SEMANTIC SEGMENTATION

submitted in partial fulfilment of the requirements

for the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRICAL ENGINEERING

by

P VENKATA BHANU TEJA EE15B015

Supervisor(s)

Dr. Rama Krishna Sai Gorthi

भारतीय प्रौद्योगिकी संस्थान तिरुपति

TIRUPATI

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY TIRUPATI

MAY 2019

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Place: Tirupati
Date: 19-05-2019

Signature
P Venkata Bhanu Teja
EE15B015

BONA FIDE CERTIFICATE

This is to certify that the thesis titled **Semantic Segmentation** , submitted by **P Venkata Bhanu Teja**, to the Indian Institute of Technology, Tirupati, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Tirupati
Date: 19-05-2019

Dr. Rama Krishna Gorthi
Guide
Assistant Professor
Department of Electrical Engineering
IIT Tirupati - 517501

Note: If supervised by more than one professor, professor's name must be included and all supervisors' signature is must.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Dr.Rama Krishna Sai Gorthi for providing their invaluable guidance, comments and suggestions throughout the course of the project.

I would thank Computer-Vision laboratory of IIT Tirupati for providing the very high end servers with very good GPU's for training various type of networks much faster.

Thanks to all those who helped directly or indirectly me during thesis and research work.

ABSTRACT

KEYWORDS: Semantic segmentation; Encoders-Decoders; SEGNET;RefineNet; PSP-net.

Semantic segmentation is about labelling each single pixel in an image with the category it belongs to. There are several applications in a wide range of areas, like robotics, autonomous driving, mapping or medical image analysis, in which pixel-level labels are of primary importance. In recent years, deep neural networks have shown impressive results and have become state-of-the-art for several recognition tasks. In this thesis, we investigate into the use of deep neural networks for the task of semantic image segmentation. We adjust state-of-the-art fully convolutional networks, which are designed to label general scenes, to the task of image segmentation.The adjusting is done in a manner that where ever the image is not segmented properly a bounding box is drawn around that and then that cropped part of image is re-segmented by applying class-imbalance. In addition, we transfer the learned feature representation from a large-scale image database of everyday objects for classification to pixel-wise labelling of images by ignoring classes and consider only required classes/objects.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

CNN	Convolutional Neural Networks
CRF	Conditional random field
FCN	Fully connected networks
FN	False Negative
FP	False Positive
G.T	Ground Truth
ISPRS	International Society for Photogrammetry and Remote Sensing
IoU	Intersection over union
PSPNET	Pyramid Scene Parsing Network
R-CNN	Regional - Convolutional Neural Networks
ResNet	Convolutional Neural Networks
SVM	Support Vector Machines
TN	True Negative
TP	True Positive
YOLO	You Only Look Once

CHAPTER 1

INTRODUCTION

1.1 Preamble

Nowadays, in the field of computer vision **Semantic Segmentation** is one of the key problems. Looking at the big picture, semantic segmentation is one of the high-level task that helps us in complete scene understanding. The importance of scene understanding as a core computer vision problem is highlighted by the fact that an increasing number of applications nourish from inferring knowledge from imagery. Some of those applications include self-driving virtual reality, self-driving vehicles,crop health, human computer interaction etc. With the popularity of deep learning in recent years, many semantic segmentation problems are being tackled using deep architectures, most often Convolutional Neural Nets, which surpass other approaches by a large margin in terms of accuracy and efficiency.

1.2 Brief outline of Semantic segmentation

Semantic Segmentation is process of assigning a label to each and every pixel of the image unlike classification problems where you give a single label to whole image. Semantic segmentation treats the multiple objects of the same class as same entity. If you want to treat multiple objects of the same class as different entity then it is **Instance Segmentation** which is a whole new problem.

Semantic segmentation is a natural step in the progression from coarse to fine inference

- **Classification**: The input image could be located and consider it as classification problem, where the predictions are to be done on each and every pixel.
- **Localization / detection**: This is needed beacuse we not only require to which class they belong we also need the spatial information of those particular classes as well.

- **Semantic segmentation**: Finally, semantic segmentation achieves fine-grained inference by making dense predictions inferring labels for every pixel, so that each pixel is labeled with the class of its enclosing object ore region.

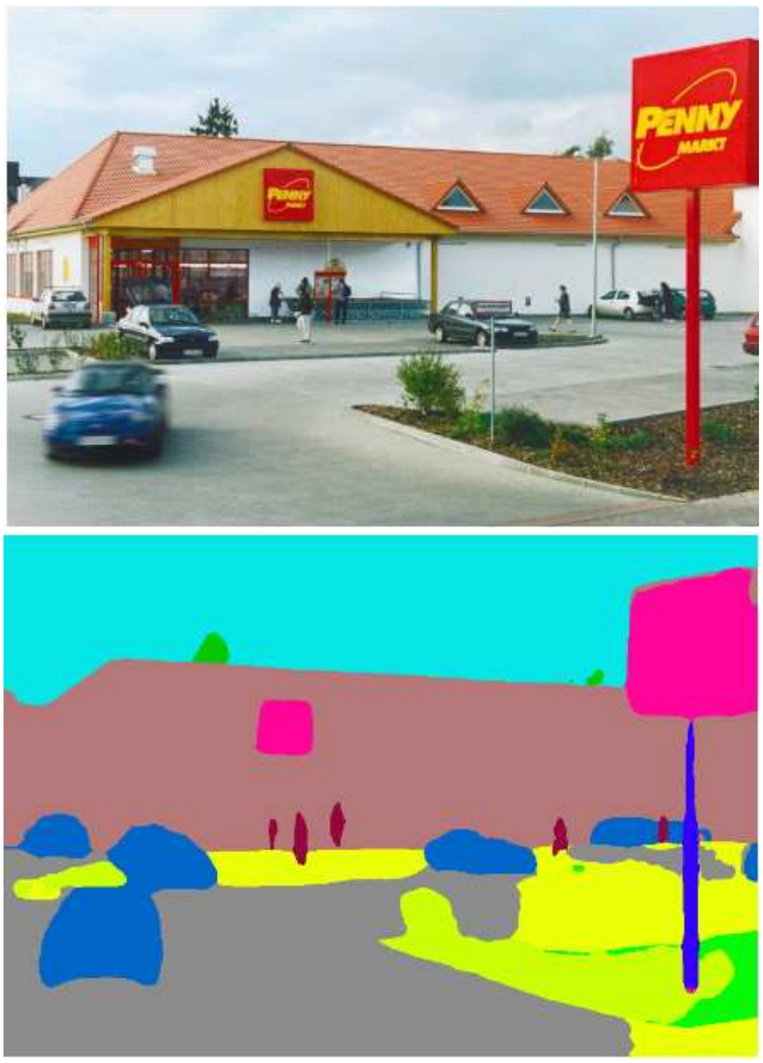


Figure 1.1: Input Image(At the top)
Segmented Image(At the bottom)

