

CAMERA BASED DRONE DETECTION

A thesis submitted in partial fulfilment of the
requirements for the award of the degree

Master of Engineering (Electrical)

from

University of Wollongong

by

Abhishek Balaji Sharma

**School of Electrical, Computer and Telecommunications
Engineering**

June 2020

Supervisor: Assoc.Prof.Son Lam Phung

Abstract

The camera-based drones have been an essential part of the sports and the entertainment industry. It has also helped in various surveillance and monitoring activities for the government and defence industries. However, usage of drones in the prohibited areas cause concern to the defence forces. There have been instances in Australia where unregistered drones are found to be disturbing the visitors and native animals in the parks. It also has impacts on interfering with the fighting bushfires and park management activities. New South Wales government have taken a stringent plan to take action on illegal Unmanned aerial vehicles flying in the designated areas such as airport, military bases, national parks and buildings having the critical infrastructures. A detector that detects the presence of drones in the restricted areas is needed to solve this issue and alert the defence personnel.

A large dataset of drone images is extracted, and segmentation is performed in the Matlab. Faster Regional Convolutional Network algorithm is applied to the labelled data and tested with the new samples to build a robust intelligence system to achieve the task. The experiments reveal the number of layers used to train the network along with the training parameters used in the neural networks. The objects in the dataset are prepared by annotation using the test Image labeller in the Matlab. The work undertaken shows significant results in the test images used in the project. It also provides an opportunity for future work on this particular topic using different methods to optimise and enhance the feature selection and object detection techniques using the RCNN family. An intelligent system is built in this project based on the computer vision and deep learning technology that can help the governing body to detect the presence of unregistered drones flying in the restricted areas. The results show that the detector model produces the mAP of 94.83% with Faster RCNN.

Acknowledgements

First of all, I would like to thank my supervisor, Dr Son Lam Phung for his constant support during this project. His feedback and guidance have been highly important and motivation to bring the project to this stage. I would like to thank the School of Electrical, Computer and Telecommunication to provide an opportunity to work on a thesis on the basis of student interest. Finally, I would like to thank the members and staff of the University of Wollongong for providing me with the resources and space for the progress made for the project in this subject.

Statement of Originality

I, Abhishek Balaji Sharma, declare that this thesis, submitted as part of the requirements for the award of Bachelor of Engineering, in the School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications or assessment at any other academic institution.

As author of this thesis, I also hereby grant, subject to any prior confidentiality agreements, SECTE permission, to use, distribute, publish, exhibit, record, digitize broadcast, reproduce and archive this work for the purposes of further research and teaching. This thesis is subject to prior confidentiality agreement.

Signature: Abhishek Balaji Sharma

Print Name: Abhishek Balaji Sharma

Student ID Number: 6237083

Date: 01 June 2020

Contents

Abstract	ii
Abbreviations and Symbols	ix
List of Changes	x
1 Introduction	1
1.1 Project Objectives	1
1.2 Report Structure	1
2 Literature Review	3
2.1 Region Proposal Network	3
2.2 Intersection of Union	5
2.3 Faster RCNN algorithm	7
3 Design and Methodology	11
3.1 Stochastic Gradient Descent Momentum	11
4 Experimentation and Results	14
4.1 Data Preparation	14
4.2 Preparing the training dataset through annotation	15
4.3 Training the dataset	17
4.4 Testing the Network with an Image	18
4.5 Analysis of the outcome	19
5 Conclusion	23
5.1 Research Summary	23

5.2 Future Research	24
References	26
A Project Plan and Specifications	27
B MATLAB CODING	30
B.1 RESIZING THE TRAINING DATASET	30
B.2 NETWORK LAYER FOR RESNET50	30
B.3 TRAINING DATASET	36
B.4 TESTING DATASET	38

List of Tables

2.1	Detection results without post processing	10
2.2	Detection results post processing	10
4.1	The table shows the relevant table for mini-batch accuracy and loss values	20
4.2	The table shows the relevant table for RPN mini-batch accuracy and loss values	22

List of Figures

2.1	A representation of convolutional neural networks[1]	3
2.2	Distribution of features of image to anchor boxes[5]	4
2.3	Distribution of pooling layers in CNN[6]	5
2.4	Performance analysis of custom Faster RCNN algorithm[25]	10
4.1	Annotated images of the training dataset	16
4.2	Test images getting detected by the Faster RCNN object detector. The detector shows fine results and can detect the objects for various views producing high accuracy rates	19
4.3	The graphical representation shows the performance analysis of Mini- batch accuracy,Mini-batch loss and Mini-batch RMSE	21
4.4	The graphical representation shows the performance analysis of RPN Mini-batch accuracy and RPN Mini-batch RMSE	21

Abbreviations and Symbols

AGA	Adaptive Generic Algorithm
BNN	Binary Neural Network
CMS	Contextual Multi-Scale
CNN	Convolutional Neural Network
FCIS	Fully convolutional instance segmentation
FPR	False Positive Rate
Fps	Frames per second
GAN	Generative Adversarial Network
GPU	Graphical Processing Unit
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
IRNN	Image Recognition Neural Network
mAP	Mean Average Precision
MCG	Multi-Scale Combinatorial Group
MS-CNN	Multi-Scale Convolutional Neural Network
NMS	Non-Maximum Suppression
RCNN	Regional Convolutional Neural Network
ROI	Region of Interest
RNN	Recurrent Neural Network
RPN	Region Proposal Network
SGDM	Stochastic Gradient Descent Momentum
SIFT	Scale Invariant Feature Transform
TPR	True Positive Rate

List of Changes

This Thesis is the updated version of ECTE940 Session 1. Below is a partial list of the changes.

Section	Statement of Changes	
Abstract	Partial changes in context of the previous submitted one for the purpose of Autumn session.	ii
Glossary	Changed to alpha numeric numbering	ix
Introduction	Added new details to the chapter	1
Literature Review	Added new content.	3
Design	Explained important features	11
Conclusion	Added new content.	23

Chapter 1

Introduction

Camera surveillance is the third eye in society and has the potential to perform robust monitoring activities. Object detection and recognition are superior technologies in the field of computer vision, and artificial intelligence has the power to detect and classify the objects. It has a variety of applications including automation security, robot navigation, industrial damage investigation. Object detection and identification involve training data under various background, and lighting conditions and the machine learns the dataset under the algorithms fed by the user. The purpose of our thesis is to build an intelligent system that detects and recognizes the presence of drones using the camera and informs the same at the receiving end. We perform the process using the Faster RCNN algorithm that is popular amongst the industry and deep learning researchers.

1.1 Project Objectives

The objective of our thesis is to:

- Conducting a Literature review of the related background details.
- Data preparation and annotation.
- Designing the network layers for the algorithm.
- Training dataset and Experimentation using Deep learning.
- Analysis of the results.

