MOVING OBJECT DETECTION IN VEHICLE MOUNTED DASHCAM VIDEOS

M.Tech. Thesis

By

HIMANSHU CHAUHAN

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2019

MOVING OBJECT DETECTION IN VEHICLE MOUNTED DASHCAM VIDEOS

A THESIS

Submitted in partial fulfillment of the
requirements for the award of the degree
of
Master of Technology

by

HIMANSHU CHAUHAN

DISCIPLINE OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE
JUNE 2019

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **MOVING OBJECT DETECTION IN VEHICLE MOUNTED DASCAM VIDEOS** in the partial fulfillment of the requirements for the award of the degree of **MASTER OF TECHNOLOGY** and submitted in the **DISCIPLINE OF ELECTRICAL ENGINEERING, Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from July 2018 to June 2019 under the supervision of Dr. Vivek Kanhangad, Associate Professor, IIT Indore.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

Signature of the student with date
(HIMANSHU CHAUHAN)

This is to certify that the above statement made by the candidate is correct to the best of my/our knowledge.

Signature of the Supervisor of

(Dr. VIVEK KANHANGAD)

HIMANSHU CHAUHAN has successfully given his/her M.Tech. Oral Examination held on_____.

Signature of Supervisor of M.Tech. thesis
Date:

Convener, DPGC
Date:

Signature of PSPC Member #1
Date:

Signature of PSPC Member #1
Date:

ACKNOWLEDGEMENTS

I express my sincere gratitude to my guide Dr. Vivek Kanhangad for the continuous support, for his patience, motivation, and immense knowledge. His guidance helped me in conducting the research and writing of this thesis.

I would like to thank the members of my thesis committee: Dr. Srivathsan Vasudevan and Dr. Somnath Dey for their insightful comments and encouragement. Their questions and suggestions helped me to widen my knowledge from various perspectives. I am thankful to all the faculty members of the Department of Electrical Engineering for their guidance and support.

I express my gratitude to members of the Pattern Recognition and Image Analysis lab (PRIAL) group to provide me such a platform to learn various software related to image processing. I am very thankful to Vijay and Shishir, who contributed significantly to my thesis work. I must also thank my fellow batch mates and all other friends for making my stay at IIT Indore delightful.

Last but not least, I also thank my family for their unceasing encouragement and support throughout this journey.

HIMANSHU CHAUHAN

M.Tech. (Communication and Signal Processing)
mt1702102003
Discipline of Electrical Engineering, IIT Indore

ABSTRACT

Moving object detection has been an active research area for many years. In spite of the advancements made in this area, moving object detection in moving cameras still remains a challenging problem due to the introduction of camera motion along with the motion of the object. In this work, we propose a CNN based method to detect moving objects in dashcam videos. The feature which we use to identify the moving objects is optical flow. First, we use the Flownet to extract the optical flow RGB images from two consecutive video frames. We then feed the obtained optical flow based RGB image to Faster-RCNN model. This model then identifies the target objects in the video. We have used two datasets, namely KITTI dataset and BDD-mod dataset, for the evaluation of the method.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

Abbreviations	Description
BS	Background Subtraction
UAVs	Unmanned aerial vehicles
RGB	Red Green Blue
HSV	Hue Saturation Value
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
FC	Fully Connected
ROI	Region of Interest
R-CNN	Region Convolutional Neural Network
YOLO	You Look Only Once
RPN	Region Proposal Network
BDD	Berkeley Deep Drive
FPS	Frames Per Second
mAP	Mean Average Precision

Chapter 1

INTRODUCTION

1.1 Overview

Moving object detection refers to the identification of moving objects in the surroundings of a moving observer. It is essential that a moving viewer can identify his / her surrounding object and determine if these objects are moving or stationary to know their situation and contact-free navigation in the surrounding. Detecting moving objects from moving cameras is a challenging problem to solve due to the introduction of double motion, i.e., the motion of the moving observer and movement of the surrounding object. It is part of many computer vision-based problems such as traffic control [1], recognizing action [2], behavior identification [3], industrial inspection [4], and many more. Attaining high detection accuracy and computational efficiency is important for such a task because they are usually used as a basis for other high-level tasks such as behavior or event analysis.

In the dashcam video, for braking and smooth movement of the observer detection of moving objects in the surrounding is the most critical. Initial approaches in motion detection were centered near geometry-based approaches [5-7]. However, there are many limitations of geometry-based approaches such as motion parallax issue. A recent trend [8-10] for learning motion in moving cameras has emerged, which was based on the pixel-wise motion segmentation. We have used detection-based approach for identifying the moving objects instead of detecting the moving pixels.

1.2 Related work

Initially, motion detection was studied on still cameras for years [11-13], which are based on various concepts like image difference [12], background subtraction (BS) [13], and optical flow [11]. These approaches are not only applicable on still cameras but also fail to deliver for moving camera applications such as autonomous car, unmanned aerial vehicles (UAVs) and mobile robots. The introduction of dual motion, varying and complex background is the main reasons behind this.

In recent years, motion in moving cameras is also studied [14], [15], [16-18]. These methods use motion segmentation to detect the moving objects in moving cameras. Motion segmentation divides the image into two parts moving region (moving objects) and stationary region (background) [19]. There are two main approaches.

1.2.1 Optical flow-based approaches

Optical Flow is an important feature to detect the motion in computer vision problems. This type of approach uses the irregularity in the radial optical flow of the target object from that of the surrounding objects [14], [15], [16]. In particular, it analyzes the consistency of the speed and direction of motion with the background. More difference between the optical flow vector of the background and object refers to a high probability that the object is moving. Some simple threshold methods are based on this strategy [14], [15], [16]. These types of approaches are straightforward and efficient. On the other hand, they lack consistency and highly prone to occlusion, noise, or color changes.

Figure 1: A moving object in an image with the bounding box.
Image Source: [20]

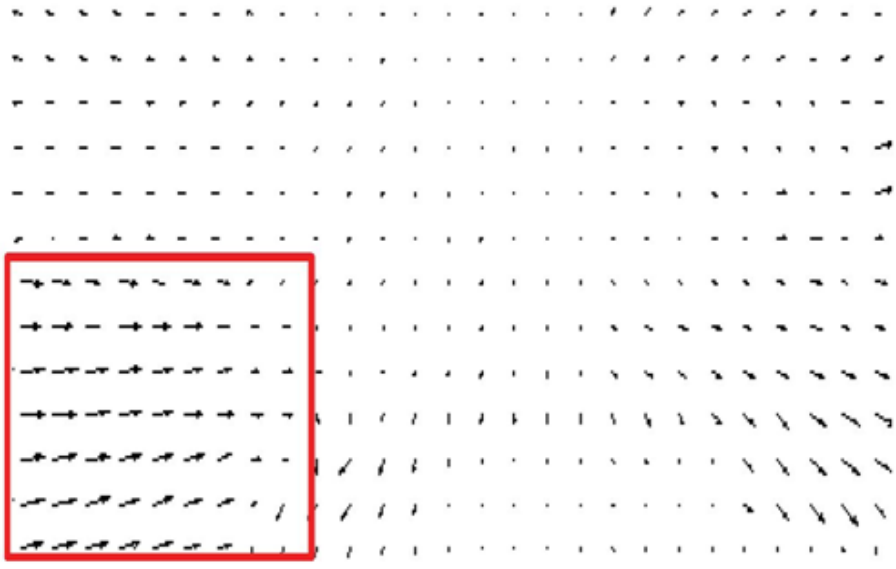

Figure 2: Optical flow deviation of Figure 1.
Image Source: [20]

1.2.2 Adaptive background modeling based approaches

These kinds of approaches use modeling of the background to calculate the motion vectors for every frame and then use image difference or BS between consecutive video frames [17], [18], [21], [22]. Such

approaches are complex due to the modeling of each frame and fail to deliver when there are abrupt changes in the background. Hence, these approaches have low estimation efficiency when the background is dynamic and high computations are involved work due to the background model, transformation matrix calculations, and feature-matching operations.

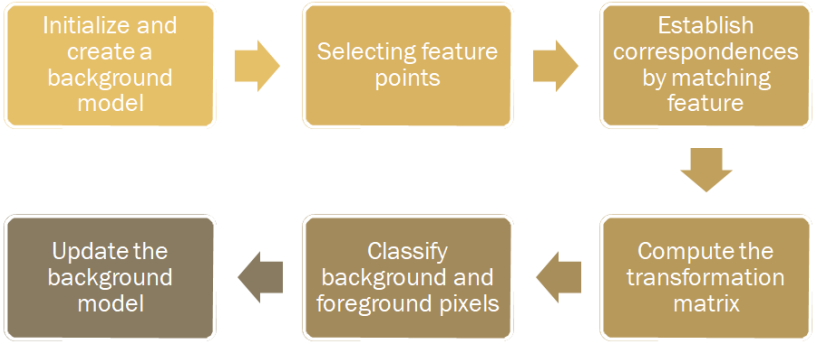

Table 1: Flow Chart of the Adaptive Background Modeling based method.

Another disadvantage of the above-mentioned approaches is that they only label all the pixels in the image without using the semantic label. Due to which other information such as the total number of object and their category is unknown, which restricts their application. Other related works are based on tracking [23-26], which targets the detection of the tracked object in each frame.

1.3 Difficulties in detecting moving objects

There are many difficulties in making a real-time robust model for this problem. In this section, we will discuss the difficulties faced in detection of moving objects in moving cameras. Some of these challenges are also faced in still the camera moving object detection. The difficulties are as follows:

1.3.1 Illumination variation

Illumination refers to light. Hence, illumination variation refers to the changing lighting conditions in the scene. The appearance of the target might change due to the presence of changing lightning condition at various times of day, reflection in mirrors, weather conditions, night condition, etc. Due to this background appearance of the scene changes making the background modeling a difficult task. Thus, background modeling based approaches must adapt to these changes due to illumination variation. Meanwhile, the efficiency of tracking-based methods also reduces because of the appearance of the object changes. Hence, it is recommended to use the features which have less effect of illumination changes. In some works, various color spaces are used to tackle this problem.