As shown in Fig 1.1 when the top image is passed through an architecture of semantic segmentation (The exact architecture and how it is done in the project will be discussed in the coming section. These example images are just to give the intuitive fell how semantic segmentation works.) we get output segmented as shown in the figure below it. From the segmented image we can clearly see segmented Cars, Buildings, Houses, Poles, Humans etc.

1.3 Segmentation Approaches and existing methods

Broadly semantic segmentation architecture can be considered as **Encoder** network followed by the **Decoder** network.

- The **Encoder** is a pre-trained classifier network like ResNet/VGG which increases the deepness of networks followed by a Decoder.
- The duty of **Decoder** network is to semantically project the discriminative features (lower resolution) learnt by the encoder onto the pixel space (higher resolution) to get a dense classification.

Unlike classification where the only important thing is end result of very deep networks.semantic segmentation not only requires discrimination at pixel level but also a mechanism to project the discriminative features learnt at different stages of the encoder onto the pixel space. Different methods have different types of mechanisms as part of decoding, let us see two most commonly and effectively used mechanisms.

1.3.1 Region based semantic segmentation

The region-based methods works on the principle of “segmentation using recognition” mechanism, which from an image first extracts regions and describes them, followed by that extracted region classification. At test time, the region-based predictions are transformed to pixel predictions, usually by labeling a pixel according to the highest scoring region that contains it.

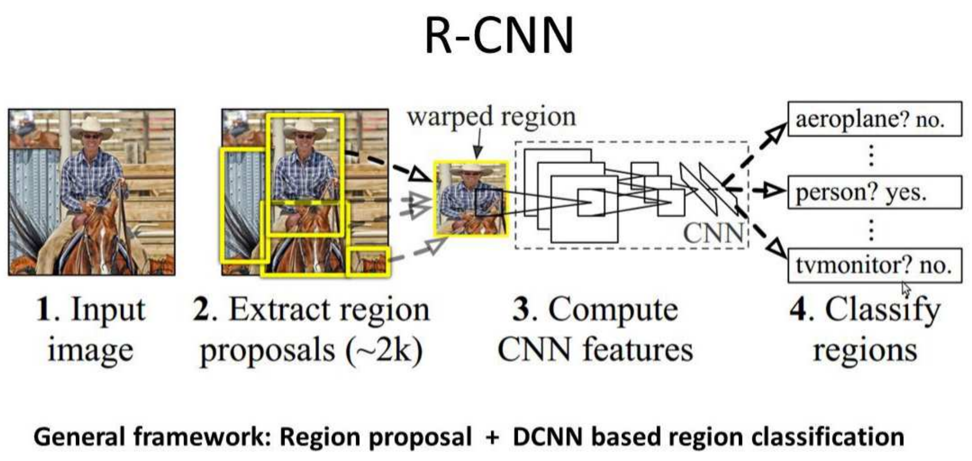


Figure 1.2: R-CNN architecture Girshick *et al.* (2013)

R-CNN (Regions with CNN feature) is one representative work for the region-based methods. It performs the semantic segmentation based on the object detection outputs. To be specific, R-CNN first utilizes selective search to extract a large quantity of object proposals and then computes CNN features for each of them. Finally, it classifies each region using the class-specific linear SVMs. Compared with traditional CNN structures which are mainly intended for image classification, R-CNN can address more complicated tasks, such as object detection and image segmentation, and it even becomes one important basis for both fields. Moreover, R-CNN can be built on top of any CNN benchmark structures, such as AlexNet, VGG, GoogLeNet, and ResNet. For the image segmentation task, R-CNN Girshick *et al.* (2013) extracted 2 types of features for each region: full region feature and foreground feature, and found that it could lead to better performance when concatenating them together as the region feature. R-CNN achieved significant performance improvements due to using the highly discriminative CNN features. However, it also suffers from a couple of drawbacks for the segmentation task:

- The feature is not compatible with the segmentation task.
- The feature does not contain enough spatial information for precise boundary generation.
- Generating segment-based proposals takes time and would greatly affect the final performance.

Due to these bottlenecks, recent research has been proposed to address the problems, including SDS, Hypercolumns, Mask R-CNN.

1.3.2 Fully Convolutional Network-Based Semantic Segmentation

FCN's learn pixel to pixel mappings without extracting the regions. FCNs networks are the extensions of CNN where the last layer becomes fully connected in FCN. The main idea is to make the classical CNN take as input arbitrary-sized images. The restriction of CNNs to accept and produce labels only for specific sized inputs comes from the fully-connected layers which are fixed. Contrary to them, FCNs only have convolutional and pooling layers which give them the ability to make predictions on arbitrary-sized inputs.

One issue in this specific FCN is that by propagating through several alternated convolutional and pooling layers, the resolution of the output feature maps is down sampled.