1.2 Report Structure

This report is organised as follows:

- **Chapter 1** provides the background and the objective of the thesis.
- **Chapter 2** deals with literature review
- **Chapter 3** deals with design and methodology

- **Chapter 4** presents the experimentation techniques and the results obtained. The results attained are analysed in detail.
- **Chapter 5** provides a summary of the report, along with the future scope of the project.

Chapter 2

Literature Review

The computational power in the recent times has been dominant compared to the later period when CNN's were reliable on low computation. The CNN provided the neural networks that proved the concept of breaking down the specimen into layers and detecting the pattern to complete the job of a classifier and feature identifier [1]. In today's era, the Faster RCNN algorithm has been a widely used concept for object detection, classification and feature extraction while building on the RPN and the Fast RCNN[2].

2.1 Region Proposal Network

The RPN collects the ROI of the target object by thresholding techniques and edge detection to create bound around the object in the specimen. It undergoes various iterations to break down the samples of the regions of the purpose in to multiple layers and perform the convolution process [?] [3].

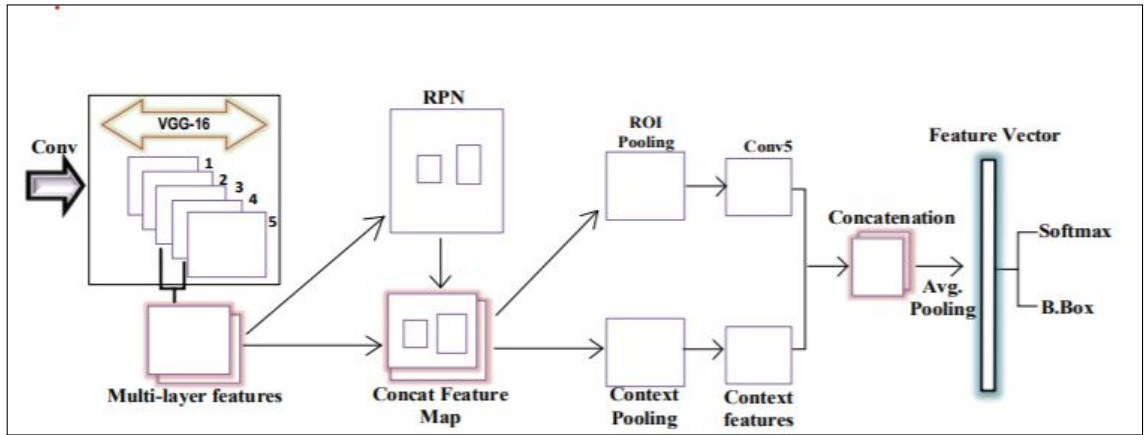


Figure 2.1: A representation of convolutional neural networks[1]

The convolution layers consist of features of the regions that will be extracted to decode and verify the pattern of the object. As the CNN model performs this task at a fundamental and slow level, the process of RPN and the fast RCNN will make the iterations much faster while the training time will be in the minimum. The

RNN and IRNN process is least efficient when performed as the single system on the training purpose as it involves the only localisation of pixels at a slower rate while the levels of abstraction skip the pooling layer [4][3]. The RPN consists of the max-pooling layer at the output of the network that generates the foreground and the background for the particular frame. The foreground usually consists of the target image that will be annotated in the bounding box and the background is least used in most of the procedure[5].

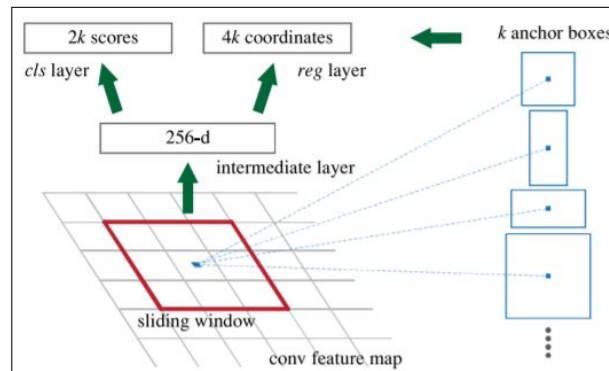


Figure 2.2: Distribution of features of image to anchor boxes[5]

The SGDM momentum that acts as the optimiser helps share the weights across the network for faster computing for the feature extraction for the classification purpose. The positive overlap range of 0 to 0.3 as the default. This overlap range decides an overlap ratio of various objects classes that can share the typical ROI. A network containing various layers in the neural network in a pre-trained entity consists of multiple layers for the convolution, max pooling, ReluLayer, softmax layers and output classification layers such as ResNet50 or the ResNet101. It is ideal for the Faster RCNN network[5].

Once this procedure is complete, the Fast RCNN comes into play. The Fast RCNN has better in mAP compared to the CNN, and the disk storage lacks the feature caching. They follow no weight sharing among the network for the computation purpose, and so the training of the data is quicker while the detection rate is slower [1]. The Fast RCNN works based on multi-stage pipelining for faster training purposes. The property of classification, feature extraction and detection lies in the hand of fine-tuning at the proposal layers and the softmax layer in the max-pooling

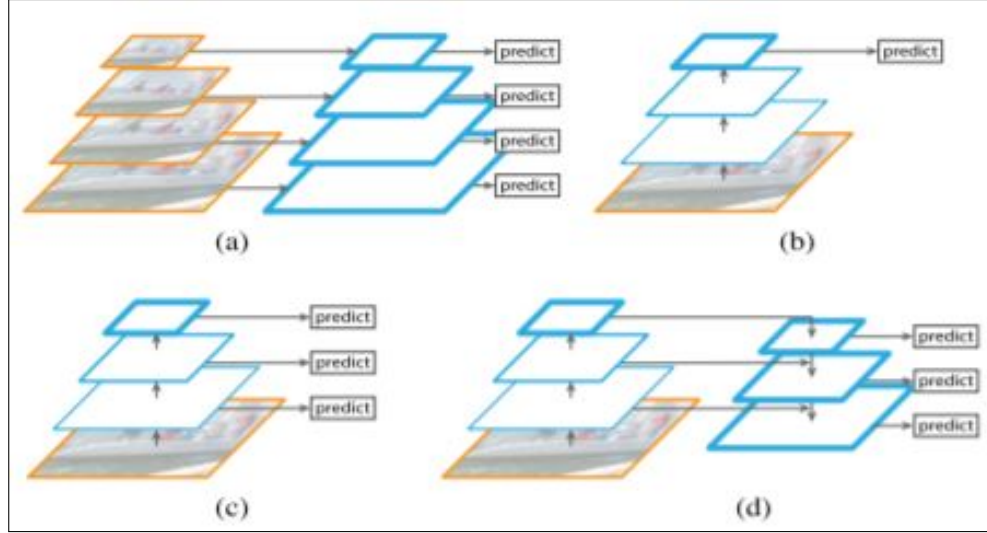


Figure 2.3: Distribution of pooling layers in CNN[6]

at the ROI pooling layer. The combination of the RPN and the fast RCNN performs a Faster RCNN for the object detector application. The upsampling for the feature extraction is done at the end of the ROI pooling layer to perform the super-resolution RCNN, thereby downsampling the ordinary CNN[6]. Downsampling the features in the initial stages leads to loss of small and vital information [1]. It is recommended to perform the feature extraction and fine-tuning after the process of convolution for better classification results. The output layer at the RNN passes a 3x3 convolution layer to the Fast RCNN[7][6].