Figure 3: Illumination in a frame.
Image Source: [27]

1.3.2 Abrupt motion

Abrupt changes in the orientation of movement like turning, speed, etc. pose a significant challenge to the problem of moving object detection. If the target object has very low speed, sometimes optical flow based approaches fail to detect it. Hence speed and direction changes are important parameters and pose a significant challenge in object detection. Meanwhile, a trail of the detected object's pixel produces

when very fast motion is present. Therefore, if such type of motion is not considered, we can not detect the object using background modeling based method.

(A)

(B)

Figure 4: Abrupt motion due to turning between consecutive frames (A) and (B). Image Source: [27]

Presence of abrupt motion in a scene is unpredictable which makes the task of moving object detection challenges. Due to unpredictability, no robust solutions are present to tackle this problem.

1.3.3 Occlusion

Occlusion means blockage or barrier. This challenge refers to detecting the target object when it gets blocked by the presence of some other objects in the scene. In some case, the partial object is hidden or occluded by others (partial occlusion) while in other cases, the target object can be entirely concealed by other objects (complete occlusion).

Figure 5: Problem of occlusion in a frame.
Image Source: [27]

For example, a moving vehicle can get occluded by trees, pedestrian, other vehicles present in the scene, etc. Occlusion has severe effects on object detection problem. Background modeling based models are highly affected by the problem of occlusion. Optical flow based methods also fail to detect a completely occluded object. Tracking of objects also becomes difficult as the appearance model of the target object changes due to occlusion. Partial occlusion problem can be tackled but tackling complete occlusion is very challenging in the real-life scenarios. Using spatiotemporal information along with good appearance model minimize the effect of occlusion.

1.3.4 Complex background

Complex background refers to the presence of different textures in the scene. There are high variations in textures of natural outdoor scenes.

Moreover, sometimes the background can be dynamic or movements are present in the scene due to tree waggle, traffic light, rain, cloud movement, etc. which should be considered in accordance with the problem while working for an algorithm related to object detection. Complex background problem affects both optical flow based and background modeling based approaches.

Figure 6: Complex background due to the presence of water on the road.
Image Source: [27]

1.3.5 Shadow

Moving object detection gets more complicated by the presence of shadows. Shadows occur due to blockage of the light sources by the object. If the target object is not moving i.e. formed shadow is non-moving, then background modeling is possible. However, if a shadow is moving in the background, then it is difficult to model the shadow and algorithm fails to detect the objects effectively as pixels of shadow are tightly connected with the object. By using properties such as edges, color, and texture, shadows can be often removed from frames. However, moving shadows still requires efforts to be differentiated from target moving objects, especially in the case of an outdoor environment (complex background).

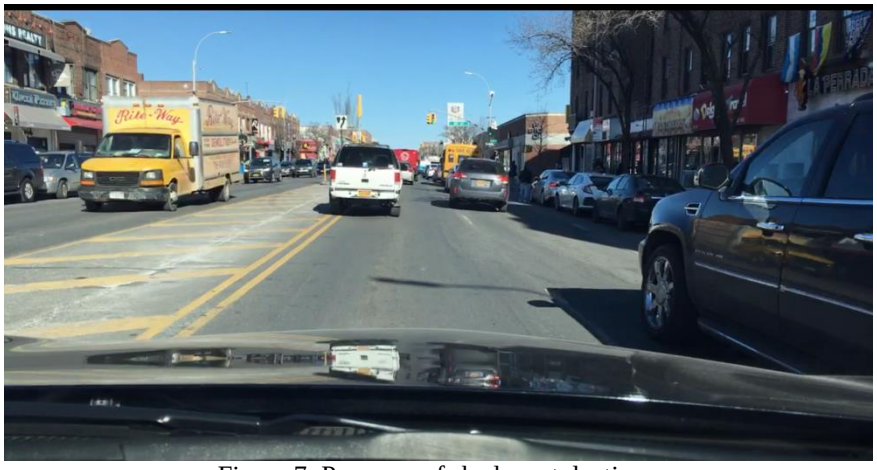

Figure 7: Presence of shadow at daytime.
Image Source: [27]

1.3.6 Non-rigid objects

Non-rigid object is the class of objects that suffer deformation or changes its appearance over time. Different parts of non-rigid objects might have separate movements such as speed, direction, and position. For example, a dog is walking and wagging his tail. In the example, the motion of a dog's tail, body, and legs is dissimilar. When trying to detect the motion in non-rigid objects, most algorithms predict parts with dissimilar motion as different moving objects. Hence, non-rigid objects pose a significant challenge to the problem of moving object detection in moving cameras.

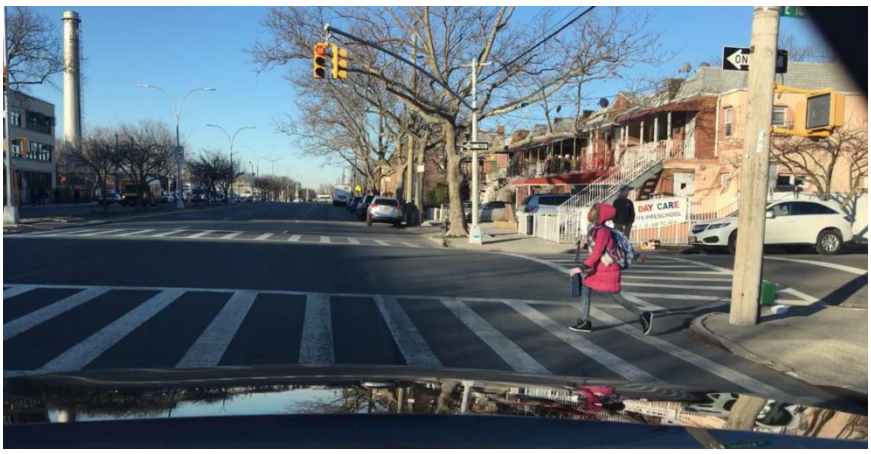

Figure 8: Presence of pedestrian in a frame.
Image Source: [27]

1.4 Organization of the report

This chapter has introduced the background, motivation of the thesis and related work done by researchers in the past. The remaining contents are organized as follows:

- **Chapter 2:** This chapter provides detail about the fundamentals used further in the thesis. Section 2.1 covers the color spaces. In section 2.2, optical flow is covered and section 2.3 covers the basics of CNN.
- **Chapter 3:** This chapter includes a detailed description of some object detection and segmentation algorithms used in CNN such as R-CNN, Fast-RCNN, Faster-RCNN, and Encoder-Decoder based architecture.
- **Chapter 4:** In this chapter, a detailed description of the pipeline structure is presented. In section 4.2 we cover the CNN architecture that we have used.
- **Chapter 5:** This chapter includes the description of the database used and results along with some implementation details. Also, a detailed description of the performance details is included.
- **Chapter 6:** In this chapter, conclusions are made and a discussion on the possibility of future work is presented.

Chapter 2

FUNDAMENTALS

In this chapter, we discuss the techniques that we have used in this thesis. In section 2.1, we provide the details about different color spaces used in image processing such as RGB, HSV, etc. In section 2.2, the concept of optical flow is discussed. The mathematical equation of the optical flow and the different type of optical flow are discussed. In the section 2.4, we present the fundamentals of CNN and how CNN is formed.

2.1 Color space

Different types of color modes used to represent every color are known as the color spaces, used in image processing for different purposes. Some widely used color spaces are RGB, YUV, YCbCr, and HSV.

2.1.1 RGB

1. RGB color space is a model of additive color.
2. RGB stands for red, green and blue which are primary colors, are added together to form a wide variety of colors.
3. Device dependent, meaning that the color is dependent on the device that detects or output the color.
4. Organized in a unit cube with coordinates (r, g, b), where (0, 0, 0) is black and (1, 1, 1) is white.
5. Cube model of RGB color space is shown in the Figure 9.

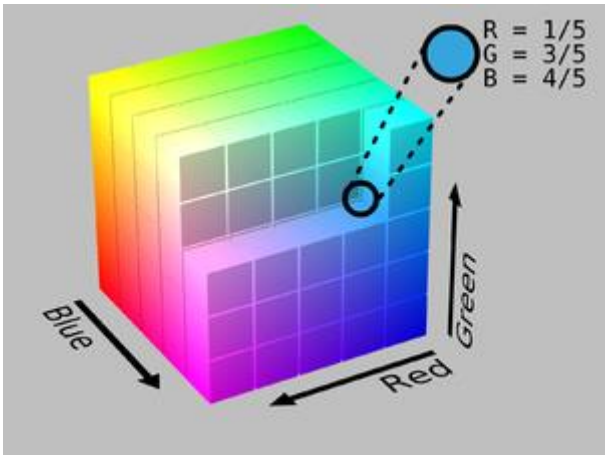

Figure 9: RGB color cube model.
Image Source: [28]

2.1.2 HSV

Unlike RGB, which has three primary colors, HSV is similar to how humans perceive color. HSV stands for hue, saturation, and value. This color space identifies colors (hue) in the form of their shade (the amount of gray or saturation) and their brightness value.