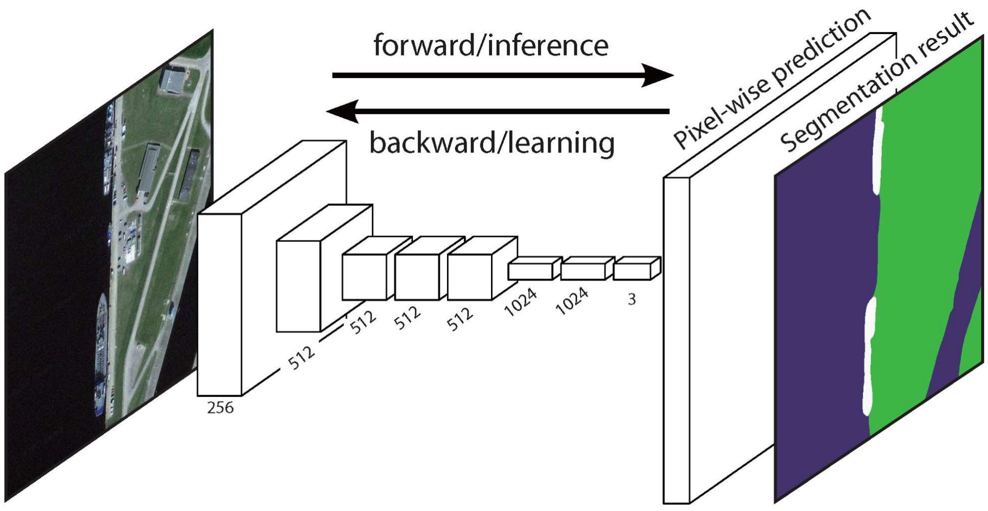


Figure 1.3: FCN Architecture Long *et al.* (2014)

Therefore, the direct predictions of FCN are typically in low resolution, resulting in relatively fuzzy object boundaries. A variety of more advanced FCN-based approaches have been proposed to address this issue, including SegNet, DeepLab-CRF, Dilated Convolutions, etc.

CHAPTER 2

Literature Review

In the course of the project various number of architectures and deep learning techniques have been applied on to various scenes(like aerial images, dense traffic images and parking lot images) to segment, these methods are discussed below in detail.

2.1 Residual Networks(ResNet) and its modifications

Regular DCNNs such as the AlexNetAlom *et al.* (2018) and VGG Simonyan and Zisserman (2014) aren't suitable for dense prediction tasks. First, these models contain many layers designed to reduce the spatial dimensions of the input features. As a consequence, these layers end up producing highly decimated feature vectors that lack sharp details. Second, fully-connected layers have fixed sizes and loose spatial information during computation(Which is not a desired situation for segmentation).

2.1.1 Theory Behind ResNet

Universal approximation theorem as stated in He *et al.* (2015) have proven that any feed-forward network with single layer is sufficient to represent any function. But layers might be very large and there are very high chances for over-fitting the data. Therefore to reduce the issue of over fitting we need to network deeper.

However increasing the network depth doesn't work simply stacking the layers together. Deep networks are hard to train because of the vanishing grad problem wikipedia (2019) and He *et al.* (2015)as the gradients are back propagated repetitive multiplication of gradients will make their value very small as a result as network goes deeper its performance gets saturated and its may even start rapidly degrading.

The core idea of ResNet is introducing a so-called "identity shortcut connection" that skips one or more layers, as shown in the following fig 2.2.

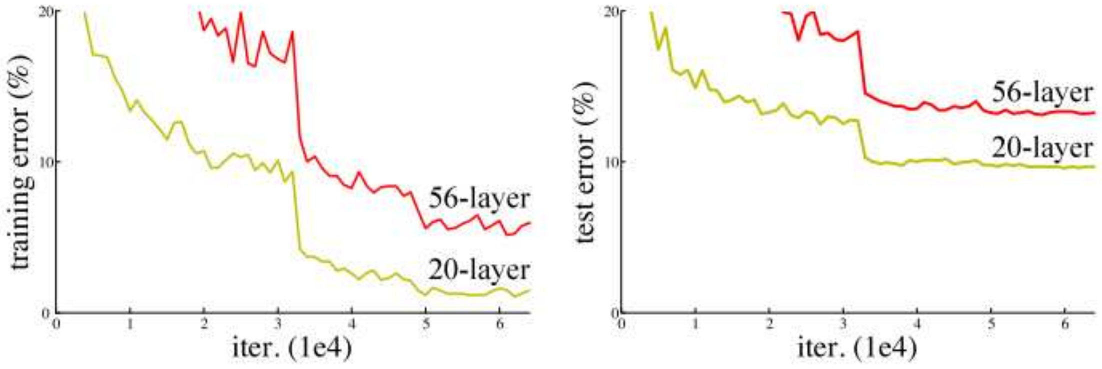


Figure 2.1: Increasing Depth Leads to Poor Performance He *et al.* (2015)

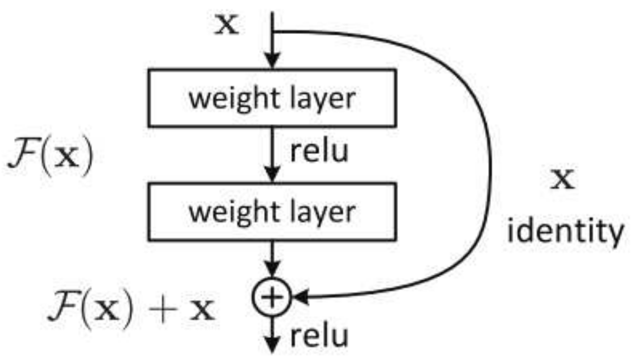


Figure 2.2: Residual Block He *et al.* (2015)

2.1.2 Identity mapping solving the issue of gradient descent

With the help this identity mappings stacking layers now shouldn't degrade the network as the issue of gradient vanishing is no-longer an issue here.However, experiments show that Highway Network performs no better than ResNet, which is kind of strange because the solution space of Highway Network contains ResNet, therefore it should perform at least as good as ResNet. This suggests that it is more important to keep these "gradient highways" clear than to go for larger solution space He *et al.* (2015).