2.2 Intersection of Union

Every proposal is represented by the anchor boxes to categorise the features between the layers. Overall the Mean average precision(mAP) must be high to produce a perfect test result. The mean average precision is based on the two factors precision and recall-

$$\text{Precision} = \frac{TP}{TP + FP} ,$$

$$\text{Recall} = \frac{TP}{TP + FN} ,$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall},$$

The precision takes care of how precise is the prediction in percentage and the recall works on how many positives are obtained from the result [8]. Higher the value of the mAP, better the detection results. The TP(True positive), TN(True negative), FP(False positive), FN(False negative) plays important factor to obtain the mean average precision [9].

$$IoU = \frac{|S_{\text{anchorBox}} \cap S_{\text{groundTruth}}|}{|S_{\text{anchorBox}}|} \cup S_{\text{groundTruth}},$$

The Intersection of Union is a crucial concept in object detection. During the process of testing the values of the annotated boundary box values should match the value of the test dataset. This process happens through layer by layer to produce a result in high precision[10]. When the selection value of the ground truth and the IoU is more significant than 0.7, then the possibility of the object to be detected is very high, and if the value is lesser than 0.3, the chances are low. For any measure in between 0.7 and 0.3, the image goes undetected [2].

$$L(\{p_i\}, \{t_i\}) = s \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*),$$

In the above equation i refers to index of anchors in mini-batch. p_i and p_i^* represent the predicted probability and true label of $anchor_i$. t_i indicates the correction of prediction boxes relative to candidate boxes.

t_i^* indicates the correction of ground truth boxes relative to candidate boxes.

As the object detection is an essential area of research in the development of deep neural networks [11][8]. The dataset was containing overhead views used with variations in the field of view, background, illumination conditions, poses, scales, sizes, angles, height, aspect ratio and camera resolutions. Faster RCNN provided TPR(True positive rate) of 94% with FPR(False positive rate) of 0.4% for overhead view and 92% per other aspects. The generalization performance of each object across completely different test images with challenging variations in backgrounds utterly different than training images and tested with both symmetric and asymmetric views in a cluttered environment [12][13]. It provides a piece of additional information on object detection compared to the traditional detection methods based on tailored features including HOG and Haar wavelets [?]. The model predicts the objects in the test image accurately and is cost-effective in terms of technical aspects compared to the traditional method. The detection did not happen in any consistent pattern and improved on constant training, and the bounding box is adjusted automatically according to the object in the test dataset. One crucial observation is no false detection produced on the shadow of the object. Multi-scaling is another concept that works at the output layer to improve the levels of detection rates [?][14].

2.3 Faster RCNN algorithm

The ordinary CNN prefers the frame size to be always fixed, but the Faster RCNN can perform the task of multi-scaling, which means the

frame size can be of any dimension. The layers breakdown the entire specimen to its roots and then the pattern is recorded[?]. A multi-scale feature extraction is not a suitable method to apply on the smaller datasets otherwise proves very costly[1]. It is a proven fact that CNN is least significant in the smaller dataset and proves to be substantial in a larger dataset of the same scale. As the computational power in the ordinary CNN is low, the concept of the sliding window is used to check the presence of an object in comparison to the size of the sliding window. This method is known as the Over feat method, and it brings an outcome for producing the classifier and detecting important small features in the framed specimen[15]. As the Faster RCNN is suitable for scaling its output layer, it lacks the capacity to handle over-segmentation issues, and hence, as discussed, the downsampling is not recommended during the initial stages of the neural network. In this paper[4], the Faster RCNN algorithm detects the object form the satellite images. Because the Faster RCNN false detects the object of smaller sizes around 17x17 pixels on average and false detects objects of similar dimensions on a test image, the RPN is tailor-made according to the required standards to improve the TPR [16]. The RPN made for three models that include ZF model, VGG 16 model, VGG_CNN_M_024 model. However, we use a VGG 16 model for our dataset. The detection standards based on IoU between the object in the test image and obtained ground truth data. This method reduces the FPR compared to traditional methods and produces an accuracy of 94.99% [17]. The RCNN algorithm has the capacity to break down the features of the specimen, and the system can also

identify the duplicate frames and features in case of the video file. One of the properties of the Faster RCNN is it avoids the redundancy in the input data and creates a static frame. This technique proves to be one of the most optimised methods for tracking objects in a frame that has many duplicate frames in the adjacent locations. This causes a chance of noise or disturbance in the frame that can be eliminated by the SIFT[18][19]. The properties of the SIFT can also be used in case of the overfeat occurs at the regular interval of time. The weights in the structure are very uneven and create a dynamic layer helps in reducing the redundancy of the large dataset, and thereby provides a benefit for the deep classification and feature extraction concepts [20]. The RCNN training procedures is more time-efficient compared to the RPN and the ordinary CNN, but the detection rate is comparatively slow. The efficient training rate is due to the multi-stage pipeline that distributes the computation of the data evenly without any particular goal of classification, feature extraction and detection and hence this method proves costly for multitasking using the GAN algorithm[21]. The other drawback of ordinary RCNN is sharing of only a few weights for the computational purpose, not even a single feature is stored as a cache due to lack of hard disk [22][23]. The Fast RCNN performs evenly well during the stages of ROI pooling and fine-tuning to extract high-quality features and properties of the small but essential information from the object pattern. The factors responsible for the feature explorations include the localisation and the annotation boundaries[24]. The accuracy of the bounding box and the regions depend on the training procedure and the background

pixels in the training and testing dataset [25].