HSV color model is commonly represented as a cone (Figure 10) or cylinder with the three components:

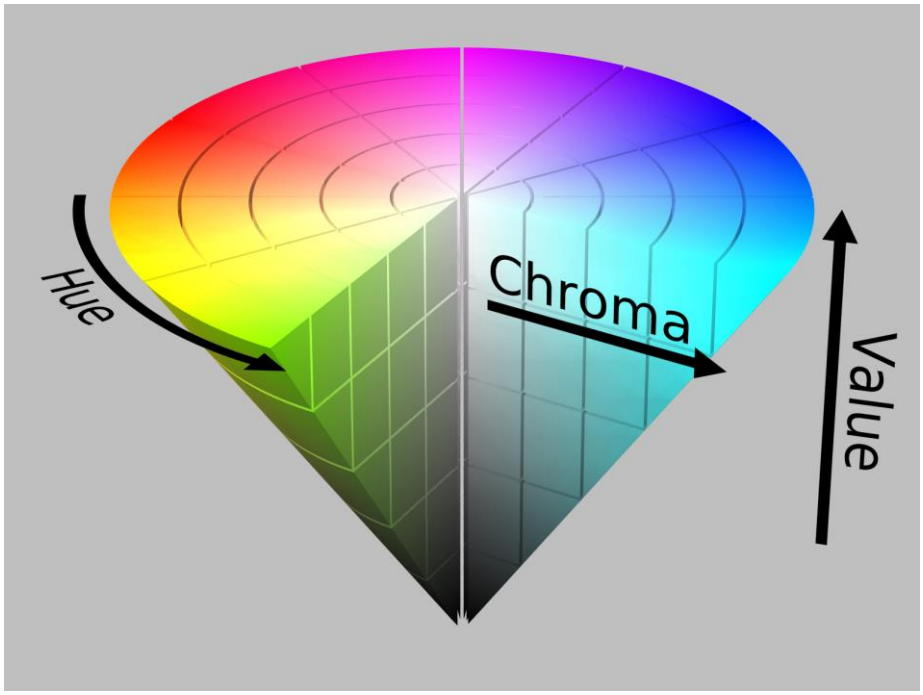

Figure 10: HSV cone model
Image Source: [29]

Hue: In the model, the color portion, which is perceived is represented by Hue. A hue is a number ranging from 0 to 360 degrees.

Saturation: Saturation is defined as the colorfulness of a stimulus relative to its own brightness. It shows the amount of grayness present in color. Its value ranges from 0 to 100 percent. The amount of gray is maximum when the value of saturation is zero. In some models, saturation is expressed in a range from just 0-1, where 1 is a primary color and 0 is gray.

Value: Value is connected with the saturation and expresses the brightness or intensity of the color. Its value ranges from 0-100 percent, where 0 is black, and 100 is the brightest color.

2.2 Optical flow

In a visual scene, optical flow or optic flow reflects the movement pattern of objects, surfaces, and edges. Optical flow recovers image motion at each pixel from spatiotemporal image brightness variations. It represents the velocities of brightness patterns in 2D form, an example is shown in Figure 11.

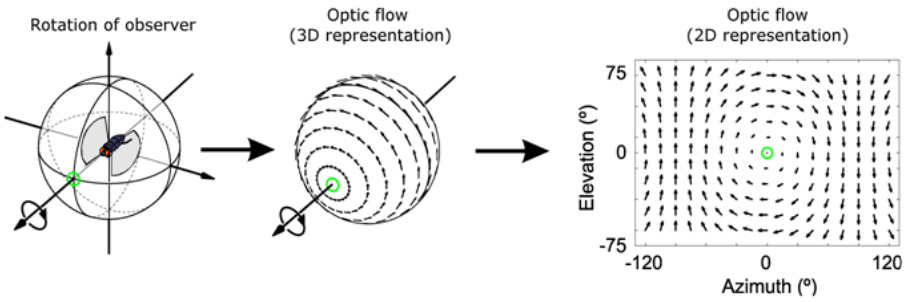

Figure 11: Optical flow experienced by a rotating observer.
Image Source: [30]

Figure 12 shows us the displacement of the brightness pattern between two consecutive frames. The image intensity (I) of a frame can be expressed as the function of spatial coordinates (x, y) and time(t). Let us consider the first image at time t i.e. I (x, y, t) and second frame at the time instant $t + dt$. If the coordinate of the pixel in the second

frame is displaced by (dx, dy), then the second frame can be represented as I$(x + dx, y + dy, t + dt)$.

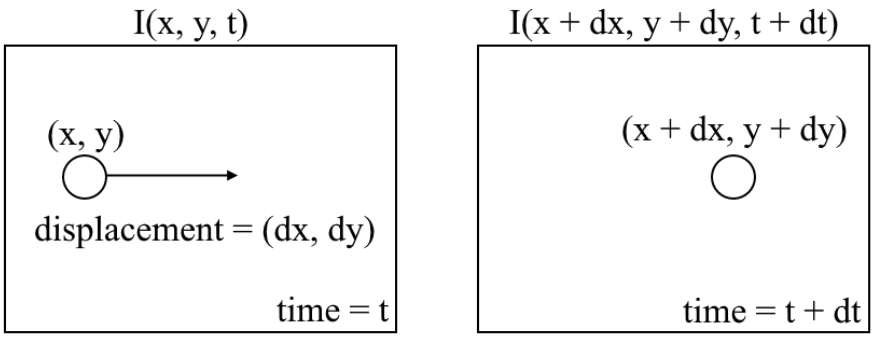

Figure 12: Displacement of a pixel in two frames.
Image Source: [31]

Assuming that pixel brightness will be the same, we get

$$I(x, y, t) = I(x + dx, y + dy, t + dt)$$

which, represents the basic optical flow equation.

Optical flow is of two types namely, Sparse and Dense optical flow.

Sparse optical flow: It gives the flow vectors at some fixed pixels (selected features such as corners or edges of the object) inside the frame. Sparse optical flow is fast and easy to calculate i.e. computationally efficient.

Dense optical flow: It gives the flow vectors at each and every pixel of the frame (one flow vector per pixel). Hence, Dense optical flow is more accurate.

Figure 13 shows the sparse (left) and dense optical flow (right). The optical flow is calculated at fixed pixels (left) and all pixels (right).

Figure 13: Sparse (left) and Dense (right) Optical Flow
Image Source: [31]

2.3 CNN

CNN refers to convolution neural network, is a class of deep neural networks which processes the data having grid-like topology such as an image. The concept behind the origin of CNN is the receptive field [32]. The receptive field is a term in biology, refers to a characteristic of the animal visual cortex [33]. Receptive field detects or senses a certain type of stimulus, such as edges. They can be found anywhere in the visual field and overlap each other. In computers, this function is close to the convolution operation.

The practical models of CNN typically consist of many layers, and each layer is divided into multiple stages, namely convolutional layer, a rectified linear function and max or average pooling. Figure 14 shows a typical CNN with different layers. The basic operations of CNN are explained further in the subsections below.

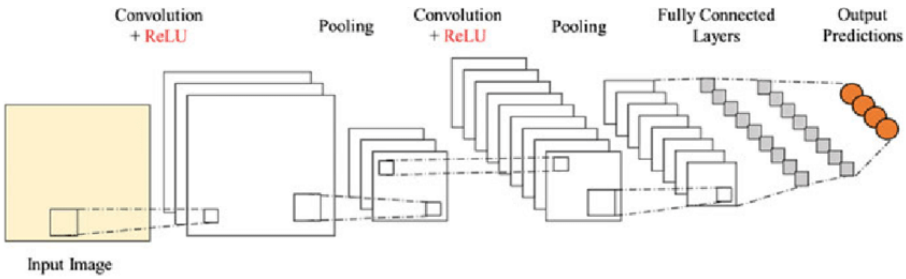

Figure 14: Typical CNN structure
Image Source: [34]

2.3.1 Convolution

Convolution is the basic building block of the CNN. It is a mathematical operation between an input image matrix (f) and a filter or kernel(g), defined as

$$h[x,y] = f[x,y] * g[x,y] = \sum_{n}\sum_{m} f[n,m]g[x-n,y-m]$$

Convolution operations can be used to perform edge detection, image sharpening or blurring, etc. by using different kernels. The output of

the convolution operation is also called a feature map in neural networks. Convolution conserves the connection among pixels by learning features of the input image.

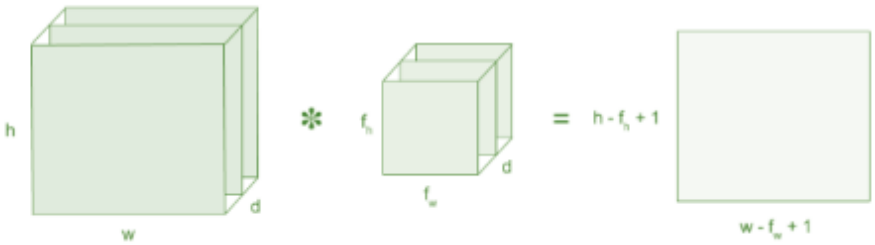

Figure 15: Convolution Operation
Image Source: [35]

Figure 15 shows an input image matrix with dimension $h \times w \times d$ and an image filter with dimension $f_h \times f_w \times d$ then the output of the convolution will be of dimension $h - f_h + 1 \times w - f_w + 1 \times d$.

Since the kernel filters used for convolution operation is the same for each pixel of the input, the unknown parameters are less as compared to a fully-connected neural network. The concept of sharing the same parameters in convolution operation is called Parameter sharing. Due to the concept of parameter sharing the number of calculations reduces significantly and translation invariance is achieved for convolution operations.

2.3.2 Pooling and stride

To reduce the number of computations involved with increasing layers of the network, the feature map must be decreased at the deep end of the network. By reducing the dimensions (height and width) of the feature volume, we can increase the number of layers in the network with the maintenance of computation time at a reasonable level. We can reduce the feature volume by the following two ways:

The first way is the addition of pooling layers. A pooling layer reduces the feature map effectively. Pooling makes the output network more translation invariant. It reduces the precision of detectors. There are many methods by which pooling can be performed such as average pooling, max pooling, etc. The output of the average pooling is the average of values of input inside the filter. While the output of max-pooling is the maximum value of the input features present inside filter. An example of max-pooling operation is shown in the Figure 16.