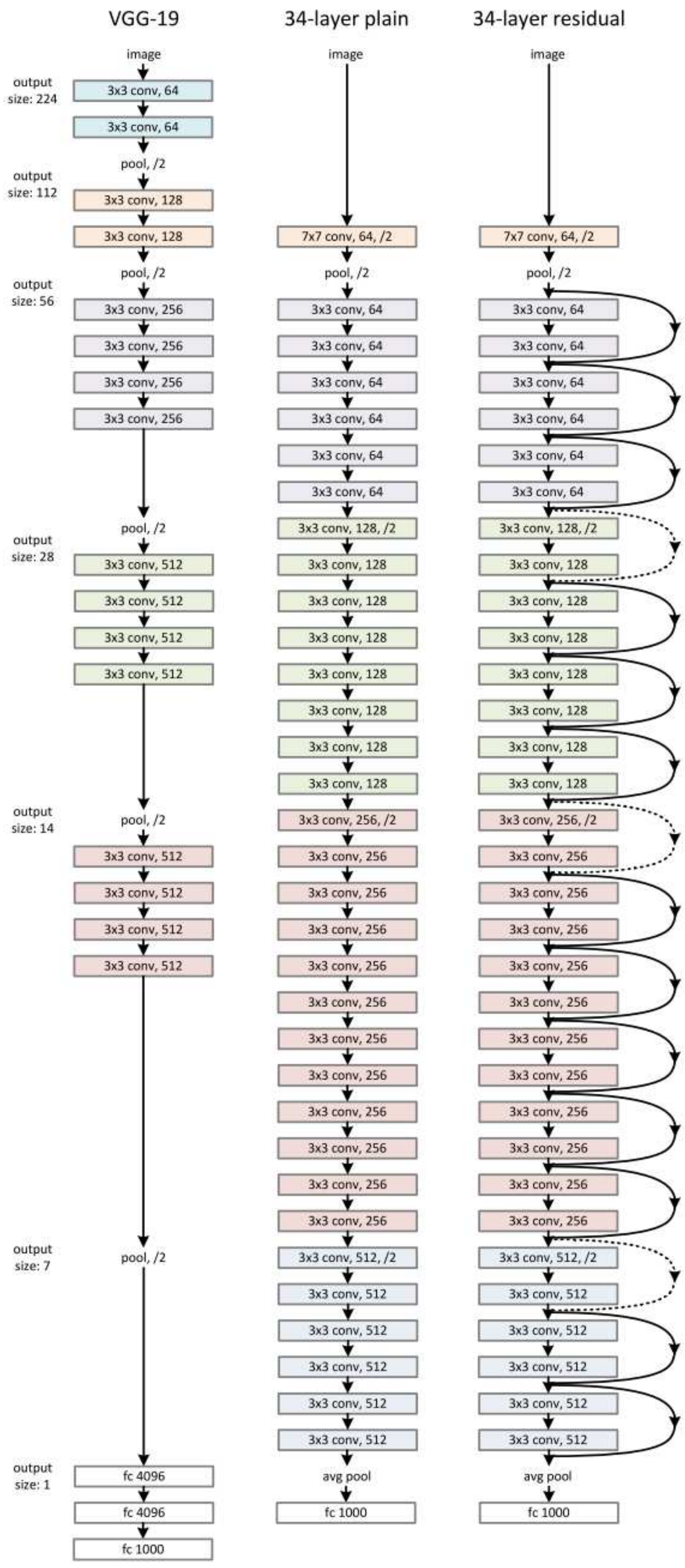


Figure 2.3: ResNet Architecture Huang *et al.* (2016)

2.1.3 Application of ResNet in segmentation

Having seen that the identity mapping of resnet over the issue of gradient descent.Now if we stack the large number of these such type resnet child. An n-layered resnet architecture is formed as shown in Fig 2.3 a deep convolutional neural network is formed. The important feature of this network is that the resolution of the output image of the architecture is not very small when compared to the of the input image. Which means the spatial information is not lost. This is the main theory behind semantic segmentation and also deep network is formed by stacking multiple layers. Hence usage of resnet showed some remarkable results in semantic segmentation training with the proper decoder.(This deep CNN helps in detecting the classes in the image and later when it passed through decoeder to upsample we get the spatial location by smooth class boundary divider).

2.2 SEGNET

SEGNET was the first state of art that outperformed every architecture till 2015 in the domain of semantic segmentation.

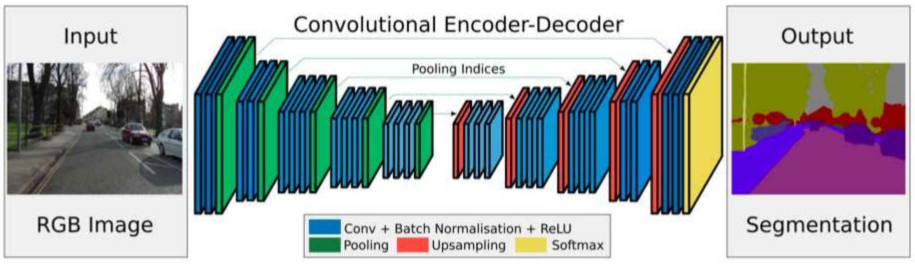


Figure 2.4: Segmentation of a camvid scene Badrinarayanan *et al.* (2015)

2.2.1 SegNet- Architecture

Encoder-Decoder pairs are used to create feature maps for classifications of different resolutions.

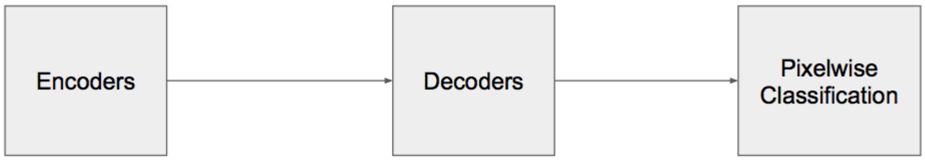


Figure 2.5: Brief SEGNET architecture Badrinarayanan *et al.* (2015)

2.2.2 Encoder

- 13 VGG16 Conv layers.
- Not fully connected, this reduces parameters from 134M to 14.7M.
- Good initial weights are available hence these layers are made non trainable.

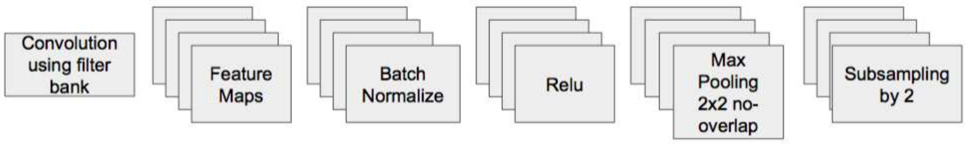


Figure 2.6: SEGNET Encoder architecture Badrinarayanan *et al.* (2015)

Each encoder as stated in Badrinarayanan *et al.* (2015) is like Fig 2.6. The novelty is in the subsampling stage, Max-pooling is used to achieve translation invariance over small spatial shifts in the image, combine that with Subsampling and it leads to each pixel governing a larger input image context (spatial window). These methods achieve better classification accuracy but reduce the feature map size, this leads to lossy image representation with blurred boundaries which is not ideal for segmentation purpose. It is desired that output image resolution is same as input image, to achieve this SegNet does Upsampling in its decoder, to do that it needs to store some information. It is necessary to capture and store boundary information in the encoder feature maps before sub-sampling. In order to to that space efficiently, SegNet stores only the max-pooling indices i.e. the locations of maximum feature value in each pooling window is memorised for each encoder map. Only 2 bits are needed for each window of 2x2, slight loss of precision, but tradeoff.

