster as more and more image were annotated. For storing the segmentation mask of each object, there is no any universally accepted format. Some of them store as polygon point, JPEG image. This task stores the original image as well as mask representation as PNG format. Once the Mask for each object in image is created, the next step is to label them, that is the value of mask for two class, the background class which is represented by 0 and foreground (cow) with 1 as ground truth mask. The detail flow chart of dataset preparation is shown in figure 2.1.

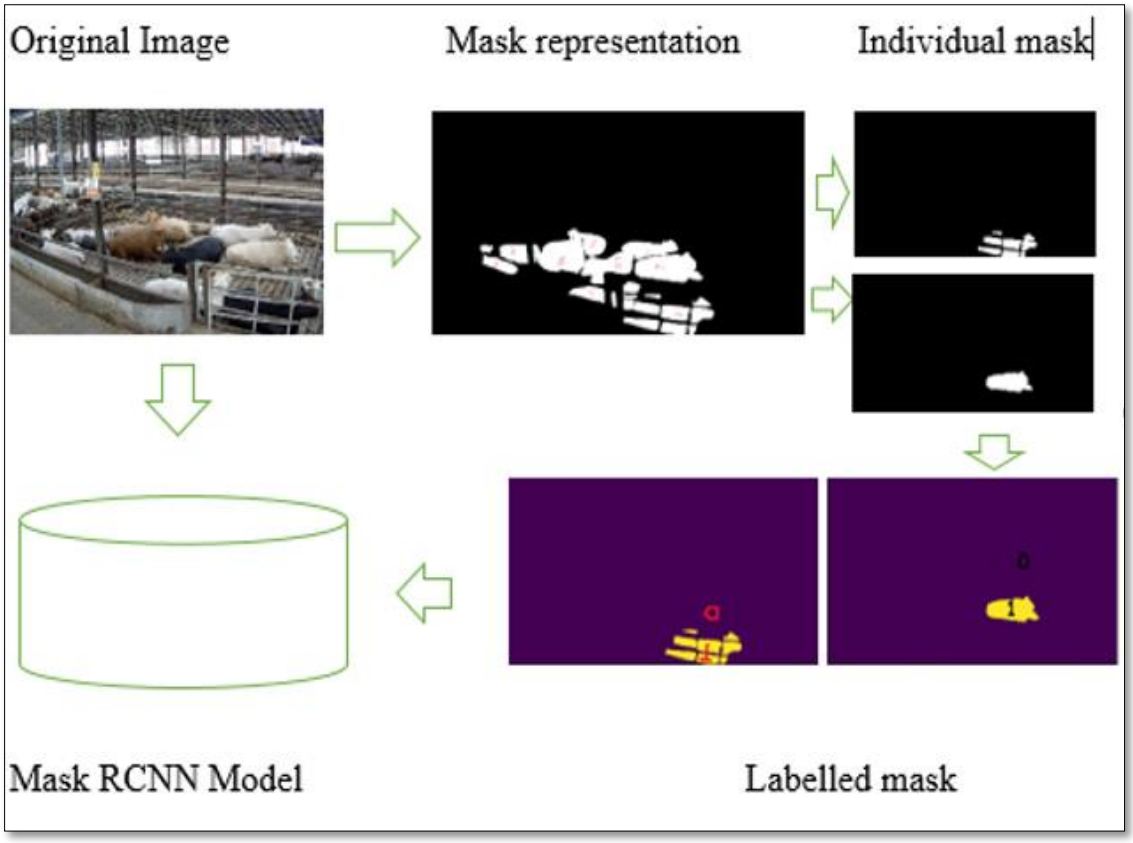

Figure 4.2: Flow chart of cow dataset construction for training the model.

Note: For shake of clarity only two cow's masks label were taken instead of 10 cows in the above flow chart.

Every image has dimension of 720x1280 and similarly the mask dimension is same as image dimension. All 10 objects ground truth mask of an image is then stacked together to form an array of dimension (720x1280x10) and then it is stored as pickle object with the same name as the image. Storing the pickle objects of stacked array with same name as image help in identifying the mask representation easily. The dataset is now ready for the training and evaluating the model. The next section discusses the network architecture and implementation of the Mask RCNN model.

4.3 Network architecture implementation

For the current work, our implementation uses FPN + ResNet101 backbone layer, that detect the features at both higher and lower level. The second stage is RPN which is light weight neural network that scans image in sliding windows fashion and finds the area that contains cows. the region which RPN scans over are called anchors which are boxes that distributed across the image. Each image contains more than 300k anchors of different sizes overlapping each other. the output from RPN is the class and bounding box regression. The class defines the foreground (likely to contains object) and background (no objects). While the bounding box regression finds the estimates of changes to be make in anchors position to fit the object better. In case of overlapping anchors, the one with highest foreground score is selected and with rest lower score will be discarded which is known non-max Suppression. Then the resulted anchor is the final region of interest (ROI) that is passed to the next stage. The ROI generated by RPN is run by this stage and it also generates bounding box regression and class. In our case,it generates two class cow and background. The background class which is discarded at this stage. Similar to RPN stage, it also proposes bounding box with more refinement in size and location with respect to object. In our implementation, crop and resize function is used to handle the varying ROI box size and convert them into fixed size.

Till now, the concept was part of Faster RCNN framework for object detection. The mask network is additional layer that is proposed by (Kaiming, Georgia, Piotr, & Ross, 2017). It takes positive anchors proposed by RPN and generates high quality mask. The generated mask is low resolution mask generally 28x28 pixels which stores more information than binary mask.