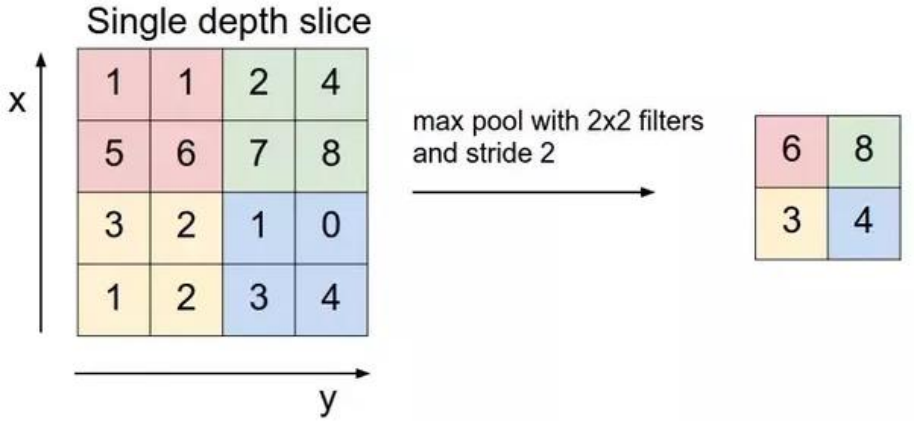

Figure 16: Max pooling operation
Image Source: [35]

The second way to reduce the output feature matrix dimension is changing the stride parameter of the convolution operation. Stride parameter refers to the number of pixels shifts over the input frame. If the stride is 1 then the kernel filter is shifted by one pixel over input frame at a time. If the stride is 2 then the kernel filter shifts by two pixels over input image at a time and so on. Figure 17 shows the convolution operation with a stride of 2. Recently, researches have shown we can achieve the same accuracy by discarding pooling layer and using convolutional layers with larger stride value [36].

2.3.3 Activation function

The activation function is a non-linear transformation of the input signal. Activation Functions are used to introduce the non-linearity in

CNN as all other operations are linear. The activation function is of many types as shown in Figure 18.

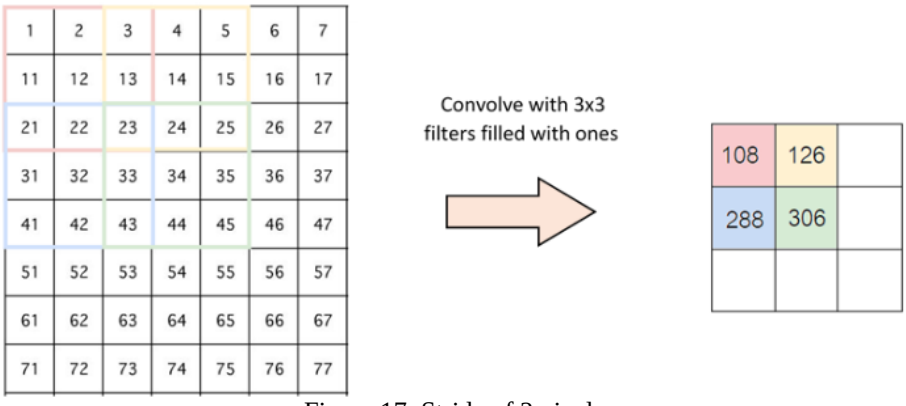

Figure 17: Stride of 2 pixels
Image Source: [35]

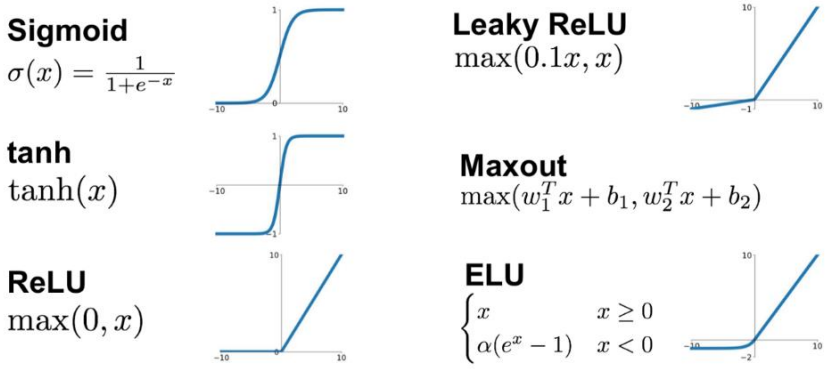

Figure 18: Different Activation Functions
Image Source: [37]

The most widely used activation function in deep learning today is ReLU function. ReLU is used over other activation functions because it does not activate all neurons at the same time. From Figure 18, we can notice that the output of ReLU is zero for negative input values and hence, the neuron does not get activated. As all the negative values are converted to zero, makes the ReLU computationally efficient and fast. In the practical scenario, ReLU is six times faster than sigmoid and tanh activation functions.

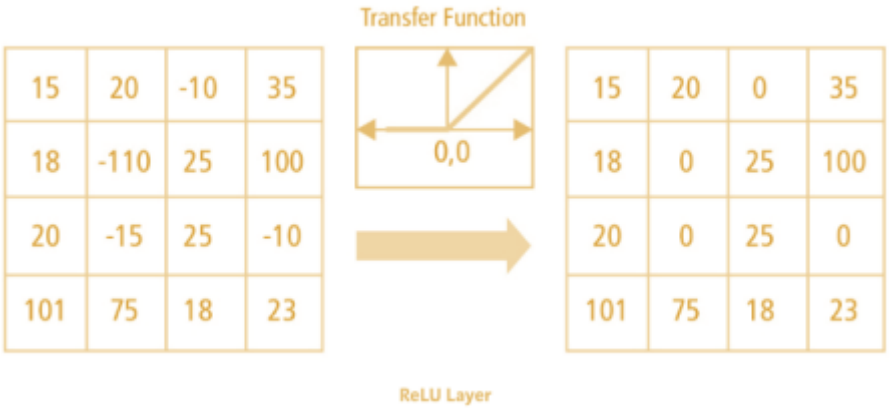

Figure 19: ReLU operation
Image Source: [35]

2.3.4 Fully connected (FC) layer

The last pooling layer's output is linked to the FC layer. In the FC layer, each output node is connected to all input nodes (fully connected).

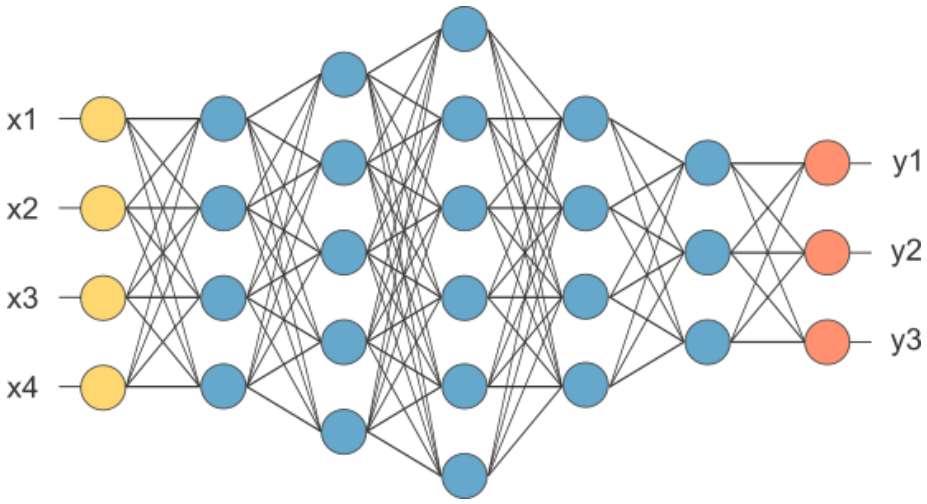

Figure 20: FC layers
Image Source: [35]

In the above diagram, the output of the last pooling layer is converted to vectors or linear features (x1, x2, x3, …). By the help of FC layers, all features are shared with each other. Finally, activation functions (softmax or sigmoid) are used to categorize the outputs into various classes.

Combining the above-discussed layers and functions, provide the final architecture of the CNN. Complete CNN architecture is shown in Figure 21. The input image is provided to the convolution layer. Then, pooling or stride is used to reduce the dimensionality and ReLU activation is applied. At the output, the FC layer and activation functions are used.

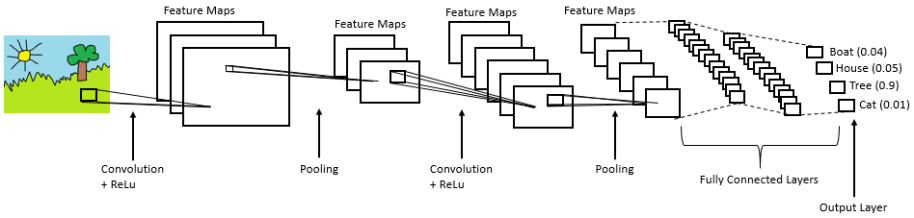

Figure 21: Complete CNN Architecture
Image Source: [35]

Chapter 3

OBJECT DETECTION AND SEGMENTATION ALGORITHMS

In this chapter, we various object detection and segmentation methods are discussed and compared which utilize convolutional neural networks. In particular, we present the detection techniques combining CNN with the classification of regional proposals. We further discuss how the regions of interest (ROI), also called region proposals, are generated. Further, we discuss the encoder-decoder approach for segmentation.

3.1 Object detection algorithms

Object detection refers to the process of identification of the objects in an image or video. Many algorithms have been proposed in recent years such as R-CNN, Fast-RCNN, Faster-RCNN, and YOLO. YOLO algorithm struggles with small objects within the image, so we discuss the other three algorithms in this section.

3.1.1 R-CNN

R-CNN first obtains the region proposals from the input image using selective search. From each region proposal, Convolutional network (Convnet) extracts the features. After this step, features are fed to the support vector machines (SVMs) to detect objects. Unfortunately, due to the presence of many Convnets and SVMs, R-CNN becomes slow.