2.2.3 Advantages

- Boundaries can be clearly distinguished (Boundary delineation).
- Less number of parameters. (reduces parameters from 134M to 14.7M Badrinarayanan *et al.* (2015)) which reduces a lot of train and test time.

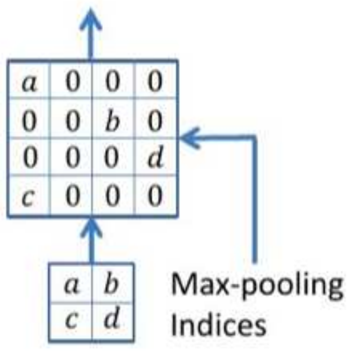


Figure 2.7: Upsampling in segnet Badrinarayanan *et al.* (2015)

2.2.4 Decoder

- For each of the 13 encoders there is a corresponding decoder which upsamples the feature map using memorised max-pooling indices.
- Sparse feature maps of higher resolutions produced.
- Sparse maps are fed through a trainable filter bank to produce dense feature maps.
- The last decoder is connected to a softmax classifier which classifies each pixel.

2.3 Refine-Net

A generic multi-path refinement network Lin *et al.* (2016)that explicitly exploits all the information available along the down-sampling process to enable high-resolution prediction using long-range residual connections. The deeper layers that capture high-level semantic features can be directly refined using fine-grained features from earlier convolutions. A chained residual pooling is also introduced which captures rich background context in an efficient manner.

2.3.1 Problems of ResNet and Dilated Convolution

- **(a) ResNet**: It suffers from downscaling of the feature maps which is not good for semantic segmentation.Lin *et al.* (2016)
- **(b) Dilated (Atrous) Convolution**: It is introduced in DeepLab and DilatedNet. Though it can help to keep the resolution of output feature maps larger, atrous filters are computationally expensive to train and quickly reach memory limits even on modern GPUs.

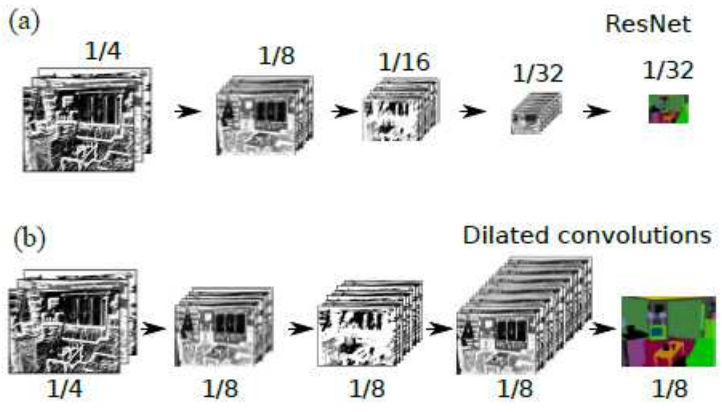


Figure 2.8: (a) ResNet (b) Dilated (Atrous) Convolution. Lin *et al.* (2016)

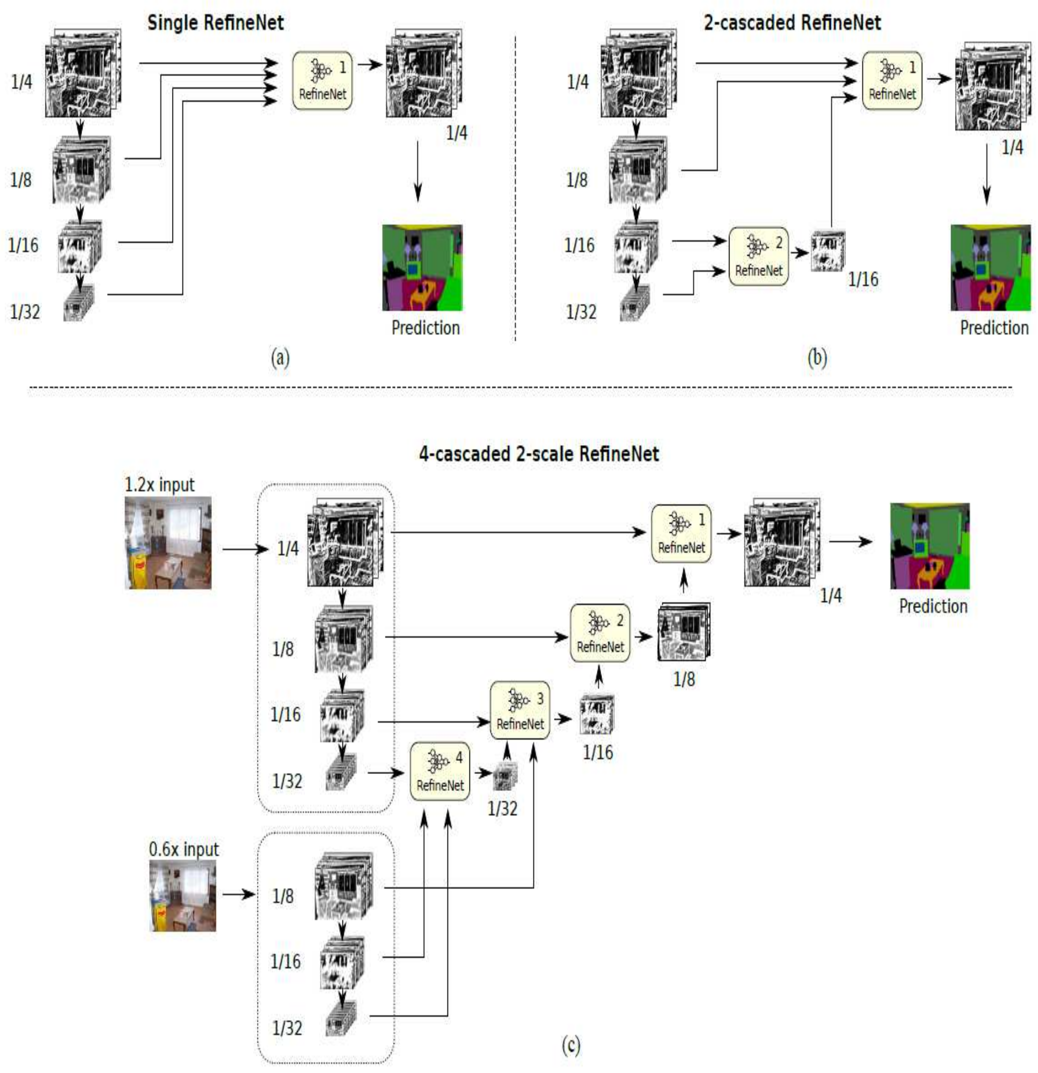


Figure 2.9: (a) Overall Architecture, (b) RCU, (c) Fusion, (d) Chained Residual Pooling. Lin *et al.* (2016)