4.3.1 Hyper parameter setting

For the first part of training model 1, following network hyperparameter were set up for the model as shown in table 3.1. Due to memory and processing power constraints the training was performed on 5 training and 3 validation images. So, there was 83 cows including mask of each cows plus the original images.

Configuration	**Description**
ResNet Architecture	ResNet101(Backbone layer)
Learning Rate	0.001
Learning rate momentum	0.9
Image min dimension	512
Image max dimension	512
Detection min confidence	0.5
Number of Epoch	2
Steps per epoch	5000
Validation steps	5
Mask shape	28x28
Num classes	2 (cow and background)
Batch size	2

Table 4.1: Hyper parameter setting for first model using smaller number of training data.

For Training, the ROI is considered positive if it has Intersection over Union (IoU) with ground truth box of at least 0.5 else negative. during training we defined, multi-task loss on each ROI as

$$L = L_{cls} + L_{box} + L_{mask} \qquad 4.1$$

where L_{cls} and L_{box} describes the class loss and bounding box loss which is as per the original paper (Ren, He, Girshick, & Sun, 2015). L_{mask} is a pixel wise cross-entropy loss taken over sigmoid of the score map of good prediction.

The model 1 is trained using the 5-training data and 3 validation data by setting the above parameter. The trained model was then applied on the test data to see how our model is

able to predict the instance of cows. the graphical illustration of our model for the test data is shown in the figure 4.3

Figure 4.3 the original test data which our model need to perform the instance segmentation using Mask RCNN algorithm

Figure 4.4 Illustration of our model 1 performance on the above test data

Similarly, four different models are built with increased number of training and validation set. The number of training image used are 19 and for validation 9 images are used with different hyper parameter to enhance the performance (mAP) of the model. The implementation of all the model is performed using the deep learning library and the system specification is provided in the next section.

4.4 Deep learning software

All the work for the image segmentation task have been implemented using the open source deep learning framework such as TensorFlow and Keras.

Keras is open source high level API for neural network. It runs on the top of TensorFlow. The most important features of Keras are user friendliness, it supports modularity. The different module in this mask RCNN architecture such as neural network, cost function, activation function, regularisation is combined together to create model. Its other advantage is easy extensible and easy to use because it is written in python.

TensorFlow is another framework which is used for this thesis. It is very effective in performing the high-performance calculation. Additionally, the optimisation of code become very easy when using the TensorFlow.

In terms of computational environment, University high performance research server infrastructure was used for training and optimising the model. Adapt-01 GPU node was used for training the model. The hardware specification the node is given below.

Two 4-core Intel Xeon Processors @ 2.8 GHz

512 GB Ram

Dual Tesla K40 GPU

4 TB Storage

The next section describes about the evaluation of each model and the comparison is made with pretrained weights (baseline) of MS-COCO dataset.

4.5 Evaluation

In order to evaluate the model's performance, the metric mAP is needed for different threshold IoU. The thorough experiment was performed using Mask RCNN algorithm on our dataset. The experiment was carried using state off the art pretrained weight of MS COCO and our trained model. The results were reported using the standard COCO metrics mean Average precision mAP (Average over IoU) AP_{50}, AP_{75}, AP@0.5:0.95. The result was reported in mAP with different IoU. The standard value of IoU for reporting is AP@0.50, AP@0.70 and AP@0.5:0.95 with step size of 5. The main result of the five different models trained on our dataset and existing baseline model result on our test data is shown in table 4.1

Model	Backbone Network	AP@0.5	AP@.70	AP@0.5:0.95
Baseline model	ResNet101	0.502	0.442	0.325
Model 1	ResNet101	**0.524**	**0.460**	0.311
Model 2	ResNet101	0.492	0.425	0.309
Model 3	ResNet101	**0.598**	**0.511**	**0.331**
Model 4	ResNet101	**0.632**	**0.522**	**0.347**

Table 4.2: Results of mAP for baseline model and our model using Mask RCNN algorithm

The result of four different models obtained above are obtained by changing the hyper parameter setting given in the table 4.1. Out of four different models, the model 4 achieved the best result and the detail mAP for this model is shown in 4.3

AP@0.5	AP@0.55	AP@0.60	AP@0.65	AP@0.70	AP@0.75	AP@0.80	AP@0.85	AP@0.90	AP@ 0.95
0.632	.624	0.591	0.553	0.522	0.321	.161	0	0	0

Table 4.3: Results of mAP for model 4 at different IoU value

The complete result analysis and comparison is provided in the chapter 5.

4.6 Summary of Implementation

This section summarizes the practical implementation of Mask RCNN algorithm for the instance segmentation of the cows. the experiment was performed to answer the research question and the evaluation of model is done.

The experiment starts with data processing to make the data ready for building the instance segmentation model. The image was normalised between 0 to 255 pixels and the mask was created for each cow in the image using the graphic tool provided iPad. The data is then labelled as 0 and 1 representing the background and cows respectively. The complete flow diagram data processing was provided in the figure 4.2. the array data was then dumped as pickle format. It was then divided into training validation and test dataset to train and evaluate the model.

After data processing, training of model was done with different hyper parameter setting. The minimum and maximum size of image was set to 512x512 for modelling. Further, the total of 4 different modes was built by tweaking the hyper parameter.

In order to evaluate the model, mAP of the four model and result was provided in the tabular form. The existing baseline model was tested on our test data and its result is also provided in table 4.2. the illustration of trained model 1 is provided in the figure 4.4. the next chapter will discuss about the results and its comparison about the baseline model. It will also provide the practical illustration of how our model perform on our data. the decision about acceptance and rejection of hypothesis will also be made.