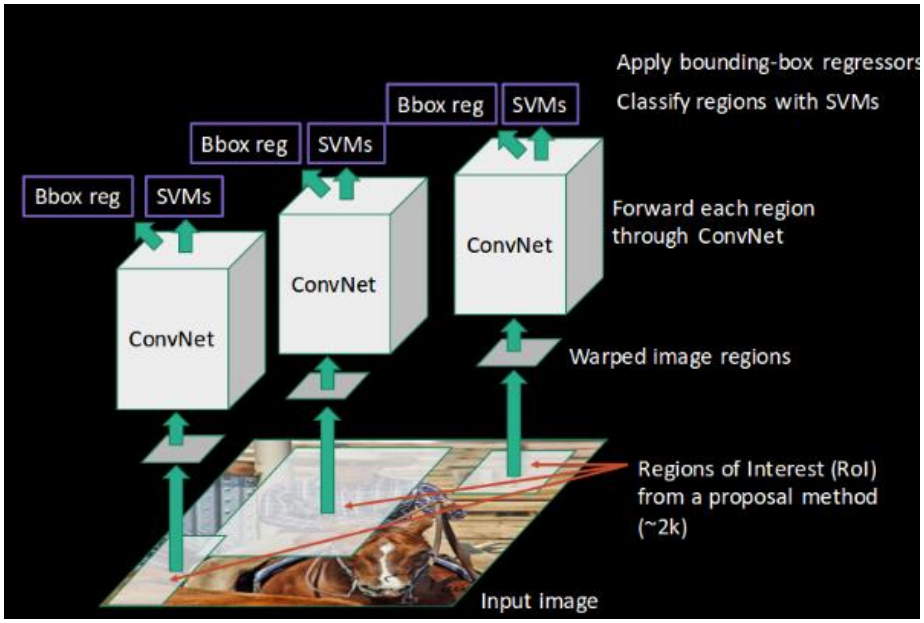

Figure 22: *R-CNN*
Image Source: [38]

3.1.2 Fast R-CNN

In Fast R-CNN, the first step is to pass the entire image to Convnet instead of extracting regions from the image and then passing each region to different Convnet. In the next step, the regions of interest (ROIs) are selected using the features that we have extracted from the entire image. Therefore, Fast R-CNN uses a single model to extract the feature and detecting the object instead of using different models for different regions.

Fast R-CNN is less complex and faster than R-CNN. However, Fast R-CNN becomes slow due to the use of selective search for extracting the regions when applied on a large dataset.

3.1.3 Faster R-CNN

Faster R-CNN has a similar structure as that of Fast R-CNN. In Faster R-CNN, Region Proposal Network (RPN) is used instead of selective search for extracting the ROIs.

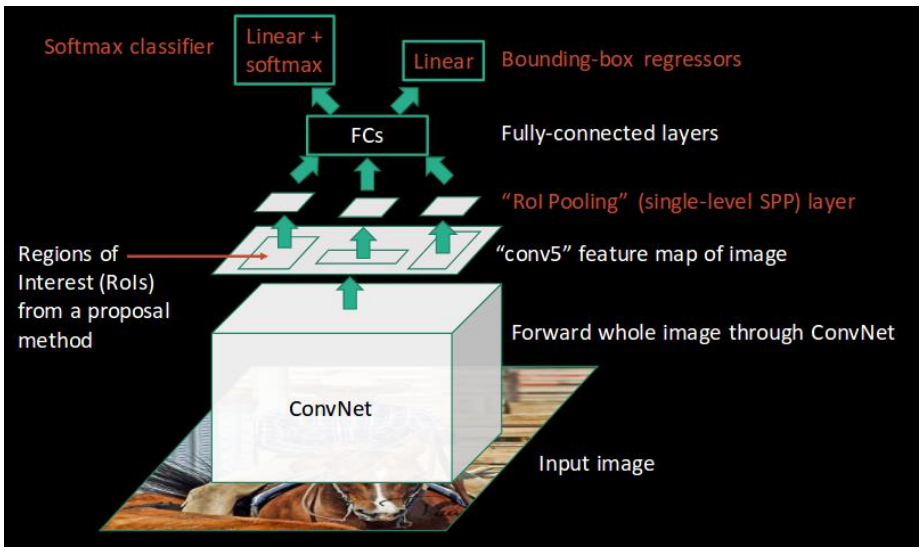

Figure 23: *Fast R-CNN*
Image Source: [38]

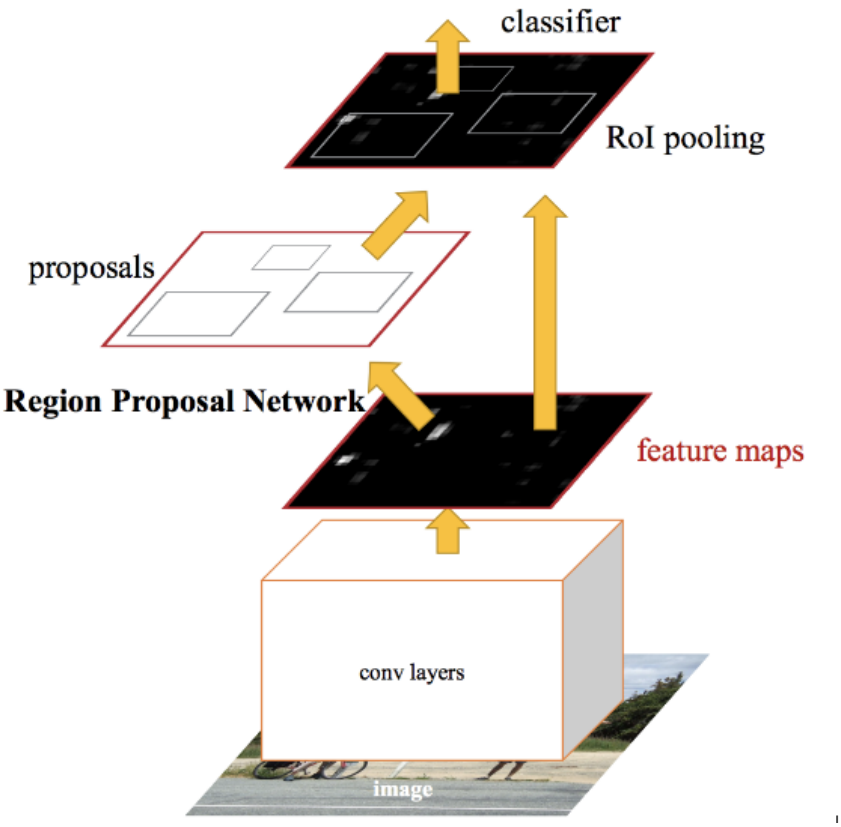

Figure 24: Faster R-CNN
Image Source: [38]

Faster R-CNN algorithm follows the following steps to detect objects in an image:

1. Pass the entire input image through the Convnet to extract the feature map.
2. Feed the feature map extracted using Convnet to the Region Proposal Network (RPN) to obtain object proposals.
3. ROI pooling layer is applied to object proposals to resize all the proposals to the same size.
4. In the last step, the bounding boxes for the image are predicted by feeding the proposals in step 3 to a fully connected layer.

Algorithm name	**Description**	**Time per image**	**Drawbacks**
CNN [39]	Split up the frame into various regions. Each region is classified into different classes.	–	High computation time due to the number of regions is high.
R-CNN [40]	Generate ROIs using selective search and feed each region to different Convnet.	40-50 secs	Generation of ROIs before feature selection makes the model complex. Resulting in high computational complexity. Computation time is large.
Fast R-CNN [41]	First, the entire image is passed through Convnet for feature extraction. Then, selective search is used to extract the regions.	2 secs	For large data selective search becomes slow.

Faster R-CNN [42]	In place of using selective search it uses Region Proposal Network. Hence, the time taken by algorithm reduces significantly.	0.2 secs	Generation of object proposals takes time as systems are working one after another in the series.

Table 2: Comparing the different detection algorithms.

3.2 Segmentation

Segmentation refers to the process of classification of each and every pixel in a frame into respective class. Segmentation makes image analysis and image understanding easier. It has many applications in artificial intelligence such as autonomous driving, remote sensing, object detection, industrial inspection, medical imaging analysis, etc. The advancement of deep learning techniques makes the task of image segmentation precise and much faster.

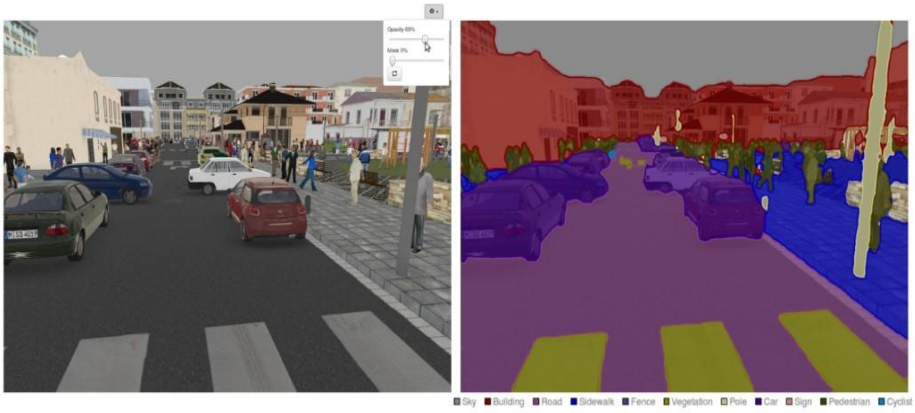

Figure 25: An example of Segmentation.
Image Source: [43]

Fig. 25 is an example of the classification of each pixel into different classes. The input image is at left and different color shows different classes in the output image (right).

Segmentation is a high-level task which helps to attain the task of scene understanding. Many approaches have been developed for

achieving the task of segmentation using deep learning methods in recent years. One of the most popular approaches is Encoder-Decoder based.

3.2.1 Encoder-Decoder based approach

Encoder-Decoder based approach is one of the most popular approaches to achieve the task of segmentation. In such approaches, the first downsampling of the pixels of the input image is performed to develop lower resolution features. Then, these features are used for learning making the process efficient and fast. In the second phase of the structure, the upsampling of the features into a segmentation map of input image size is performed.