Index	R1	R2	R3	R4	R5	R6
TP	1137	1044	1024	971	451	402
FP	62	40	58	61	69	104
FN	55	33	54	20	20	54
Precision(%)	94.83	96.31	94.64	94.09	86.73	79.45
Recall(%)	95.39	96.94	94.99	97.98	95.75	88.16
F1-score(%)	95.11	96.62	94.81	96.00	91.02	83.58

Table 2.1: Detection results without post processing

Index	R1	R2	R3	R4	R5	R6
TP	1136	1043	1023	971	449	400
FP	49	32	49	35	26	32
FN	56	34	55	20	22	56
Precision(%)	95.86	97.02	95.43	96.52	94.53	92.59
Recall(%)	95.30	96.84	94.90	97.98	95.33	87.72
F1-score(%)	95.58	96.93	95.16	97.25	94.93	90.09

Table 2.2: Detection results post processing

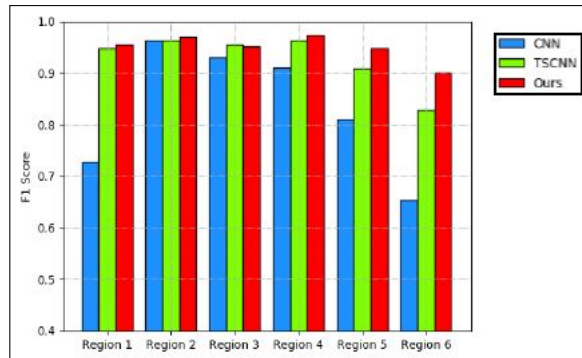


Figure 2.4: Performance analysis of custom Faster RCNN algorithm[25]

Table 2.1 and Table 2.2 shows the performance features of different Faster CNN models. Figure 2.4 presents the graphical representation of the same. The results show that an increased accuracy and performance is seen with the post processing Faster RCNN algorithm.

Chapter 3

Design and Methodology

Deep learning requires constant learning and understanding of mathematical concepts to produce values that differentiate the elements in the network. A training option in the deep learning must have suitable parameters to fulfill the objectives of training[26].

3.1 Stochastic Gradient Descent Momentum

SGDM is an optimizing technique to achieve a smoothness in the computation burden. As the network requires a lot of memory to perform the high computation techniques in the best efficient time, an iterative method is required which is performed by SGDM. The iterative method performed to work on iterations in a less convergence rate by performing faster iterations. Momentum provides the linear combination of the gradients through the technique of backpropagation learning. It has the capacity to perform averaging of all descents to produce a final value to ease the computation process according to the training dataset or requirement[27].

CNN has a grid like structure and is useful for applications in the fields like image processing and natural language processing. It works across the dimensions of various elements by employing a mathematical operation to derive weighting function. By applying this method in every time interval the convolution takes place.

$$s(t)=\int x(a)w(t-a)da = s(t) = (x * w)(t) ,$$

where the x represents the input value, w for kernel and s represents the output for feature map based on continous time intervals[26].

$$s(t)=(x * w)(t)=\sum_{a=-\infty}^{\infty} x(a)w(t-a) ,$$

The above equation performs discrete convolution operation and the values of x and w are based on the integer values of t[26].

$$S(i, j)=(I * K)(i, j)=\sum_m \sum_n I(i+m, j+n)K(m, n) ,$$

A two dimensional image I can produce relevant number of kernels. The elements i and j are dimensions of the image I with kernel K. These mathematical operation provide the model to build convolution network[26].

$$\theta_{n+1} = \theta_n - \alpha \nabla E(\theta_n) + \gamma (\theta_n - \theta_{n-1}) ,$$

The SGDM uses certain parameters for the network such as weights and biases. The momentum helps in reducing the oscillations to provide stability in learning to the system. In the above equation, the value of θ is the parameter vector and $E(\theta)$ is the loss function[27].

$$\alpha_{m+1} = \beta \alpha_m, \quad \text{if } e = mD ,$$

The learning rate drop factor plays an important role in training the data. The intial learning rate is given by the user and it improves by multiplying throughout the training process. In the above equation, β is the learning rate drop factor for the performance of learning rate[27].

$$f_{fit} = (1 - A_{tav}) + (I - A_{vav}) ,$$

The fitness function plays an important role in determining the population of inputs and layers for the training options. It is flexible according to the size of the dataset and capacity of the network. In the above equation A_{vav} and A_{tav} the average accuracy values for next iteration is obtained.

Hence, we have seen the proof of mathematical concept of convolution neural network and it working mathematical operation and control via SGDM. This plays a crucial role in determining the network for the training purpose[28].

Chapter 4

Experimentation and Results

This chapter deals with the data preparation procedure and the cleaning process to obtain the usable data for the training purpose of the Faster RCNN that is useful for object detection. It explains in detail the work done for the project to achieve the required outcome and analysing the result in comparison to the expectation.

The experimentation can be classified into five categories-

- 2.1. Data preparation.
- 2.2. Preparing the training dataset through annotation.
- 2.3. Training the dataset.
- 2.4. Testing the network on an image or testing dataset.
- 2.5. Analysis of the outcome.

4.1 Data Preparation

Data preparation plays a crucial role in the field of machine learning and artificial intelligence. The vital part of the data analysis is cleaning the data that plays a significant role in data preparation than working on the algorithm. An experienced data scientist will work on cleaning the data than to working on the efficiency of the algorithm. The data that we collect here is the video data, which consists of drones in the frames. The video can be of any format, be it MP4, 3GP, WMV, OGG, WEBM, FLV, AVI, and many other formats. We

store this in a separate folder (02-Drone-video-original-MP4) using the Matlab coding. The obtained videos contain various frames of drone object during the training process. All the frames that have the potential to produce false detection rates are ignored. These frames are to be transferred manually using the free video cutter and stored in the separate folder (03-Drone-video-cut). The obtained segmented video frames are also in the (.Mp4) format.

The video frames are converted to the .jpg image files. The obtained image files are renamed using the Matlab code and stored in a separate folder (04-Drone-images). We have precisely received 6000 images in total. The captured images require to divide in the ratio of 80:20, where eighty per cent of the image dataset is part of the training dataset, and the remaining twenty per cent is for the testing purposes.

4.2 Preparing the training dataset through annotation

The image dataset resized to constant value of (224 x 224) pixels for application in Faster RCNN algorithm. The dataset is further loaded in an application of Matlab known as the Image Labeler. Image labeller does the job of annotating the training dataset. We use a rectangular bounding box for labelling the data.

Every image in the league of the training dataset is annotated one-by-one using the rectangular bounding box. The class that we create for our project is the 'Drone'. Any number of classes are created for

the session. The session must be saved in the regular periodic intervals to avoid using the annotated data. As the annotating tool is a rectangular bounding box, there will be precisely four labels available for each annotation. It is of the format [x-axis, y-axis, width, height]. The locations are stored as the pixel location of the particular image in the dataset. More than one annotation is possible per image in the Image labeller application in the Matlab. An image of object annotation using Matlab is shown in figure 8.