Chapter 5

Discussion and finding

This chapter provides in-depth evaluation of the result obtained from the algorithm. It also provides the visual analysis of the outputs, which illustrates success and failure cases of the model. The performance analysis of the models is compared with baseline model. Finally, the decision on the acceptance or rejection of hypothesis will be made and discussion about the experiment's strength and limitation is concluded.

5.1 Discussion

The goal of this research was to build an instance segmentation model and perform an experiment to fine tune the baseline model to achieve better performance on our challenging indoor dataset of cows. initially, pretrained weights from MS-COCO dataset was used to evaluate the model. The dataset preparation steps such as segmenting each cow in an image, normalising the image and labelling the object was performed. 4 different models were built by tweaking the hyper parameter in order to achieve better mAP. The results of the model (in mAP) is shown in table 4.2.

5.1.1 Comparison of models result

Four different models were build and the evaluation of the model was performed on the test data. The model 1 achieved a better AP@ 0.5 and AP@ 0.70 but overall the mean average precision (mAP) of the baseline model was better than by a margin of 4 %. On the other hands, the model 2 didn't perform better than baseline model and its average precision was less for all threshold value. This was least performing model. Again, it was fine-tuned with changing the hyper parameter setting and trained with more number epochs and validation steps and changing image dimension. The model 3 and model 4 get the better result than baseline model at different level of IoU as mentioned in the table 4.2. The best model after training with different parameter setting is achieved using model 4. The detail of performance mAP for this model is provided in the table 4.3. From this table it is clear that model has performed very well for the IoU

threshold of 50% to 70%. The model achieved mAP of 0.632 which 11% more than baseline model. Similarly, there is an increment of 6 % of mAP at 70% of IoU with baseline model. On the other hand, the model performs worst from 70 % to 90% of IoU. It has achieved mAP of zero from 85% IoU to 95% IoU.

5.1.2 Visual assessment of good and bad cases of the model

To try to gain the insights of how the baseline pretrained weights from MS-COCO dataset perform on our dataset, the new image from our dataset was input into this model. The visualisation of the baseline model is shown in the figure 5.1

Figure 5.1 Visualisation of the instance segmentation for the baseline model

From the above visualisation it can be seen that baseline model is able to perform instance segmentation quite well, but at the same time it doesn't able to detect some of cows in frame. The baseline model achieved a mAP of 0.325 at AP@ 05:0.95 IoU. few of the instances where baseline is not able to perform segmentation or badly performed is shown in the figure 5.2

Figure 5.2 cows which baseline model was not able to recognise properly or bad instance segmentation.

Some of the good and bad cases of our model 3 is shown in figure 5.3. compared to baseline model our trained model was able to perform better than baseline model

Figure 5.3 our trained model with good and bad segmentation of cows respectively

Although our model is also not able to completely perform the instance segmentation, but it has achieved better mAP than baseline model after fine tuning the model. The model has performed significantly well from 50% IoU to 70% IoU than baseline model, but at higher IoU after 70% to 85%, the baseline model has achieved marginally better result. Overall, our trained model was able to achieve better result.

5.1.3 Hypothesis Evaluation

This section will discuss about the hypothesis testing of the experiment. The hypothesis for the experiment that is to accept or rejected is given below:

"H0 Fine tuning the off-the-shelf instance segmentation Mask RCNN algorithm on indoor cow dataset produces a significant improvement in the classification accuracy in terms of Mean Average precision."

Experiment were carried out by fine tuning the Mask RCNN algorithm on the indoor cow dataset to achieve the better accuracy (mAP). After fine tuning the existing mask RCNN algorithm on our dataset, our model was able to achieve better mean average precision (mAP) than the baseline model. Thus, it can be concluded that our hypothesis cab be accepted as fine tuning the algorithm has achieved better result.

5.2 Strength and limitation

The contribution of deep learning instance segmentation techniques in cow behaviour pattern was studied as part of the thesis study. The experiment uses state-of-art Mask RCNN algorithm for performing the instance segmentation of cows. the baseline model based on mask RCNN algorithm which was trained on MS-COCO dataset, tested on our data and it was evaluated. Then new model was trained on our dataset in order fine tune the accuracy of model.

Performing training on our dataset was challenging as the dataset poses many challenges like similarity between the objects, partial occlusion, bad lightening, overlapping between the cows. Additionally, four different models were build using the different hyper parameter setting to fine the model. The model parameter like changing the image dimension, number of epochs and increasing image number have significant effect in increasing the accuracy of the model. To get the better result pretrained weights from MS-COCO dataset were used as the initial point and then training using these weights were performed on our dataset so that model optimisation doesn't suffer from mis localisation.

Data pre-processing technique such as normalisation of pixel was applied on the image using the python PIL to improve the result. the model was trained using training and validation so that models doesn't suffer from overfitting.

Lastly, the model is very simple and modular, and it can be altered with more training feature and instances. The use of model can also accommodate other cattle related problem.

One of the major limitation of the research is the use of lesser number of images. Only 27 images were used to train and validate the model. In deep learning scenario, the higher the number of images or feature used for the training the better the models perform. The experiment was performed by using ResNet backbone layer only, further research can be used by trying the other backbone layer such VGG-16, google Net to test for improvement in the result. Similarly, the tuning of the models has no effect on the higher-level threshold IoU and it has achieved very poor performance.