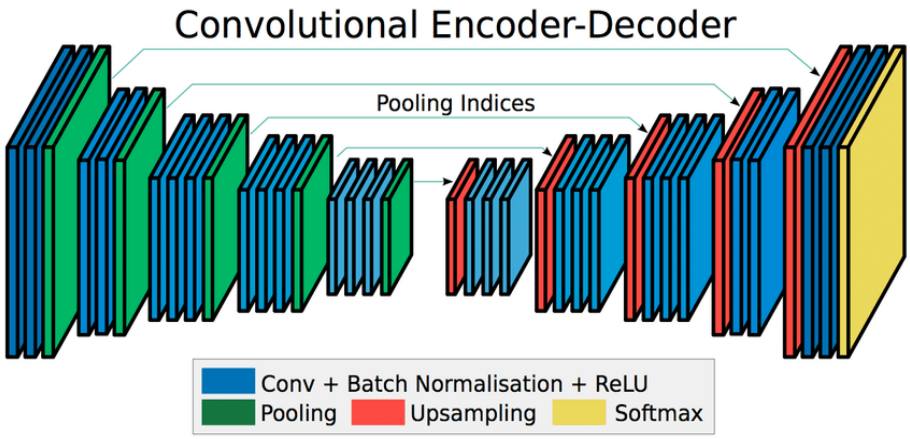

Figure 26: Encoder-Decoder Architecture
Image Source: [43]

There are many methods which are used for upsampling of the feature map such as:

3.2.2 Nearest neighbor

In this method of upsampling, output pixels are mapped according to their nearest neighboring pixels present from input pixel values. An example of the nearest neighbor is shown in Figure 27.

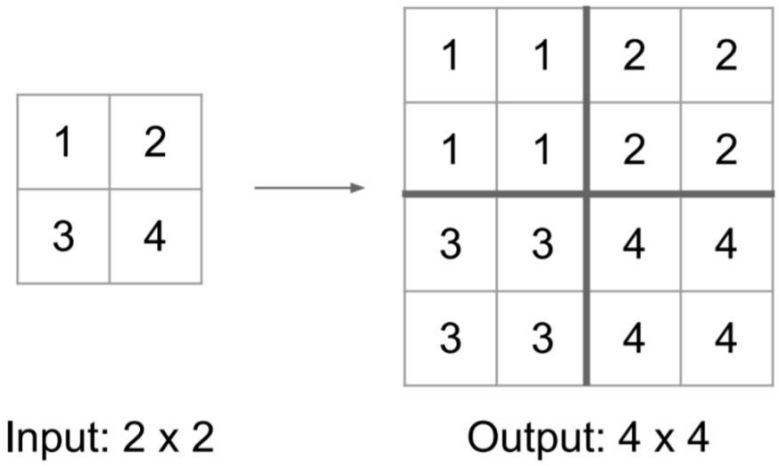

Figure 27: Nearest Neighbor

3.2.3 Bed of nails

In this type of upsampling, only input pixels are retained in the output and value of interpolated pixels is set to 0, an example is shown below.

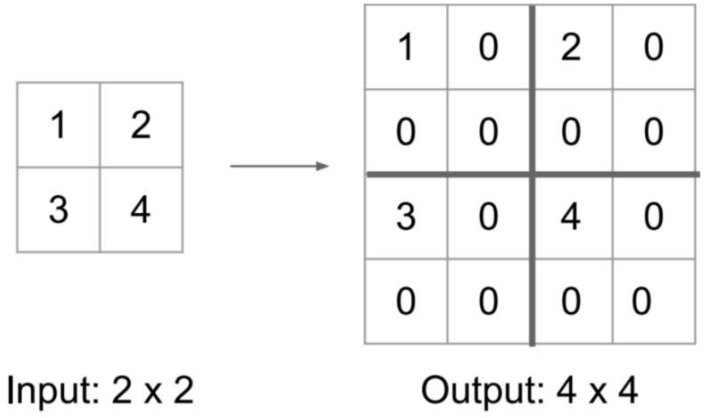

Figure 28: Bed of Nails

3.2.4 Transpose convolution

Transpose convolutions are the most famous upsampling method. Transpose Convolution is also referred to as the opposite of convolution or deconvolution. Deconvolution or a transposed convolution is simply a normal convolution operation with special padding.

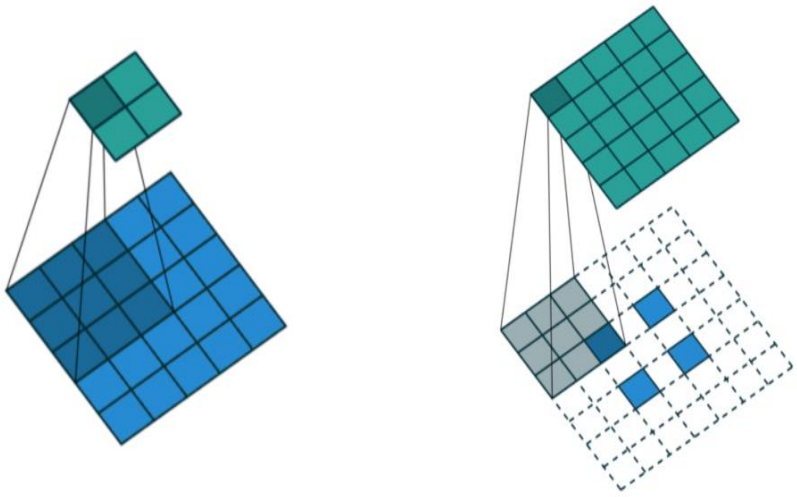

Figure 29 : Normal convolution (left) and transposed convolution (right)
Image Source: [44]

In the Figure 29 (left), 5×5 layer (blue) is the input of normal convolution with filter size 3 and stride 2 and output is a 2×2 layer (green). With some fancy padding in the transposed convolution, we can convert the 2×2 layer back to 5×5 layer.

Pooling operations are used to downsample the resolution by converging a local area with a single value.

Chapter 4

METHODOLOGY

In this section, details of our approach are provided. In section 4.1, we present the overview of the pipeline used in the thesis. We also discuss the optical flow RGB image calculation. Then in section 4.2, we discuss the detailed Faster-RCNN architecture that we have used. Additionally, some implementation details are presented in this chapter.

4.1 Overview of the pipeline

Figure 30 shows the block diagram explaining the pipeline of our approach. The feature which we have used to solve our problem is optical flow. The optical flow is obtained using the current and previous frame using FlowNet [45]. FlowNet is a CNN based network which estimates the optical flow RGB image as a supervised learning task. FlowNet takes two consecutive frames of the video as input and then returns the optical flow RGB image as output. The RGB image obtained from the FlowNet is fed as input to the CNN architecture based on Faster R-CNN. The detailed architecture of Faster-RCNN is explained in section 4.2 of this chapter along with some implementation details.

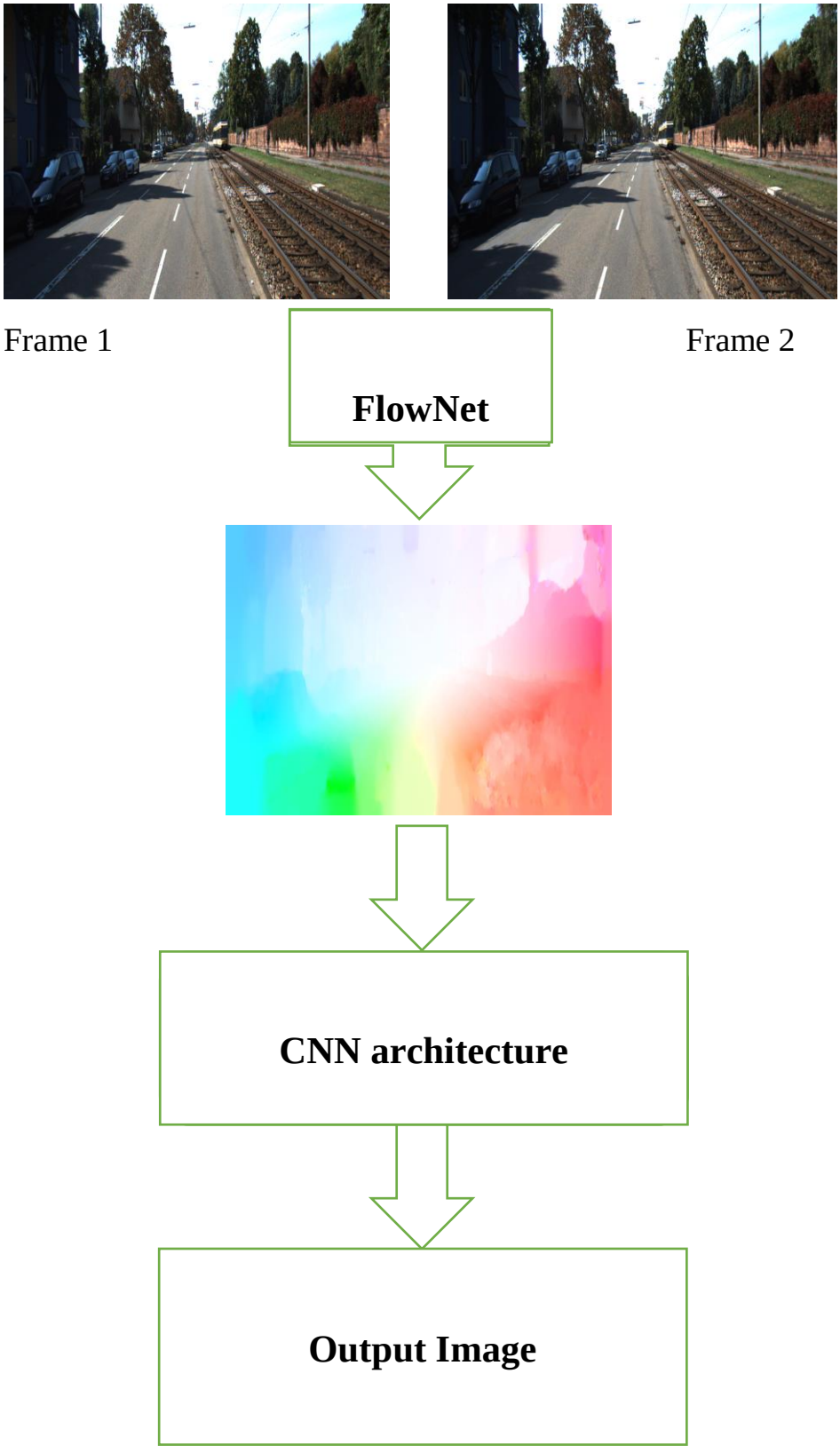

Figure 30: Flow chart of the pipeline structure

4.2 Object detection

Faster R-CNN is used as the main method for object detection experiments. Faster R-CNN is now entrenched and has implementations and pre-trained architectures accessible for many platforms. Faster R-CNN provides increased accuracy and speed. Faster R-CNN network can be divided into two main parts, namely region proposal network (RPN) and a network using these proposals to detect objects. The time cost is reduced significantly by the use of RPN for region proposals generation as we have seen in section 3.1.3.