2.3.2 RefineNet

- (**a**): At the top left of the figure, it is the ResNet backbone. Along the ResNet, different resolutions of feature maps go through Residual Conv Unit (RCU). Pre-Activation ResNet is used.
- **(b) RCU**: Residual block is used but with batch normalization removed.
- **(c) Fusion**: Then multi-resolution fusion is used to merge the feature maps using element-wise summation.
- **(d) Chained Residual Pooling**: The output feature maps of all pooling blocks are fused together with the input feature map through summation of residual connections. It aims to capture background context from a large image region. Lin *et al.* (2016)
- **(a) Output Conv**: At the right of the figure, finally, another RCU is placed here, to employ non-linearity operations on the multi-path fused feature maps to generate features for further processing or for final prediction.

2.4 PSPNet(Pyramid Scene Parsing Network)

By using Pyramid Pooling Module Zhao *et al.* (2016), with different-region-based context aggregated, PSPNet surpasses state-of-the-art approaches such as FCN, DeepLab, and DilatedNet. And PSPNet finally:

- Got the champion of **ImageNet Scene Parsing Challenge 2016**.
- Arrived 1st place on **PASCAL VOC 2012** and **Cityscapes datasets** at that moment.

2.4.1 The Need of Global Information

- **Mismatched Relationship**: FCN predicts the boat in the yellow box as a "car" based on its appearance. But the common knowledge is that a car is seldom over a river.
- **Confusion Categories**: FCN predicts the object in the box as part of skyscraper and part of building. These results should be excluded so that the whole object is either skyscraper or building, but not both. Tsang (2018)
- **Inconspicuous Classes**: The pillow has similar appearance with the sheet. Overlooking the global scene category may fail to parse the pillow.

So we need global information of the image.

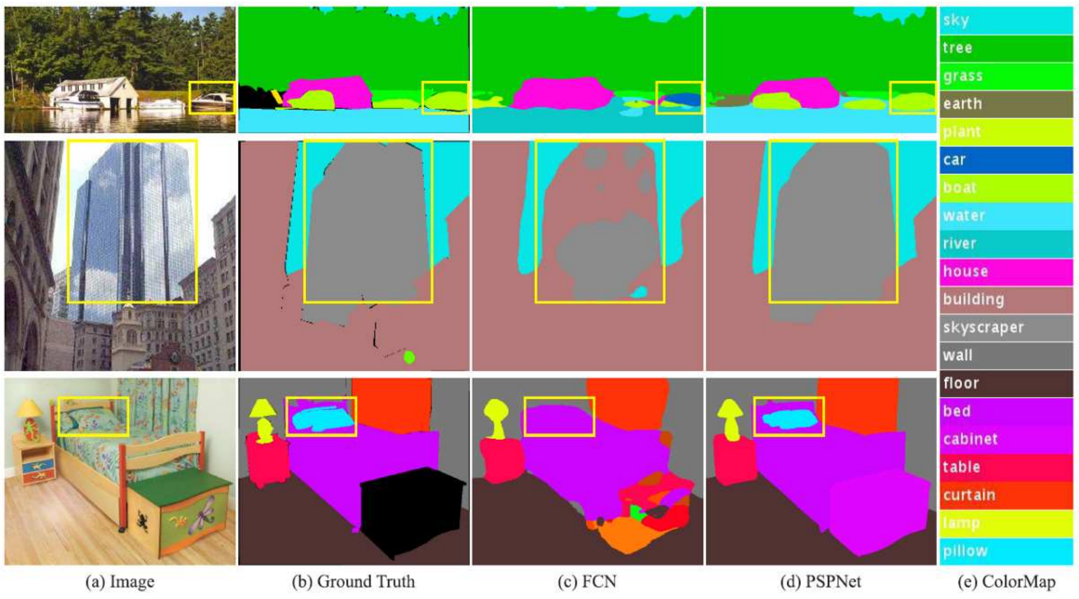


Figure 2.10: (c) Original FCN without Context Aggregation, (d) PSPNet with Context Aggregation Tsang (2018)

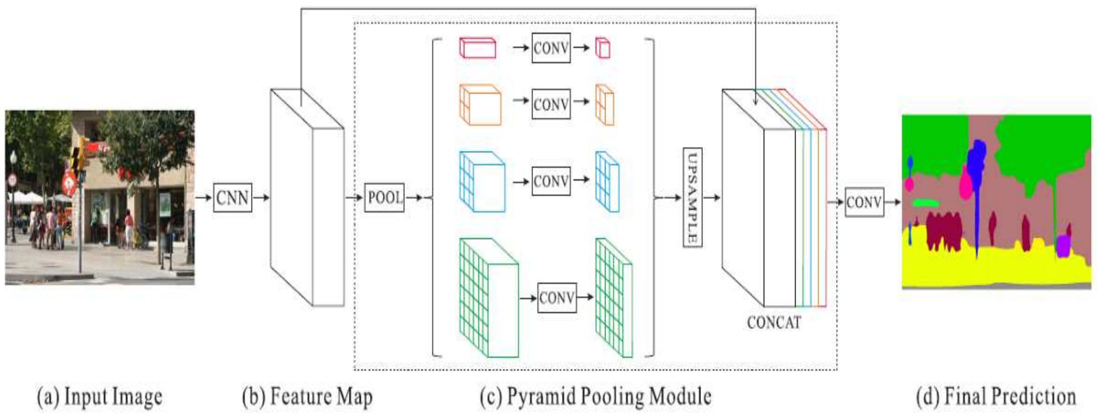


Figure 2.11: Pyramid Pooling Module After Feature Extraction (Colors are important in this figure! Zhao *et al.* (2016))

2.4.2 Pyramid Pooling Module

Input image(a) and resnet(b) is used with dilated network strategy (DeepLab / DilatedNet) for extracting features. The dilated convolution is following DeepLab. The feature map size is 1/8 of the input image here.