Another limitation for this research, it focusses only on the cattle diary of particular farm and data was only prepared from one enclosures. Therefore, model might have performed well when data from different sources were taken as it can learn more feature from the data.

Chapter 6

Conclusion

6.1 Research overview and contribution

This research formulated, built and evaluated the mask RCNN algorithm for performing the instance segmentation of the cows. this research is the contribution towards the behavioural analysis of the cattle such as early lameness, interaction and automatic monitoring of the cattle. Initially, the critical literature review is conducted which summarizes the various state-of-art computer vision algorithm to perform the segmentation task.

A thorough experiment was performed to build the model based on the deep learning mask RCNN algorithm. Different models were built to improve the mean average precision in performing the instance segmentation of the cows. At initial stage, pretrained weights of MS-COCO dataset used, which used a basis to improve our model by training with our challenging dataset. Four different models were built by changing the various hyper parameter such as number of epochs, training, validation set, image dimension in order to get better results. the network architecture used for this research was similar to the original mask RCNN paper (Kaiming, Georgia, Piotr, & Ross, 2017).

The architecture which was best rated using this mask RCNN algorithm was model 4. mAP was used to evaluate all the model. Then by comparing this metric, it was found that model 4 achieved the highest mAP@ 0.5: 0.95 IoU of 34.7%. All the models are implemented using TensorFlow and Keras framework. A visual assessment of the baseline model and trained model on our dataset is presented where it showcases how model is able to perform good instance segmentation as well some bad cases of segmentation.

The research is contribution towards continuous monitoring of the cattle which helps in detecting diseases such early lameness, lactation problem and interaction among each cow. This research can help in identifying each heifer in the farm by performing

instance segmentation. The future work can be done by tuning the model to achieve more accurate result.

6.2 Future work and recommendations

This project focusses on Mask RCNN algorithm to perform the instance segmentation task, however performance on other algorithm such Fully Connected Network with different backbone layer can be further compared to find the best model. From the experiment, it is found that tuning the hyper parameter such as number of iteration, epochs, image dimension, learning rate has significant impact the performance of model. The future work can involve tuning more hyper parameter such as image dimension, flipping image, momentum rate. Hence, model performance can be further enhanced.

The dataset used in this project is very limited and it is trained using 27 images only due to time constraint of preparing the data. Further work can be done by capturing more images from the video and use that images to build the model. This could have significant impact on the performance of the model. In the mask RCNN architecture implementation, ResNet backbone is used for the feature extraction, the future work can also involve changing this backbone layer with other deep neural architecture like VGG-16, Google Net and compare the result to see if they provide the better performance. For ROI, all positive anchor (area containing object) are resized to fixed size feature map, but instead of resizing to the fixed size, other techniques like binary interpolation (He, Gkioxari, Dollár, & Girshick, 2017) can be used and this adjusting parameter can produce a great impact on the result performance.

Further work can include to perform error analysis to optimise the model like plotting the precision-recall at different number of iteration. These techniques could help in analysing the model in clear manner and could help in avoiding overfitting. Additionally, instead of using our validation dataset, MS-COCO cow validation dataset (Lin et al., 2014) could be used which has variety of cow's image. This could help in better generalising the model.

References

Adrian Carrio, Carlos Sampedro, Alejandro Rodriguez-Ramos, and Pascual Campoy, "A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles," *Journal of Sensors*, vol. 2017, Article ID 3296874, 13 pages, 2017. https://doi.org/10.1155/2017/3296874.

Ardo, H., Guzhva, O., Nilsson, M., & Herlin, A. H. (2018). Convolutional neural network-based cow interaction watchdog. *IET Computer Vision*, *12*(2), 171–177. https://doi.org/10.1049/iet-cvi.2017.0077

Adamczyk, K., Górecka-Bruzda, A., Nowicki, J., Gumułka, M., Molik, E., Schwarz, T., and Klocek, C. (2015). Perception of the environment in farm animals. A review. *Annals of Animal Science*, 15, pp. 565 – 589.

Awad, A. I. (2016). From Classical Methods to Animal Biometrics. *Comput. Electron. Agric.*, *123*(C), 423–435. https://doi.org/10.1016/j.compag.2016.03.014

Adachi, S. (2014.). Application of Quantum Annealing to Training of Deep Neural Networks, 18.

Barkema, H., von Keyserlingk, M., Kastelic, J., Lam,T., Luby, C., Roy, J.-P., LeBlanc, M., Keefe, G. and Kelton, D. (2015). Invited review: Changes in the dairy industry affecting dairy cattle health and welfare, *Journal of Dairy Science* 98 (11), pp. 7426-7445.

Banhazi, T.M. and Tscharke, M. (2016). A brief review of the application of machine vision in livestock behaviour analysis, *Journal of Agricultural Informatics*, 7 (1).

BURGES, C. J. C. (2010). A Tutorial on Support Vector Machines for Pattern Recognition, 43

Busse, M., Schwerdtner, W., Siebert, R., Doernberg, A., Kuntosch, A., König, B., & Bokelmann, W. (2015). Analysis of animal monitoring technologies in Germany from an innovation system perspective. *Agricultural Systems*, *138*(C), 55–65.

Coulon, M., Baudoin, C. & Heyman, Y., & Deputte, B.L. (2011). Cattle discriminate between familiar and unfamiliar conspecifics by using only head visual cues. *Animal Cognition*, 14, pp. 279 – 290.