For feature extraction, we have used imagenet [46] pre-trained VGG-16 [47] network because it has a simple structure and good performance. The network model of the pre-trained network is kept the same, which is shown in fig 31. The fully connected layer of the VGG-16 network is replaced by bottleneck convolution (conv 1x1) to reduce the computational cost of the network. Then, these features are fed to Region Proposal Network (RPN).

4.2.1 Region proposal network (RPN)

RPN is the backbone of Faster-RCNN and has demonstrated to be exceptionally effective until now. Its motivation is to propose multiple objects that are recognizable inside a specific frame. This method was proposed by S. Ren, K. He, R. Girshick and J. Sun [42].

RPN is made up of two parts classifier and a regressor. Classifier decides the likelihood of a proposal having the object. Regression regresses the coordinates of the proposals. For any image, scale and aspect-ratio are two important parameters.

The total number of proposals generated at a pixel is defined by the anchors. Anchors are boxes. Anchors have two parameters size and height to width ratio. The anchors used by us are of size 64,128, 256 and 512 and their height to width ratios are 1:1, 1:2 and 2:1 respectively. So, a total of 12 proposals is generated for each pixel. The

total number of anchors generated is called K, for our case K=12. Total anchors generated for an image is the width of image*length of image*K.

Therefore, the primary work of Region proposal is to discover all feasible places where target object can be located in the input image. The output of RPN is a list of bounding boxes of most probable object locations. These bounding boxes are often referred as region proposals or ROIs. These proposals act as input to ROI pooling and classification network.

4.2.2 ROI pooling and classification network

For every region proposal from the previous stage, the final output is to get classified as one of the target classes or the background.

If ROIs are not generated during the RPN stage, then, we can not classify the target in this phase. Therefore, a high recall is recommended for the region proposals and it can be attained by generating a very large number of ROIs (usually, a few thousand per frame). Most of these ROIs belong to the background. The task of reducing the ROI is performed by ROI pooling.

There are some problems with generating a very large number of ROIs. The large number of ROI can prompt execution issues which lead to performance problems. This would make locating the position of the target object difficult. It is below optimum in terms of processing speed. Moreover, end-to-end training cannot be performed, i.e., all the components of the system cannot be trained in one run (which would produce much better outcomes). ROI pooling plays an important role in overcoming the mentioned problem.

Region of interest pooling: It is a neural network layer which is very useful for object detection problem. It was first proposed by Ross Girshick in April 2015 and it accomplishes enormous speed of both training and testing. Also, detection accuracies are maintained along with speed. ROI pooling layer has two inputs:

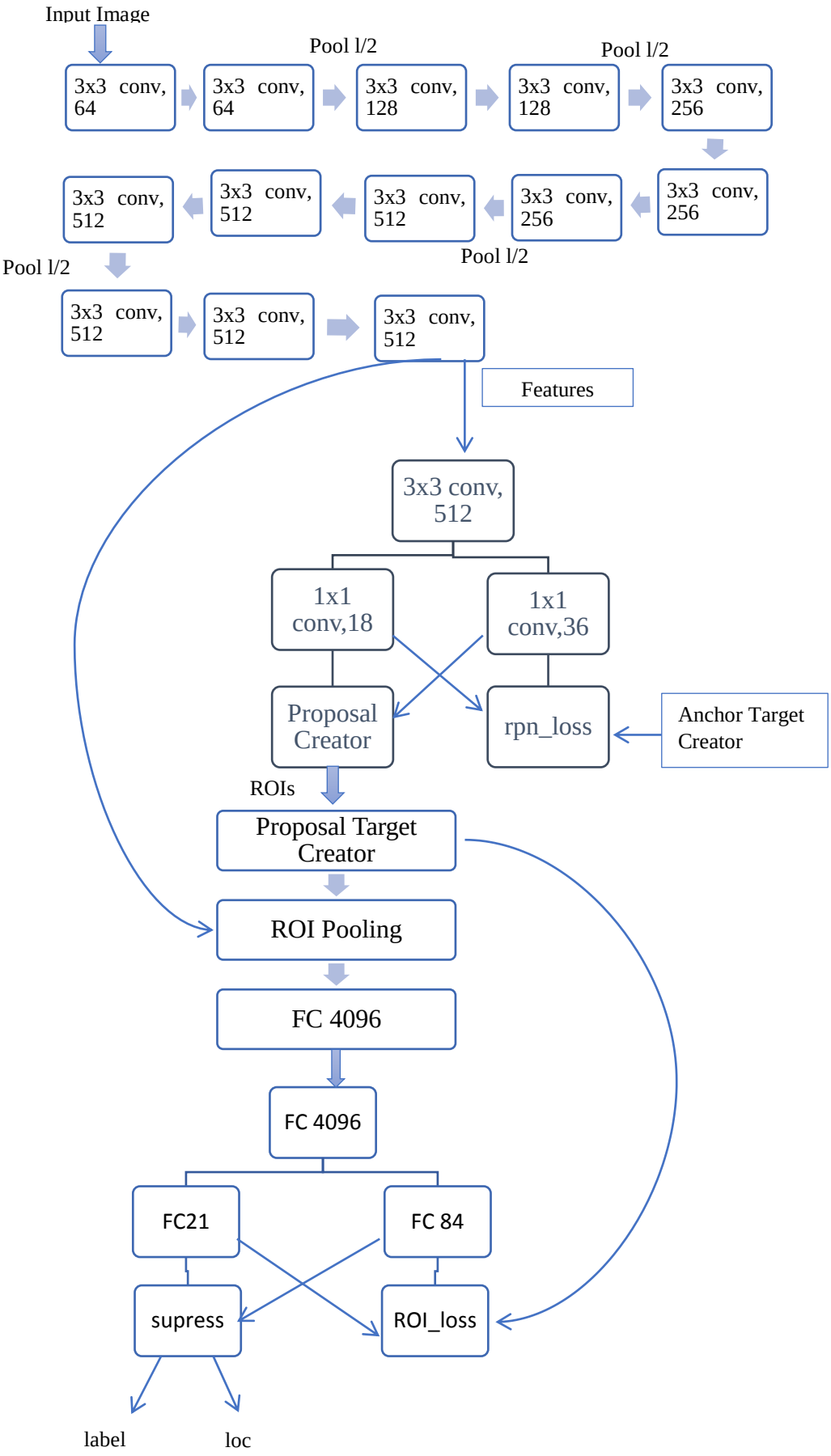

Figure 31: Detailed Faster-RCNN architecture.

1. The feature map obtained from VGG-16 network with various convolution and max-pooling layers.
2. A $N \times 5$ matrix which is indicating a list of ROIs, where N represents the number of region proposals. The first column of the matrix represents the image number and the remaining four columns are the position of the top left and bottom right corners in the X-Y plane of the region.

For every ROI from the input list, the region of interest pooling takes the corresponding section of the input feature matrix and resizes it to pre-defined dimension. The scaling is carried out by:

1. Splitting the ROIs into fixed size segments.
2. Find the highest value in each feature segment.
3. Copying the highest values obtained in step 2 to the output buffer.

The outcome is that from a list of rectangles (region proposals) with different sizes, the list of corresponding feature maps is obtained with a fixed dimension. The ROI pooling output dimension is determined exclusively by the number of segments in which proposals are splitted. ROI pooling boosts the processing speed and for multiple object proposals on the frame, the same input feature map can be used. Since the computation of the convolution operation at the primary stages of the process is very time costly, a lot of time can be saved by using this approach.

Chapter 5

DATASET, RESULTS, AND DISCUSSION

In this chapter, we present the dataset details, result, and discussion. In section 4.1, we discuss the datasets mainly KITTI BDD datasets. Then in section 4.2, we present some implementation details along with the results that we have obtained. Further in section 4.3, there is a brief discussion about the results.

4.1 Dataset

There are many datasets present for the problem of object detection in dashcam videos, but for moving object detection, annotations are not present. Most of the datasets present for moving object detection are synthetic, relatively small or has limited camera motion. Due to this, we have modified the raw datasets according to requirement. These datasets are KITTI and BDD. Our proposed framework is trained and tested on these datasets.

	KITTI	Cityscapes	ApolloScape	Mapillary	BDD100K
# Sequences	22	~50	4	N/A	100,000
# Images	14,999	5000 (+2000)	143,906	25,000	120,000,000
Multiple Cities	No	Yes	No	Yes	Yes
Multiple Weathers	No	No	No	Yes	Yes
Multiple Times of Day	No	No	No	Yes	Yes
Multiple Scene types	Yes	No	No	Yes	Yes

Table 3: Comparison of raw datasets

4.1.1 KITTI

In this dataset, odometry information is used. The odometry information gives information about the velocity of the camera by using GPS values. Then, the velocities of vehicles are computed. The velocity vector per vehicle is compared to the odometry ground-truth and vehicles are classified into static/moving classes. Vehicles such as the car, truck and van are mainly focused on this dataset.