Figure 4.1: Annotated images of the training dataset

Once the annotation is complete for all the training images, the entire session is saved as the .mat file in the required directory. The labelled data is then exported to the file/workspace as the gTruth file, which gets saved as the .mat extension in the required folder. The gTruth file consists of the imageFileNames, and the 'Drone' annotated labelled data combined. The convert_python_format.mat file helps in converting the exported labels to the python-format, thereby gives the exact number of annotated data and its corresponding labels. The output of the .mat files produces a .txt file in the required location.

4.3 Training the dataset

To train the annotated dataset in the Matlab using the Faster RCNN, we use the function known as `trainFasterRCNNObjectDetector`. The `gTruth` obtained is used in the function `objectDetectortrainingData` that produces the training data that includes the `imageFileNames` and the 'Drone' for the labelled data. This training data is used to train the faster R-CNN detector. We also consider the training options and the network layer as the parameter for the training process. The training options include the optimiser required for the algorithm, execution environment, mini-batch size, momentum, initial learning rate, max epochs, verbose, and verbose frequency. Stochastic gradient descent momentum is optimal optimiser for our project. The faster RCNN does not plot the training and testing process. The confidence or accuracy levels measure the correctness of the training process.

The network used can be of a pre-trained network such as Resnet50, Resnet101, which requires large datasets. The number of layers we use in our project is 11, which consists of `ImageInputLayer`, `Convolution2DLayer`, `ReLULayer`, `MaxPooling2DLayer`, `FullyConnectedLayer`, `SoftmaxLayer`, and the `Classification OutputLayer`.

`PositiveOverlapRange` and the `NegativeOverlapRange` decide the probability of the target object in each frame of the training set. We provide a default value as per the algorithm. The max epochs can be set in the range of 7 up to 200 iterations, and the verbose is set to value 1. The verbose frequency is set in the range of 50 up to 200. The

execution environment can be set to 'CPU' as we run the program on the integrated graphics. The obtained trained data is stored in the variable known as a detector as per our Matlab coding. This data can be used for testing purposes on an image or the dataset containing of test images for our project. Proper execution of the Matlab code for the training purpose is shown in the figure 9 that describes the RPN and training of fast RCNN network. The re-training of RPN and fast RCNN is shown in figure. The retraining of fast RCNN provides the faster RCNN, and hence the detector is trained completely using the drone images.

4.4 Testing the Network with an Image

The images stored as the test dataset can be used to test the performance of the trained network. A random test image is read in the Matlab, and the detect function in the Matlab is used to predict the accuracy of the confidence score of the test image with the training dataset. The bounding box and the label are recorded and shown in the output screen. The algorithm is designed in such a way that it shows the maximum accurate score for the training performed on the dataset. The insertObjectAnnotation command in the Matlab is used to show the output of the detected object using the rectangular box or any other mode of detection in the particular function.



Figure 4.2: Test images getting detected by the Faster RCNN object detector. The detector shows fine results and can detect the objects for various views producing high accuracy rates

4.5 Analysis of the outcome

Faster RCNN is used for the object detection purposes in our project. The outcome of training is made to happen due to a series of convolutions. The automatic scaling of the network in the output layer has been an efficient feature in this algorithm. As the number of the training dataset increases, the machine learns the features by using the feature extraction through the region of interest align and region of interest pooling.

As the number of max epochs increases, the accuracy of the training images keeps developing. The RPN (Region proposal networks) works with all kinds of the region extraction by convoluting each pixel of an image. Many such proposals are formed by further extraction of the regions in the picture. The pixels are considered as an anchor

and compared with the original image by the reshaping process that occurs automatically through the training process. The NMS takes place to extract the feature of every anchor box along with the sliding window of each pixel corresponding to the neighbouring pixel. As a particular region matches the labels of the training data, the accuracy in the detection of the object improves with a low loss rate. Table 4.1 provides an insight of the training process and the results. The training provides an mAP of 94.83% with a base learning rate of 0.001. We can see that the loss rate goes down with the increase in number of epochs with increase in accuracy. This proves that the machine learns with constant training.

Epoch	Iteration	Mini-batch Loss	Mini-batch Accuracy	Mini-batch RMSE	Base Learning Rate
1	1	7.0819	39.37%	1.53	0.001
5	50	0.5545	93.65%	0.15	0.001
9	100	0.7400	100.00%	0.16	0.001
13	150	0.4625	100.00%	0.10	0.001
17	200	0.3414	100.00%	0.12	0.001
21	250	0.2449	100.00%	0.12	0.001
25	300	0.2109	100.00%	0.08	0.001
30	350	0.2507	99.81%	0.10	0.001
34	400	0.2568	100.00%	0.11	0.001
38	450	0.2752	100.00%	0.08	0.001
42	500	0.2231	100.00%	0.09	0.001
46	550	0.1755	100.00%	0.09	0.001
50	600	0.1887	100.00%	0.08	0.001

Table 4.1: The table shows the relevant table for mini-batch accuracy and loss values

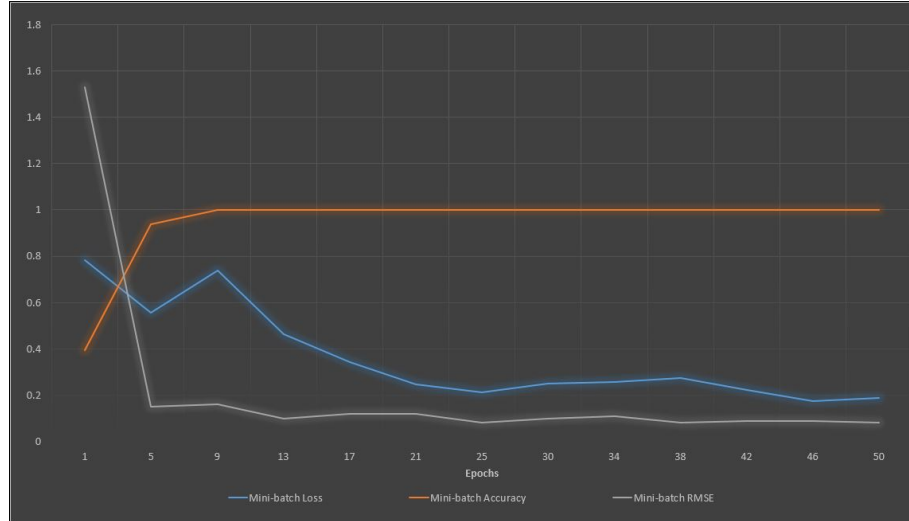


Figure 4.3: The graphical representation shows the performance analysis of Mini-batch accuracy, Mini-batch loss and Mini-batch RMSE

The Figure 4.3 represents the training process of Faster RCNN algorithm. It is seen that mini-batch accuracy stabilizes after 9 epochs and constantly improving before. A steady decrease in mini-batch loss is seen in the performance system which reflects the performance of the mini-batch RMSE(Root mean square error).