Cangar, Ö., Leroy, T., Guarino, M., Vranken, E., Fallon, R., Lenehan, J., Berckmans, D. (2008). Automatic real-time monitoring of locomotion and posture behaviour of pregnant cows prior to calving using online image analysis. *Computers and Electronics in Agriculture*, *64*(1), 53–60.https://doi.org/10.1016/j.compag.2008.05.014

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, *88*(2), 303–338. https://doi.org/10.1007/s11263-009-0275-4

Freier, N. G. (2011.). The Fast-Paced Change of Children's Technological Environments, 11.

Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3354–3361). Providence, RI: IEEE. https://doi.org/10.1109/CVPR.2012.6248074

Grandin, T. (2015). Improving animal welfare: *A practical approach (2nd ed). Oxfordshire*, UK: CABI

Hermans, G., Ipema, A., Stefanowska, J. and Metz, J. (2003). The Effect of Two Traffic Situations on the Behaviour and Performance of Cows in an Automatic Milking System, *Journal of Dairy Science* 86 (6), pp. 1997-2004.

Haidet, K. K., Tate, J., Divirgilio-Thomas, D., Kolanowski, A., & Happ, M. B. (2009). Methods to improve reliability of video-recorded behavioral data. *Research in Nursing & Health*, *32*(4), 465–474. https://doi.org/10.1002/nur.20334

I., Metz, J.H.M., Maltz, E., Dijkhuizen, A.A., Speelman, L. (2000). Designing the optimal robotic milking bar, Part 1: *Quantifying facility usage. J. Agric. Eng. Resou*r. 76, pp. 37–49.

Ionescu, D., Ionescu, B., Gadea, C., & Islam, S. (2011). An intelligent gesture interface for controlling TV sets and set-top boxes. In *2011 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI)* (pp. 159–164). https://doi.org/10.1109/SACI.2011.5872992/

J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, "Deep learning for content-based image retrieval: A comprehensive study," in Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014, pp. 157–166.

J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Las Vegas, Nevada, June 2016, pp. 770–778

Kiley-Worthingthon, M. & De La Plain, D. (1983). The social organization of the herd. In Folsch, D. W. (Ed.), *The behaviour of beef suckler cattle* (pp. 105 – 123).

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., … Dollár, P. (2014). Microsoft COCO: Common Objects in Context. *ArXiv:1405.0312 [Cs]*. Retrieved from http://arxiv.org/abs/1405.0312

L. Nagl, R. Schmitz, S. Warren, T. Hildreth, H. Erickson, and D. Andresen, "Wearable sensor system for wireless state-of-health determination in cattle," *Proceeding of the 25th Annual International Conference of the IEEE EMBS*, Cancun, Mexico, vol. 4, 2003, pp. 3012–3015

McTavish, E. J., Decker, J. E., Schnabel, R. D., Taylor, J. F., & Hillis, D. M. (2013). New World cattle show ancestry from multiple independent domestication events. *Proceedings of the National Academy of Sciences*, 110, E1398 – E1406.

Marchant, J.A. (1988). Computer vision in agricultural engineering. *Agric. Eng*., 43(2), pp. 40-42.

Miguel-Pacheco, G., Kaler, J., Remnant, J., Cheyne, L., Abbort, C., French, A., Pridmore, T., Huxley, J. (2014). Behavioural changes in dairy cows with lameness in an automatic milking system. *Appl. Animal Behav. Sci*. 150, pp. 1–8.

Nahari, R. V., Jauhari, A., Hidayat, R., & Alfita, R. (2017). Image Segmentation of cows using Thresholding and K-Means Method. *International Journal of Advanced Engineering, Management and Science*, *3*(9), 913–918. https://doi.org/10.24001/ijaems.3.9.2

P. Vincent, H. Larochelle, I. Lajoie, and P. Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010 Phillips, C. (2002). *Cattle behavior and welfare. Oxford, UK: Blackwell Publishing.*

Porto, S. MC., Arcidiacono, C., Anguzza, U. and Cascone, G. (2015). The automatic detection of dairy cow feeding and standing behaviours in free-stall barns by a computer vision-based system. *Biosystems Engineering*, 133, pp. 46–55.

Rutten, C., Velthuis, A., Steeneveld, W., and Hogeveen, H. (2013). Invited review: Sensors to support health management on dairy farms. *Journal of Dairy Science* 96 (4), pp. 1928-1952.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *ArXiv:1506.01497 [Cs]*. Retrieved from http://arxiv.org/abs/1506.01497

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *ArXiv:1506.01497 [Cs]*. Retrieved from http://arxiv.org/abs/1506.01497

Santoni, M. M., Sensuse, D. I., Arymurthy, A. M., & Fanany, M. I. (2015). Cattle Race Classification Using Gray Level Co-occurrence Matrix Convolutional Neural Networks. *Procedia Computer Science*, *59*, 493–502. https://doi.org/10.1016/j.procs.2015.07.525

Simonyan, K. and Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. Published as a conference paper at ICLR 2015, arXiv:1409.1556.

Silberman, N., Sontag, D., & Fergus, R. (2014). Instance Segmentation of Indoor Scenes Using a Coverage Loss. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014* (Vol. 8689, pp. 616–631). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-10590-1_40

Shelhamer, E., Long, J., & Darrell, T. (2016). Fully Convolutional Networks for Semantic Segmentation. *ArXiv:1605.06211 [Cs]*. Retrieved from http://arxiv.org/abs/1605.06211

Soniya, Paul, S., & Singh, L. (2015). A review on advances in deep learning. In *2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI)* (pp. 1–6). https://doi.org/10.1109/WCI.2015.7495514

Salakhutdinov, R. (2015). Learning Deep Generative Models. *Annual Review of Statistics and Its Application*, *2*(1), 361–385. https://doi.org/10.1146/annurev-statistics-010814-020120

Shin, H.-C., Orton, M. R., Collins, D. J., Doran, S. J., & Leach, M. O. (2013). Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1930–1943. https://doi.org/10.1109/TPAMI.2012.277

Taylor, A. A., and Davis, H. (1998). Individual humans as discriminative stimuli for cattle Bos taurus. *Applied Animal Behavior Science*, 58, pp. 13 – 21.