(C):1 - Sub-Region Average Pooling

At (c), **sub-region average pooling is performed for each feature map.**

- **Red**: This is the coarsest level which perform global average pooling over each feature map, to generate a single bin output.
- **Orange**: This is the second level which divide the feature map into 2X2 subregions, then perform average pooling for each sub-region.
- **Blue**: This is the third level which divide the feature map into 3X3 sub-regions, then perform average pooling for each sub-region.
- **Green**: This is the finest level which divide the feature map into 6X6 sub-regions, then perform pooling for each sub-region.

(C):2 - 1X1 Convolution for Dimension Reduction

Then **1X1 convolution** is performed for each pooled feature map to **reduce the context representation to 1/N of the original one** (black) if the level size of pyramid is N.

- In this example, N=4 because there are 4 levels in total (red, orange, blue and green).
- If the number of input feature maps is 2048, then the output feature map will be (1/4)X2048 = 512, i.e. 512 number of output feature maps.

(C):3 - Bilinear Interpolation for Upsampling

Bilinear interpolation is performed to up-sample each low-dimension feature map to have the same size as the original feature map (black).

(C):4 - Concatenation for Context Aggregation

All different levels of up-sampled feature maps are concatenated with the original feature map (black). These feature maps are fused as global prior. That is the end of pyramid pooling module at (c).

Finally, it is followed by a convolution layer to generate the final prediction map at (d) Tsang (2018). The idea of sub-region average pooling actually is quite similar to Spatial

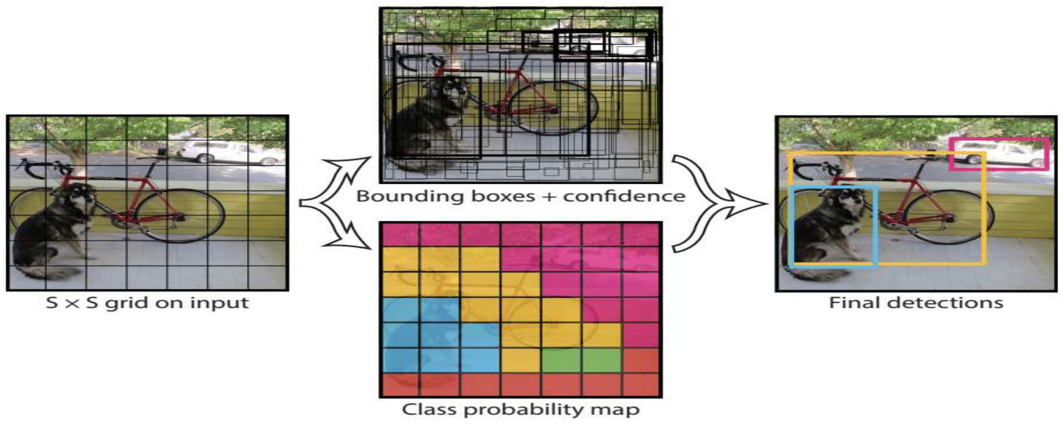


Figure 2.12: YOLO Redmon *et al.* (2015)

Pyramid Pooling in SPPNet. The 1X1 convolution then concatenation is quite similar to the depth-wise convolution in Depth-wise Separable Convolution used by Xception or MobileNetV1 as well, except that bi-linear interpolation is used to make all the feature maps' sizes equal.

2.5 You Only Look Once(YOLO)

All previous object detection algorithms use regions to localize the object within the image. The network does not look at the complete image. Instead, parts of the image which have high probabilities of containing the object. YOLO Redmon *et al.* (2015) is an object detection algorithm much different from the region based algorithms. In YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes.

This algorithm is used in this paper for segmenting the images where the normal state of art networks fail to classify an object as single class.

Working of algorithm is we take an image and split it into an SxS grid, within each of the grid we take m bounding boxes. For each of the bounding box, the network outputs a class probability and offset values for the bounding box. The bounding boxes having the class probability above a threshold value is selected and used to locate the object within the image.

CHAPTER 3

Evaluation metrics and benchmark datasets

3.1 Evaluation metrics

When evaluating a standard machine learning model, we usually classify our predictions into four categories: true positives, false positives, true negatives, and false negatives. However, for the dense prediction task of image segmentation, it's not immediately clear what counts as a "true positive" and, more generally, how we can evaluate our predictions.

3.1.1 Intersection over Union(IoU)

The Intersection over Union (IoU) metric, also referred to as the Jaccard index, is essentially a method to quantify the percent overlap between the target mask and our prediction output. This metric is closely related to the Dice coefficient which is often used as a loss function during training. image segmentation models. (2018)

Quite simply, the IoU metric measures the number of pixels common between the target and prediction masks divided by the total number of pixels present across both masks.

$$\boldsymbol{IoU = \frac{target \cap prediction}{target \cup prediction}}$$

As a visual example, let's suppose we're tasked with calculating the IoU score of the following prediction, given the ground truth labelled mask. The intersection ($A \cap B$) is comprised of the pixels found in both the prediction mask and the ground truth mask, whereas the union ($A \cup B$) is simply comprised of all pixels found in either the prediction or target mask.