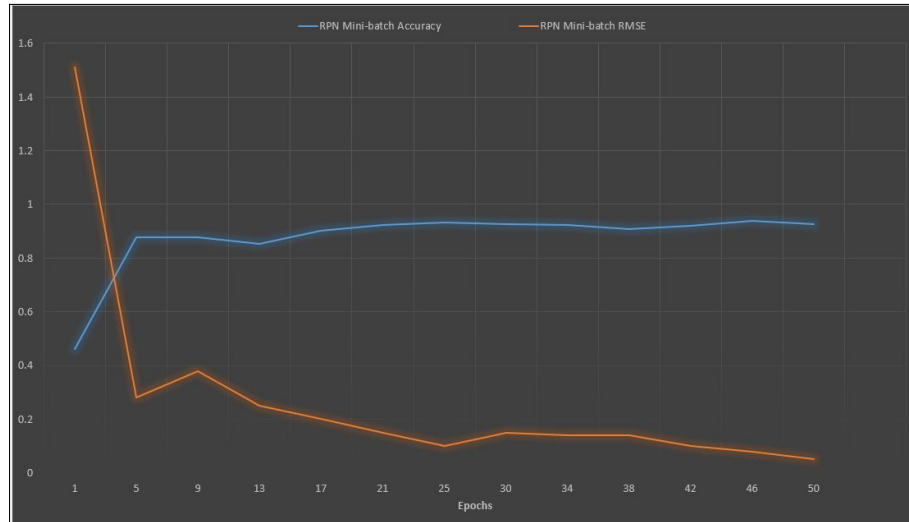


Figure 4.4: The graphical representation shows the performance analysis of RPN Mini-batch accuracy and RPN Mini-batch RMSE

The Figure 4.2 represents the performance analysis of the RPN layers prior performing the Faster RCNN algorithm on the training dataset.

Epoch	Iteration	RPN Mini-batch Accuracy	RPN Mini-batch RMSE	Base Learning Rate
1	1	46.12%	1.51	0.001
5	50	87.68%	0.28	0.001
9	100	87.80%	0.38	0.001
13	150	85.44%	0.25	0.001
17	200	90.23%	0.20	0.001
21	250	92.28%	0.15	0.001
25	300	93.40%	0.10	0.001
30	350	92.54%	0.15	0.001
34	400	92.27%	0.14	0.001
38	450	90.72%	0.14	0.001
42	500	92.01%	0.10	0.001
46	550	93.79%	0.08	0.001
50	600	92.75%	0.05	0.001

Table 4.2: The table shows the relevant table for RPN mini-batch accuracy and loss values

The table 4.2 shows that the accuracy of the performance constantly increases with increase in number of epochs and the root mean square error decreases on constant learning with a constant learning rate of 0.001. Similar to the mini-batch size accuracy the RPN accuracy rate stabilizes and does not decrease with the increase in number of epochs. This proves the postive learning of the system.

Chapter 5

Conclusion

Deep learning has been an essential concept in the field of image processing and computer vision. It has a great significance in the big data by exploring the specimen through the deep neural networks while creating various layers on the small-sized objects and small features. These are necessary for the classification and feature extraction techniques. It can explore the unstructured data and figure out the pattern of the specimen irrespective of scale and size. It can store the sequence of the pattern and provides the opportunity to perform the engineering in the sub-layers.

5.1 Research Summary

The Research works that has been reflected in the previous chapters are summarized as follows -

- We have conducted a literature review and explained in detail about the Region proposal networks, Intersection of Union and Faster RCNN algorithm developed to detect the drone in our thesis.
- We presented the suitable equations and explained in detail the network used for training the drone detection. A detailed study about the methodologies are presented that play a crucial role for the working of the detector. The network used in our thesis Resnet50 proves the efficiency of strengthen the layers to predict

the object with accuracy and precision.

- We conducted experiments on the data preparation and annotation methods using the MATLAB coding. The results are well explained and the analysis is made on every concept. Results of this method are analysed to prove the efficiency of the technique and algorithm. Hence, the state-of-art technique confirms the capacity and efficiency in the field of object detection.

5.2 Future Research

The outcome of our project detects the presence of the drone in the test set. The possible research work that can undergo a smooth transition for attaining higher efficiency using novel methods -

- Applying the hybrid model techniques to combine synthetic pre-trained layers with the existing layers in the network.
- Introducing segmentation techniques such as Mask RCNN and panoptic segmentation during the training process.
- Modifying the proposal layers to develop time efficient techniques.
- Improving the annotation style through speech recognition by applying Natural language processing.
- Novel methods of data annotation and data collection methods.

References

- [1] A. J. Kapoor, H. Fan, and M. S. Sardar, “Intelligent detection using convolutional neural network (id-cnn),” in *IOP Conference Series: Earth and Environmental Science*, vol. 234, 2019.
- [2] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, “A unified multi-scale deep convolutional neural network for fast object detection,” in *European conference on computer vision*. Springer, 2016.
- [3] A. Laishram and K. Thongam, “Detection and classification of dental pathologies using faster-rcnn in orthopantomogram radiography image,” in *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 2020.
- [4] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” *IEEE transactions on neural networks and learning systems*, vol. 30, 2019.
- [5] S. L. M. Oo and A. N. Oo, “Child face recognition with deep learning,” in *2019 International Conference on Advanced Information Technologies (ICAIT)*. IEEE, 2019.
- [6] Y. Ren, C. Zhu, and S. Xiao, “Object detection based on fast/faster rcnn employing fully convolutional architectures,” *Mathematical Problems in Engineering*, vol. 2018, 2018.
- [7] Q. Wang and F. Qi, “Tomato diseases recognition based on faster rcnn,” in *2019 10th International Conference on Information Technology in Medicine and Education (ITME)*. IEEE, 2019.
- [8] S. M. Abbas and S. N. Singh, “Region-based object detection and classification using faster rcnn,” in *2018 4th International Conference on Computational Intelligence & Communication Technology (CICT)*. IEEE, 2018.
- [9] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015.
- [10] M.-C. Roh and J.-y. Lee, “Refining faster-rcnn for accurate object detection,” in *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*. IEEE, 2017.
- [11] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Simultaneous detection and segmentation,” in *European Conference on Computer Vision*. Springer, 2014.
- [12] F. Bu, Y. Cai, and Y. Yang, “Multiple object tracking based on faster-rcnn detector and kcf tracker,” 2016.
- [13] S. Bell, C. Lawrence Zitnick, K. Bala, and R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.
- [14] J. Zheng, W. Li, M. Xia, R. Dong, H. Fu, and S. Yuan, “Large-scale oil palm tree detection from high-resolution remote sensing images using faster-rcnn,” in *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2019.
- [15] G. Li and Y. Yu, “Deep contrast learning for salient object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