Turk, M., Morgenthaler, D., Gremban, K., & Marra, M. (1987). Video road-following for the autonomous land vehicle. In *1987 IEEE International Conference on Robotics and Automation Proceedings* (Vol. 4, pp. 273–280). https://doi.org/10.1109/ROBOT.1987.1088030

Ter-Sarkisov, A., Ross, R., Kelleher, J., Earley, B., & Keane, M. (2018). Beef Cattle Instance Segmentation Using Fully Convolutional Neural Network. *ArXiv:1807.01972 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1807.01972

Ustyuzhaninov, I., Michaelis, C., Brendel, W., & Bethge, M. (2018). One-shot Texture Segmentation. *ArXiv:1807.02654 [Cs]*. Retrieved from http://arxiv.org/abs/1807.02654

von Keyserlingk, M., Rushen, J., de Passille, A. and Weary, D. (2009). Invited review: The welfare of dairy cattle. Key concepts and the role of science, *Journal of Dairy Science* 92 (9), pp. 4101-4111.

Appendix

All the Code for this thesis research is made available at the GitHub repository at the link **https://github.com/danish07me/Mask-RCNN/ .** The implementation code and idea are taken from matter plot Mask RCNN available at https://github.com/matterport/Mask_RCNN/. The repository contains all the directory of images and implementation code for training and evaluation. The below section gives code for training and visualisation. The complete code is provided on the above URL.