Total 1750 frames are present in the dataset. In addition, 200 frames are included from KITTI scene flow, which increases our total frames count to 1950. In the 1950 frames, 2383 vehicles are moving, which is a very small number for the training of deep networks.

Figure 32: Sample images from KITTI Dataset

4.1.2 BDD-mod

Since, the datasets which are present for moving object detection are synthetic, relatively small or has limited camera motion. We tried to create our own dataset using the raw videos from BDD dataset because BDD dataset is most diverse as shown in the Table 3. We call BDD-mod to this new dataset.

From the 100k videos present in BDD dataset, we have chosen 40 videos with multiple cities, multiple kinds of weather and multiple times of the day. Each video is of 40 sec at 30 fps. This gives us 1200 frames per video. From 40 videos we get total 48000 frames.

Figure 33: Sample images from BDD Dataset

All the videos are annotated manually by using an open source annotation tool called CVAT (Computer vision annotation tool). CVAT is licensed under the MIT License. It has many powerful features such as interpolation of bounding boxes between keyframes, automatic annotation using tensor-flow api, shortcuts for critical operations, etc.

4.2 Results

The network architecture is the same as discussed in section 4.2. Other parameters are as follows:

1. Image size 600 is used.
2. Ratio of anchors is 1:1, 1:2, 2:1 and anchor size is 64, 128, 256, 512.
3. Max number of non-max-suppression is 300.
4. Number of ROI which are processed simultaneously is 4.
5. Adam optimizer with the learning rate of $1e^{-5}$ is used.
6. The network is trained for 100 epoch.

Figure 34 shows the results that we have obtained using the proposed methodology. The Figures in the left side show the input image, middle images are the optical flow obtained by using Flownet. Images present at the right side show the detected moving object.

(a) (b) (c)

Figure 34: (a) The input image frame. (b) The optical flow RGB images obtained using Flownet. (c) Output images.

To measure the performance of the proposed method, we used the mean average precision (mAP) as the performance metric. A box is positive if the overlap score is more than 0.5. The mAP comes out to be 48% for KITTI dataset and 42% for BDD-mod dataset.

5.3 Discussion

Moving object detection from the dashcam videos is a formidably challenging problem due to the complex background, illumination variation, abrupt motion, presence of shadows, etc. as discussed in section 1.3. Also, the camera moves along with the motion of the target object poses a significant challenge to this type of problem. We have explored the performance of object detection based CNN for moving object detection in a dashcam. For performance evaluation, two datasets are used KITTI and BDD-mod. The performance metric used for evaluation is mAP and it comes out to be 48% for KITTI dataset and 42% for BDD-mod dataset. From the results, we can say that the problem is far from being solved and the main reason behind this is the use of only optical flow features.

The proposed algorithm fails to detect moving objects:

1. When target objects are small.
2. When the speed of the target object is very slow.
3. When the target object is occluded.
4. Illumination variation.

Chapter 6

Conclusion and Future Work

5.1 Conclusion

This work is an attempt towards the solution of moving object detection in moving cameras problem. The proposed method is based on the detection network in deep learning in which we are using the RGB optical flow images as the input.

The performance of the proposed approach is evaluated on KITTI dataset and BDD-mod and mAP comes out to be 48% and 42% respectively.

The proposed method fails to detect objects that are moving very slow. As the optical flow variation are very less for such type of objects. Also, the method fails to detect distant objects.

5.2 Future work

Our method is solely based on the optical flow images that we have extracted using the Flownet, therefore the performance is very low. Hence, a joint model which uses the original image, the optical flow image and other features can be explored. It will be very helpful in detecting the objects in complex scenes.

Also, segmentation based CNN can also be applied to detect the moving objects but for that datasets must be prepared. Also, new datasets with varying weather and light conditions can be prepared for deployment of the problem to the real world.

REFERENCES

1. Cucchiara, R., et al. *Statistic and knowledge-based moving object detection in traffic scenes*. in *ITSC2000. 2000 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 00TH8493)*. 2000. IEEE.

2. Wu, S., O. Oreifej, and M. Shah. *Action recognition in videos acquired by a moving camera using motion decomposition of lagrangian particle trajectories*. in *2011 International conference on computer vision*. 2011. IEEE.

3. Hu, W., et al., *A survey on visual surveillance of object motion and behaviors.* 2004. **34**(3): p. 334-352.

4. Malamas, E.N., et al., *A survey on industrial vision systems, applications and tools.* 2003. **21**(2): p. 171-188.

5. Torr, P.H.J.P.T.o.t.R.S.o.L.S.A.M., Physical and E. Sciences, *Geometric motion segmentation and model selection.* 1998. **356**(1740): p. 1321-1340.

6. Papazoglou, A. and V. Ferrari. *Fast object segmentation in unconstrained video*. in *Proceedings of the IEEE International Conference on Computer Vision*. 2013.

7. Ochs, P., et al., *Segmentation of moving objects by long term video analysis.* 2013. **36**(6): p. 1187-1200.

8. Tokmakov, P., K. Alahari, and C. Schmid. *Learning motion patterns in videos*. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

9. Jain, S.D., B. Xiong, and K. Grauman. *Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos*. in *2017 IEEE conference on computer vision and pattern recognition (CVPR)*. 2017. IEEE.

10. Drayer, B. and T.J.a.p.a. Brox, *Object detection, tracking, and motion segmentation for object-level video segmentation.* 2016.

11. Gao, P., X. Sun, and W. Wang. *Moving object detection based on kirsch operator combined with optical flow*. in *2010 International Conference on Image Analysis and Signal Processing*. 2010. IEEE.

12. Radke, R.J., et al., *Image change detection algorithms: a systematic survey.* 2005. **14**(3): p. 294-307.

13. Stauffer, C. and W.E.L. Grimson. *Adaptive background mixture models for real-time tracking*. in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*. 1999. IEEE.

14. Royden, C.S. and K.D.J.V.r. Moore, *Use of speed cues in the detection of moving objects by moving observers.* 2012. **59**: p. 17-24.

15. Royden, C.S. and E.M.J.V.r. Connors, *The detection of moving objects by moving observers.* 2010. **50**(11): p. 1014-1024.

16. Royden, C.S. and M.A.J.V.r. Holloway, *Detecting moving objects in an optic flow field using direction-and speed-tuned operators.* 2014. **98**: p. 14-25.

17. Sheikh, Y., O. Javed, and T. Kanade. *Background subtraction for freely moving cameras*. in *2009 IEEE 12th International Conference on Computer Vision*. 2009. IEEE.

18. Yu, H., et al. *Moving object detection using an in-vehicle fish-eye camera*. in *2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*. 2010. IEEE.

19. Alsinet, T. *Motion Segmentation: a Review*. in *Artificial Intelligence Research and Development: Proceedings of the 11th International Conference of the Catalan Association for Artificial Intelligence*. 2008. IOS Press.

20. Chen, T., S.J.I.T.o.C. Lu, and S.f.V. Technology, *Object-level motion detection from moving cameras.* 2016. **27**(11): p. 2333-2343.

21. Xiao, J., M.J.I.t.o.p.a. Shah, and m. intelligence, *Motion layer extraction in the presence of occlusion using graph cuts.* 2005. **27**(10): p. 1644-1659.

22. Hayman, E. and J.-O. Eklundh. *Statistical background subtraction for a mobile observer*. in *null*. 2003. IEEE.

23. Choi, W., et al., *A general framework for tracking multiple people from a moving camera.* 2012. **35**(7): p. 1577-1591.

24. Alahi, A., V. Ramanathan, and L. Fei-Fei. *Socially-aware large-scale crowd forecasting*. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

25. Shi, X., et al. *Multi-target tracking with motion context in tensor power iteration*. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

26. Ess, A., et al. *A mobile vision system for robust multi-person tracking*. in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. 2008. IEEE.

27. *Berkeley Deep Drive (BDD) Dataset, https://bdd-data.berkeley.edu/.*

28. *RGB color model, https://en.wikipedia.org/wiki/.*

29. *HSL and HSV, https://en.wikipedia.org/wiki/HSL_and_HSV.*

30. *Optical Flow, https://en.wikipedia.org/wiki/Optical_flow.*

31. *Introduction to Motion Estimation with Optical Flow, https://blog.nanonets.com/optical-flow/.*

32. Fukushima, K.J.N.n., *Neocognitron: A hierarchical neural network capable of visual pattern recognition.* 1988. **1**(2): p. 119-130.

33. Hubel, D.H. and T.N.J.T.J.o.p. Wiesel, *Receptive fields and functional architecture of monkey striate cortex.* 1968. **195**(1): p. 215-243.

34. *CNN architecture, https://researchgate.net/.*

35. *Understanding of Convolutional Neural Network (CNN) — Deep Learning, https://medium.com/@RaghavPrabhu/.*

36. Springenberg, J.T., et al., *Striving for simplicity: The all convolutional net.* 2014.

37. *Understanding activation functions in neural networks*, https://medium.com/the-theory-of-everything/. .

38. *Object Detection in images*, https://www.saagie.com/.

39. Roska, T., et al., *The CNN universal machine: an analogic array computer*. 1993. **40**(3): p. 163-173.

40. Girshick, R., et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.

41. Girshick, R. *Fast r-cnn*. in *Proceedings of the IEEE international conference on computer vision*. 2015.

42. Ren, S., et al. *Faster r-cnn: Towards real-time object detection with region proposal networks*. in *Advances in neural information processing systems*. 2015.

43. *Understand semantic segmentation with the fully convolutional network*, https://medium.com/@pallawi.ds/.

44. *Up sampling with transposed convolution*, https://towardsdatascience.com/.

45. Dosovitskiy, A., et al. *Flownet: Learning optical flow with convolutional networks*. in *Proceedings of the IEEE international conference on computer vision*. 2015.

46. Simon, M., E. Rodner, and J.J.a.p.a. Denzler, *Imagenet pre-trained models with batch normalization.* 2016.

47. Simonyan, K. and A.J.a.p.a. Zisserman, *Very deep convolutional networks for large-scale image recognition.* 2014.