- [16] B. Liu, W. Zhao, and Q. Sun, "Study of object detection based on faster r-cnn," in *2017 Chinese Automation Congress (CAC)*. IEEE, 2017.
- [17] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *The 25th Annual International Conference on Mobile Computing and Networking*, 2019.
- [18] Z. Zhang, C. Xu, and W. Feng, "Road vehicle detection and classification based on deep neural network," in *2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*. IEEE, 2016.
- [19] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based r-cnns for fine-grained category detection," in *European conference on computer vision*. Springer, 2014.
- [20] A. A. Micheal and K. Vani, "Automatic object tracking in optimized uav video," *The Journal of Supercomputing*, vol. 75, 2019.
- [21] L. Yan, Y. Wang, T. Song, and Z. Yin, "An incremental intelligent object recognition system based on deep learning," in *2017 Chinese Automation Congress (CAC)*. IEEE, 2017.
- [22] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of deep learning for object detection," *Procedia computer science*, vol. 132, pp. 1706–1717, 2018.
- [23] M. Lokanath, K. Sai Kumar, and E. Sanath Keerthi, "Accurate object classification and detection by faster-rcnn," in *Materials Science and Engineering Conference Series*, vol. 263, 2017.
- [24] S. Xiao, T. Li, and J. Wang, "Optimization methods of video images processing for mobile object recognition," *Multimedia Tools and Applications*, 2019.
- [25] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015.
- [26] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri, "diffgrad: An optimization method for convolutional neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2019.
- [27] P. Szymak, "Selection of training options for deep learning neural network using genetic algorithm," in *2019 24th International Conference on Methods and Models in Automation and Robotics (MMAR)*. IEEE, 2019.
- [28] E. Dogo, O. Afolabi, N. Nwulu, B. Twala, and C. Aigbavboa, "A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks," in *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*. IEEE, 2018.

Appendix A

Project Plan and Specifications

University of Wollongong



SCHOOL OF ELECTRICAL, COMPUTER AND TELECOMMUNICATIONS ENGINEERING ECTE940 –(session 2) Project Proposal Form

1. Candidate Details

Name: **Abhishek Balaji Sharma**Student No: **6237083**Supervisor: **Dr. Son Lam Phung**

Title of Project:
Camera based drone detection

Brief Overview:

In this project, we develop a camera-based system to detect the presence of drones flying in restricted areas. The project will involve collecting images and video of flying drones, annotating the image data, and training and testing a drone detector in MATLAB.

2. Project Description:

Drones have been an integral part of fields such as defense, military, sports and entertainment purposes. Drones play a key role in monitoring the activities in the unmanned locations but have also been a concern in drone restricted area. A drone detector will be designed to detect the presence of flying drones in the restricted area.

A large collection of drone-based images and video dataset will be collected and the image data will be annotated for the drones present in the video or image. Based on the processed dataset the drone detector will be trained and tested using the MATLAB. We use the Faster Regional Convolution Neural Network algorithm to train the images.

3. Project Plan:

Expected Outcomes

- A literature review of drone detection methods.
- MATLAB programs for training and testing a drone detector.
- A data set of images and video and the annotated ground-truth for drone detection.
- Project reports, presentations, and a poster.

Project Tasks

- Conduct a literature review on various drone detection methods.
- Collect images and video of flying drones, and annotate them.
- Develop MATLAB or Python programs for drone detection using Faster RCNN. Prepare reports, presentations and a poster for ECTE940.

Spring Session

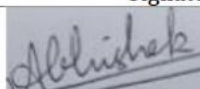
Week	1	2	3	4	5	6	7	8	9	10	11	12	13
Project Proposal													
Literature Review													
Understood the MATLAB tools													
Collected the drone samples													
Code for drone detection													
Prepared Spring Poster													
Prepared Spring Report													
Prepared Spring Presentation													

Autumn Session

Week	1	2	3	4	5	6	7	8	9	10	11	12	13
Revising the Project Proposal													
Revising the code													
Annotating the data													
Debugging the drone detector.													
Preparing Autumn Poster													
Preparing report for submission													
Preparing Autumn presentation													

4. Adaption of Supervisor and Examiners feedback in the ECTE940s1 report:

Feedback was received from the Supervisor and the Examiners post ECTE940s1 wherein I had to improvise the MATLAB program. I was asked to strictly follow the guidelines of report writing. I will have to work on the presentation of my output with the detailed parameters for the in-depth changes. The algorithm used is Faster RCNN and improvements will be made on the accuracy. The technical aspects of the coding which include the epoch size and mini-batch size will be modified for the improvements. A detailed report with the mathematical functions that plays a key role in the formation of a neural network will be added to the report.

Student Signature		
Declaration by the student: I have understood the feedback provided to me by the supervisor.		
	Signature	Date
Student Name:		05/04/20
ABHISHEK BALAJI SHARMA		

A marked assessment rubric will be appended once completed

Appendix B

MATLAB CODING

B.1 RESIZING THE TRAINING DATASET

```
1 srcFiles = dir('D:\ECTE940s2\04-Drone images\*.jpg'); % the folder in which ur images exists
2 for i = 1 : length(srcFiles)
3     filename = strcat('D:\ECTE940s2\04-Drone images\',
4         srcFiles(i).name);
5     im = imread(filename);
6     k=imresize(im,[224,224]);
7     newfilename=strcat('D:\ECTE940s2\04-Drone images\',
8         ,srcFiles(i).name);
9     imwrite(k,newfilename,'jpg');
10 end
```