Training and Testing Code for instance Segmentation:

```
import os,sys
import random
import math
import re
import time
import numpy as np
import cv2
import matplotlib
import matplotlib.pyplot as plt
from config import Config
from PIL import Image
from skimage.measure import label, regionprops
import utils
import model as modellib
import visualize
from model import log
import argparse
#import coco
os.environ["CUDA_VISIBLE_DEVICES"]="0"
import pickle
from skimage.transform import resize
```

```
root_dir = os.getcwd()
MODEL_DIR = os.path.join(root_dir, 'logs/onlycoco')
#pretrained_model = 'pretrained_weights/mask_rcnn_coco.h5'
pretrained_model = os.path.join(MODEL_DIR,
"cows20180709T1147/mask_rcnn_cows_0003.h5")
def natural_sort(l):
    convert = lambda text: int(text) if text.isdigit() else text.lower()
    alphanum_key = lambda key: [convert(c) for c in re.split('([0-9]+)', key)]
    return sorted(l, key=alphanum_key)


class CowConfig(Config):
  NAME = 'cows'
  GPU_COUNT = 1
  IMAGES_PER_GPU = 2
  NUM_CLASSES = 1 + 1
  DETECTION_MIN_CONFIDENCE = 0.5
  IMAGE_MIN_DIM = 512
  IMAGE_MAX_DIM = 512
  VALIDATION_STEPS = 1
  RPN_ANCHOR_SCALES = (16,32,64,128,256)
  TRAIN_ROIS_PER_IMAGE = 32
# roughly 5000 divides 4, the mini-batch size, I think...
  STEPS_PER_EPOCH = 3000
  VALIDATION_STEPS = 5
class CowTestConfig(Config):
  NAME = 'cows'
  GPU_COUNT = 1
  DETECTION_MIN_CONFIDENCE = 0.5
  IMAGES_PER_GPU = 1
```

```
NUM_CLASSES = 1 + 1
  IMAGE_MIN_DIM = 1280
  IMAGE_MAX_DIM = 1280
  VALIDATION_STEPS = 5
#RPN_ANCHOR_SCALES = (16,32,64,128,256)
  #TRAIN_ROIS_PER_IMAGE = 32
  # roughly 5000 divides 4, the mini-batch size, I think...
  #STEPS_PER_EPOCH = 3000
  #VALIDATION_STEPS = 5
class CowDataset(utils.Dataset):


  # this creates the 'dataset' without loading the actual images
  def load_cows(self, count, stage):
    # randomize the set
    self.add_class("cow_dataset", 1, "cow")
    # if this is a validation set, select 100 imgaes at random
    if stage == 'validation':
      list_of_validation_items = np.random.choice(range(count), size= 2,
replace=False)
    # idx will be a random number
    if stage == 'training':
      for fs in os.listdir('\\Users\\Danish\\kaggle-ds-bowl-2018-
baseline\\cow_imgs_train'):
        if fs.split('.')[1] == 'png':
          #self.add_image("cow_dataset", image_id = idx, path =
os.path.join('/home/ICTDOMAIN/453615/NewData/total_cows_large_clean/total_tr
aining_clean', str(idx) + '.png'), width=width, height=height, stage = 'training')
          self.add_image("cow_dataset", image_id = fs.split('.')[0], path =
os.path.join('\\Users\\Danish\\kaggle-ds-bowl-2018-baseline\\cow_imgs_train', fs),
stage = 'training')
```

```
    elif stage == 'validation':

        for fs in os.listdir('\\Users\\Danish\\kaggle-ds-bowl-2018-
baseline\\cow_imgs_val'):

            if fs.split('.')[1] == 'png':

                self.add_image("cow_dataset", image_id = fs.split('.')[0], path =
os.path.join('\\Users\\Danish\\kaggle-ds-bowl-2018-baseline\\cow_imgs_val', fs),
stage = 'validation')

                print (fs, count, self._image_ids)

    elif stage == 'testing_mydata':

        for fs in os.listdir('\\Users\\Danish\\kaggle-ds-bowl-2018-
baseline\\cow_imgs_val_large'):

            if fs.split('.')[1] == 'png':

                #self.add_image("cow_dataset", image_id = idx, path =
os.path.join('/home/ICTDOMAIN/453615/NewData/total_cows_large_clean/total_t
raining_clean', str(idx) + '.png'), width=width, height=height, stage = 'training')

                self.add_image("cow_dataset", image_id = fs.split('.')[0], path =
os.path.join('\\Users\\Danish\\kaggle-ds-bowl-2018-baseline\\cow_imgs_val_large',
fs), stage = 'testing_mydata')

                print (fs, count, self._image_ids)

def load_mask(self, image_id):


    if self.image_info[image_id]['stage'] == 'training':

        path = '/Users/dhananjaykittur/mask/kaggle-ds-bowl-2018-
baseline/cow_imgs_gt_train'

        this_mask = np.array(pickle.load(open(os.path.join(path,
self.image_info[image_id]['id'] + '.r'), "rb")))

    elif self.image_info[image_id]['stage'] == 'validation' or
self.image_info[image_id]['stage'] == 'testing_mydata':

        path = '\\Users\\Danish\\kaggle-ds-bowl-2018-
baseline\\cow_imgs_val_gt_large'

        this_mask = np.array(pickle.load(open(os.path.join(path,
self.image_info[image_id]['id'] + '.r'), "rb")))
```

```
  class_ids = np.array(10*[1], dtype = np.int32)
      return this_mask, class_ids
if __name__ == '__main__':


   # Parse command line arguments
   parser = argparse.ArgumentParser(description='Train Mask R-CNN on cow data')
   parser.add_argument("stage", help="'train' or 'val'")
   args = parser.parse_args()
   if args.stage == 'train':
       # load model
       source = '\\Users\\Danish\\kaggle-ds-bowl-2018-baseline'
       print (MODEL_DIR)
       config = CowConfig()
       config.display()
       model = modellib.MaskRCNN(mode="training", config = config,
model_dir=MODEL_DIR)
       model.load_weights(pretrained_model, by_name=True,
exclude=["mrcnn_class_logits", "mrcnn_bbox_fc", "mrcnn_bbox", "mrcnn_mask"])
         print ('Weigths loaded')
       # load dataset
       train_dataset = CowDataset()
       count = len(os.listdir(os.path.join(source, 'cow_imgs_train')))
       print ('c', count)
       train_dataset.load_cows(count = count, stage = 'training')
       train_dataset.prepare()
       val_dataset = CowDataset()
       count = len(os.listdir(os.path.join(source, 'cow_imgs_val')))
       print ('s', count)
```

```
        val_dataset.load_cows(count=count, stage = 'validation')
        val_dataset.prepare()
        print ('train', train_dataset.image_ids)
        print ('val', val_dataset.image_ids)
        ##############################################
        # OK train model#############################
        # HEADS ONLY! same learning rate as in config
        ##############################################
        model.train(train_dataset, val_dataset, learning_rate =
config.LEARNING_RATE, epochs=3, layers='all')
  elif args.stage == 'inference_mydata':
        #source = '/home/ICTDOMAIN/453615/NewData/total_cows_large_clean/'
        # just for the base model
        good_classes=[20]
        source = '\\Users\\Danish\\kaggle-ds-bowl-2018-baseline'
        config = CowTestConfig()
        config.display()
        model = modellib.MaskRCNN(mode="inference", config = config,
model_dir='pretrained_weights')
        model.load_weights(pretrained_model, by_name=True)
        print ("Weights loaded")
        # load dataset
        test_dataset = CowDataset()
        count = len(os.listdir(os.path.join(source, 'cow_imgs_val_large')))
        #count = len(os.listdir(os.path.join(source, 'cow_imgs_val')))
        test_dataset.load_cows(count=count, stage = 'testing_mydata')
        test_dataset.prepare()
        print("Images: {}\nClasses: {}".format(len(test_dataset.image_ids),
test_dataset.class_names))
        #print (test_dataset.image_ids)
         all_AP_list=[]
```

```
            for th in thresholds:
        AP_list=[]
        for img in test_dataset.image_ids:
            #image, image_meta, gt_mask = modellib.load_image_gt(test_dataset, config,
img)
            image, image_meta, class_ids, gt_bbox, gt_mask=
modellib.load_image_gt(test_dataset, config, img)
            print (img)
            plt.imsave(str(img)+'.png', image)
            mask,gt_class_ids = test_dataset.load_mask(img)
            results = model.detect([image], verbose=1)
            r = results[0]
            print (r.keys())
            plt.imsave('gt' + str(img)+'.png', mask[:,:,0])
            # for the base model-keep only cow predictions
            present_class = r['class_ids']
            for idx, cls in enumerate(present_class):
               if cls not in good_classes:
                    np.delete(r['rois'], idx, axis=0)
                    np.delete(r['class_ids'], idx, axis=0)
                    np.delete(r['scores'], idx, axis=0)
            print ('class', r['rois'].shape, r['scores'].shape, r['class_ids'].shape, r['masks'].shape,
gt_bbox.shape, mask.shape)
            AP, precisions, recalls, overlaps = utils.compute_ap(gt_bbox, gt_class_ids, mask,
r['rois'], r['class_ids'], r['scores'], r['masks'], iou_threshold = th)
            print (AP, precisions, recalls)
            AP_list.append(AP)
            print (np.mean(AP_list), th)
        all_AP_list.append(np.mean(AP_list))
        all_AP_list.append(np.mean(all_AP_list))
print ('all_AP_list', all_AP_list)
```

Visualisation Code for instance segmentation

```
import os, re
import sys
import random
import math
import numpy as np
import skimage.io
import matplotlib
import matplotlib.pyplot as plt
os.environ["CUDA_VISIBLE_DEVICES"]="0"
import pickle
#import tensorflow as tf
from skimage.measure import label,regionprops
from PIL import Image
from tensorflow.python.client import device_lib
OOT_DIR = os.path.abspath("../")
import utils
import model as modellib
import visualize
from model import log
sys.path.append(os.path.join(ROOT_DIR, "samples", "coco"))  # To find local version
import pdb; pdb.set_trace()
import coco
import config
MODEL_PATH = os.path.join(ROOT_DIR, 'pretrained_weights/mask_rcnn_coco.h5')
#MODEL_PATH = os.path.join(ROOT_DIR, 'saved_weights/heads/mask_rcnn_cows_0001.h5')
MODEL_DIR = os.path.join(ROOT_DIR, 'logs')
```

```
def natural_sort(l):
    convert = lambda text: int(text) if text.isdigit() else text.lower()
    alphanum_key = lambda key: [convert(c) for c in re.split('([0-9]+)', key)]
    return sorted(l, key=alphanum_key)
class InferenceConfig(coco.CocoConfig):
    # Set batch size to 1 since we'll be running inference on
    # one image at a time. Batch size = GPU_COUNT * IMAGES_PER_GPU
    NAME = 'BaseNetwork'
    GPU_COUNT = 1
    NUM_CLASSES = 80 + 1
    IMAGES_PER_GPU = 1
    IMAGE_MIN_DIM = 512
    IMAGE_MAX_DIM = 512
    VALIDATION_STEPS = 1
    #RPN_ANCHOR_SCALES = (16,32,64,128,256,512)
    #RPN_ANCHOR_STRIDE = 1
    #TRAIN_ROIS_PER_IMAGE = 32
class CowTestConfig(coco.Config):
   NAME = 'cows'
   GPU_COUNT = 1
   DETECTION_MIN_CONFIDENCE = 0.7
   IMAGES_PER_GPU = 1
   NUM_CLASSES = 80 + 1
   IMAGE_MIN_DIM = 1280
   IMAGE_MAX_DIM = 1280
   RPN_NMS_THRESHOLD = 0.7
config = CowTestConfig()
config.display()
model = modellib.MaskRCNN(mode='inference',config=config,model_dir =
```

```
model.load_weights(MODEL_PATH,
by_name=True)class_names_alex=['BG','cows']
class_names = ['BG', 'person', 'bicycle', 'car', 'motorcycle', 'airplane',
               'bus', 'train', 'truck', 'boat', 'traffic light',
               'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird',
               'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear',
               'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie',
               'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball',
               'kite', 'baseball bat', 'baseball glove', 'skateboard',
               'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup',
               'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple',
               'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza',
               'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed',
               'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote',
               'keyboard', 'cell phone', 'microwave', 'oven', 'toaster',
               'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors',
               'teddy bear', 'hair drier', 'toothbrush']
for idx, val in enumerate(class_names):
    print (idx, val)
file_names = natural_sort(next(os.walk(IMAGE_DIR))[2])
#print (file_names[0], class_names_alex[1])
#file_names = [file_names[0]]
pallete = [0, 255]
for idx, im in enumerate(file_names):
   #print (im, im.shape)
   image = skimage.io.imread(os.path.join(IMAGE_DIR, im))
   print ('iii', image.shape)
   image = image[:,:,0:3]
   results = model.detect([image], verbose=1)
   r = results[0]
```

```
    good_class=[20]
   #good_class=[]
   good_preds = 0
   for pred in r['class_ids']:
      if pred in good_class:
         good_preds += 1
   output_mask = np.zeros([image.shape[0],image.shape[1], good_preds],
dtype=np.uint8)
   current_pred = 0
   for idw, x in enumerate(r['scores']):
      print(idw, x, r['class_ids'][idw], type(r['class_ids'][idw]),
np.unique(r['masks'][:,:,idw]))
      if r['class_ids'][idw] in good_class:
            maskrcnn_mask = r['masks'][:,:,idw]==1
            output_mask[maskrcnn_mask,current_pred]=1 #
            current_pred +=1
   present_class = r['class_ids']
   print ('AA', r['class_ids'])
   correct_class = [True if y in good_class else False for y in present_class]
   print (correct_class, present_class, good_class, r['masks'])
    r['class_ids'] = r['class_ids'][correct_class]
   r['rois'] = r['rois'][correct_class]
   r['masks'] = r['masks'][:,:,correct_class]
   r['scores'] = r['scores'][correct_class]
   print ('BB', r['class_ids'])
   #pickle.dump(output_mask, open(os.path.join(SAVE_DIR, im.split('.')[0] +
'.r'),'wb'))
   #print (r['rois'])
   visualize.display_instances(SAVE_DIR, im, image, r['rois'], r['masks'],
r['class_ids'], class_names, r['scores'])
```