B.2 NETWORK LAYER FOR RESNET50

```
1 % Load a pretrained ResNet-50.
2 net = resnet50;
3 lgraph = layerGraph(net);
4 % Remove the last 3 layers.
5 layersToRemove = {
6     'fc1000',
7     'fc1000_softmax',
8     'ClassificationLayer_fc1000',
9     };
10
```



```
10 lgraph = removeLayers(lgraph, layersToRemove);
11
12 % Specify the number of classes the network should
    classify.
13 numClasses = 1;
14 numClassesPlusBackground = numClasses + 1;
15
16 % Define new classification layers.
17 newLayers = [
18     fullyConnectedLayer(numClassesPlusBackground,
        'Name', 'rcnnFC')
19     softmaxLayer('Name', 'rcnnSoftmax')
20     classificationLayer('Name', '
        rcnnClassification')
21 ];
22 % Add new object classification layers.
23 lgraph = addLayers(lgraph, newLayers);
24
25 % Connect the new layers to the network.
26 lgraph = connectLayers(lgraph, 'avg_pool', 'rcnnFC
    ');
27
28 % Define the number of outputs of the fully
    connected layer.
29 numOutputs = 4 * numClasses;
30
```

```
31 % Create the box regression layers.
32 boxRegressionLayers = [
33     fullyConnectedLayer(numOutputs, 'Name', '
        rcnnBoxFC')
34     rcnnBoxRegressionLayer('Name', 'rcnnBoxDeltas')
35 ];
36
37 % Add the layers to the network.
38 lgraph = addLayers(lgraph, boxRegressionLayers);
39
40 % Connect the regression layers to the layer named
    'avg_pool'.
41 lgraph = connectLayers(lgraph, 'avg_pool', '
    rcnnBoxFC');
42
43 % Select a feature extraction layer.
44 featureExtractionLayer = 'activation_40_relu';
45
46 % Disconnect the layers attached to the selected
    feature extraction layer.
47 lgraph = disconnectLayers(lgraph,
    featureExtractionLayer, 'res5a_branch2a');
48 lgraph = disconnectLayers(lgraph,
    featureExtractionLayer, 'res5a_branch1');
49
50 % Add ROI max pooling layer.
```

```
51 outputSize = [14 14];
52 roiPool = roiMaxPooling2dLayer(outputSize, 'Name', '
    roiPool');
53 lgraph = addLayers(lgraph, roiPool);
54
55 % Connect feature extraction layer to ROI max
    pooling layer.
56 lgraph = connectLayers(lgraph,
    featureExtractionLayer, 'roiPool/in');
57
58 % Connect the output of ROI max pool to the
    disconnected layers from above.
59 lgraph = connectLayers(lgraph, 'roiPool', '
    res5a_branch2a');
60 lgraph = connectLayers(lgraph, 'roiPool', '
    res5a_branch1');
61
62 % Define anchor boxes.
63 anchorBoxes = [
64     16 16
65     32 16
66     16 32
67 ];
68
69 % Create the region proposal layer.
```

```
70 proposalLayer = regionProposalLayer(anchorBoxes, '  
    Name', 'regionProposal');  
71  
72 lgraph = addLayers(lgraph, proposalLayer);  
73 % Number of anchor boxes.  
74 numAnchors = size(anchorBoxes,1);  
75  
76 % Number of feature maps in coming out of the  
    feature extraction layer.  
77 numFilters = 1024;  
78  
79 rpnLayers = [  
80     convolution2dLayer(3, numFilters, 'padding', [1  
        1], 'Name', 'rpnConv3x3')  
81     reluLayer('Name', 'rpnRelu')  
82 ];  
83  
84 lgraph = addLayers(lgraph, rpnLayers);  
85  
86 % Connect to RPN to feature extraction layer.  
87 lgraph = connectLayers(lgraph,  
    featureExtractionLayer, 'rpnConv3x3');  
88 rpnClsLayers = [  
89     convolution2dLayer(1, numAnchors*2, 'Name', '  
        rpnConv1x1ClsScores')  
90     rpnSoftmaxLayer('Name', 'rpnSoftmax')
```

```
91     rpnClassificationLayer( 'Name', '
        rpnClassification' )
92 ];
93 lgraph = addLayers(lgraph, rpnClsLayers);
94
95 % Connect the classification layers to the RPN
    network.
96 lgraph = connectLayers(lgraph, 'rpnRelu', '
        rpnConv1x1ClsScores' );
97
98 rpnRegLayers = [
99     convolution2dLayer(1, numAnchors*4, 'Name', '
        rpnConv1x1BoxDeltas' )
100     rcnnBoxRegressionLayer( 'Name', 'rpnBoxDeltas' )
        ;
101 ];
102
103 lgraph = addLayers(lgraph, rpnRegLayers);
104
105 % Connect the regression layers to the RPN network
    .
106 lgraph = connectLayers(lgraph, 'rpnRelu', '
        rpnConv1x1BoxDeltas' );
107
108 % Connect region proposal network.
```

```
109 lgraph = connectLayers(lgraph, '
    rpnConv1x1ClsScores', 'regionProposal/scores');
110 lgraph = connectLayers(lgraph, '
    rpnConv1x1BoxDeltas', 'regionProposal/boxDeltas'
    );
111
112 % Connect region proposal layer to roi pooling.
113 lgraph = connectLayers(lgraph, 'regionProposal', '
    roiPool/roi');
```

B.3 TRAINING DATASET

```
1
2 % Loading the data required for the training
    purpose.
3 data = load('sample_drone.mat');
4 trainingData = data.trainingData;
5 %trainer = int8(imresize(trainingData,[224 224 3])
    );
6 %
7 %
8 %% Randomly shuffling the training data before
    the training procedure.
9 rng(0);
10 shuffledIdx = randperm(height(trainingData));
11 trainingData = trainingData(shuffledIdx,:);
12 %
```

```
13 %% Sets up the required layers for training the
    data.
14 %lgraph = layerGraph(data.dronedata.detector.
    Network);
15 %
16 %
17 %% Training options required for training data.
    The program runs on the CPU
18 %% and plot is obtained.
19 options = trainingOptions('sgdm', ...
20     'Momentum', 0.9000, ...
21     'ExecutionEnvironment', 'auto',...
22     'MiniBatchSize', 5, ...
23     'InitialLearnRate', 1e-3, ...
24     'MaxEpochs', 50, ...
25     'VerboseFrequency', 50, ...
26     'Verbose', 1, ...
27     'CheckpointPath', tempdir);
28 %
29 %
30 %% Faster RCNN is trained using a set of
    parmeters
31 [detector, info] = trainFasterRCNNObjectDetector(
    trainingData, lgraph, options, ...
32     'NegativeOverlapRange', [0 0.2], ...
33     'PositiveOverlapRange', [0.6 1]);
```

B.4 TESTING DATASET

```
1 % Reading a particular image from the test dataset
2 img = imread('D:\laptop\laptops_test\9.jpg');
3
4 % detect matlab command is used to predict the
   confidence level of region
5 % inside the boundary box
6 [bbox, score, label] = detect(detector, img);
7
8 % Picking out the maximum score and applying
   inverse
9 [score, idx] = max(score);
10 bbox = bbox(idx, :);
11
12 % Printing the output on the object by inserting
   object annotation
13 annotation = sprintf('%s: (Accuracy = %f)', label(
   idx), score);
14 detectedImg = insertObjectAnnotation(img, '
   rectangle', bbox, annotation);
15 figure
16 imshow(detectedImg) % Displaying the result
```