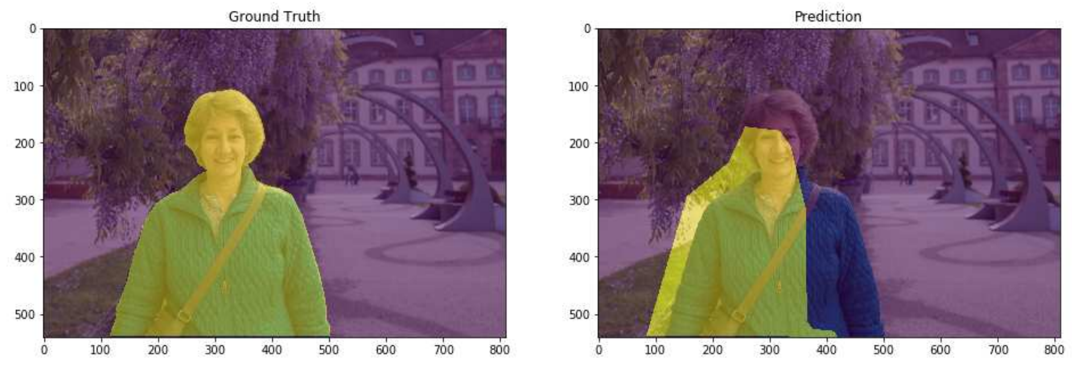


Figure 3.1: IoU - 1
Source

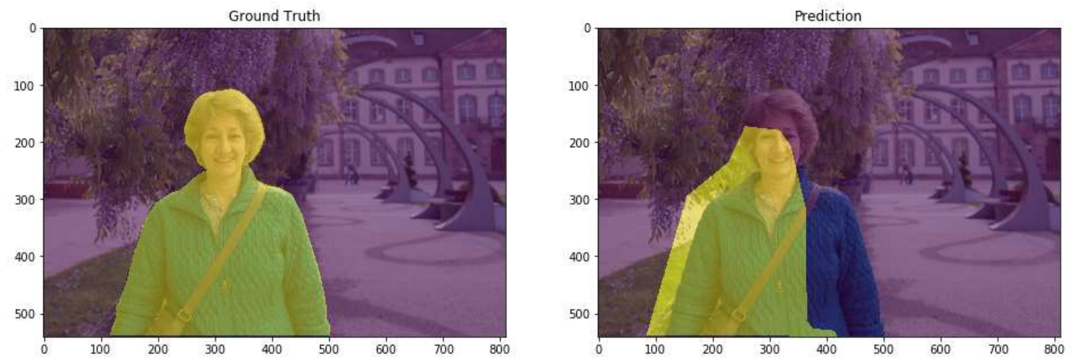


Figure 3.2: IoU - 2
Source

3.1.2 Pixel Accuracy

An alternative metric to evaluate a semantic segmentation is to simply report the percent of pixels in the image which were correctly classified. The pixel accuracy is commonly reported for each class separately as well as globally across all classes.

When considering the per-class pixel accuracy we're essentially evaluating a binary mask; a true positive represents a pixel that is correctly predicted to belong to the given class (according to the target mask) whereas a true negative represents a pixel that is correctly identified as not belonging to the given class.

$$\boldsymbol{accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

).

3.2 Datasets Used For Training Various Architectures

The segmentation architectures are trained in various datasets which are openly available to all the researchers. Knowing the which type of dataset where is use is also key factor as many constraints might come into picture.

Two mainly used datasets is explained below:

3.2.1 ISPRS Potsdam dataset

As shown in fig 3.3 Potsdam dataset is an complete aerial view of city divided in to 38 patches of high resolution images having an each of size 6000X6000 px. ISPRS.

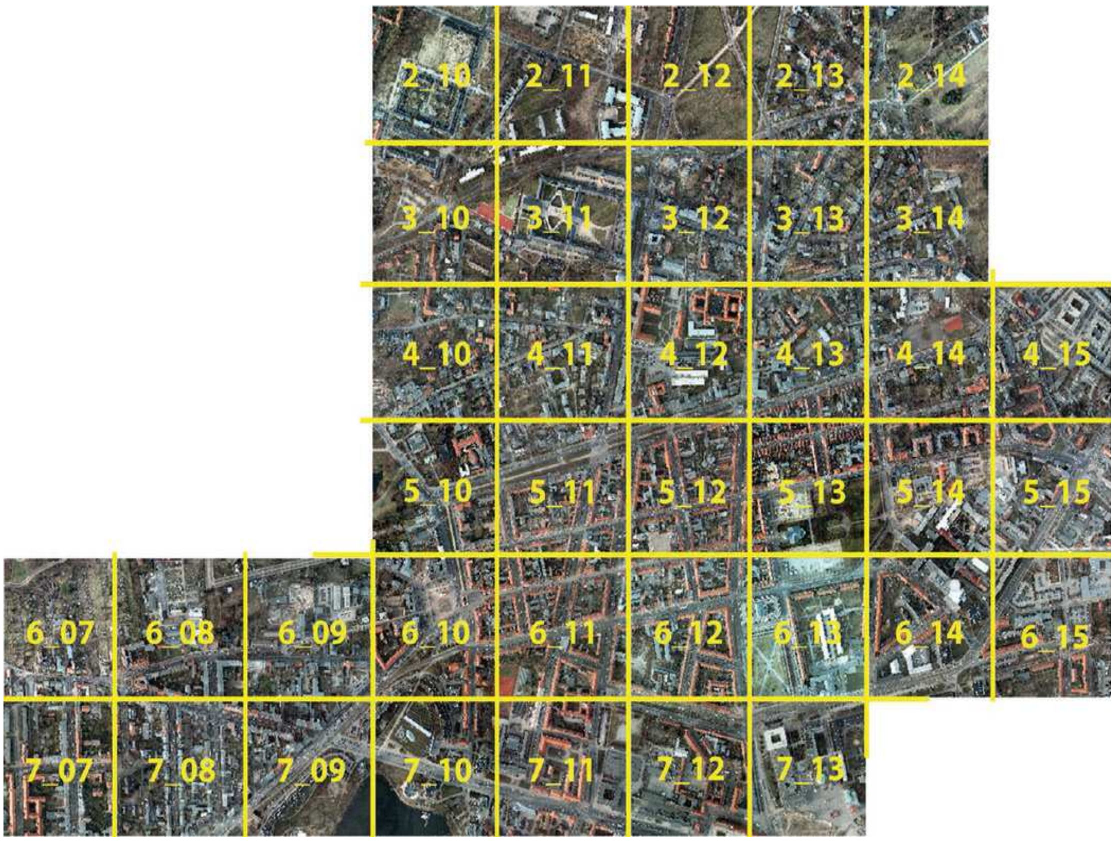


Figure 3.3: An overview of potsdam dataset. ISPRS

Apart from labelling dataset also provides Depth data which is very useful for discriminating high land objects like buildings from the ground surface.

An example of Ariel image depth information and its labelling is shown in fig 3.4.

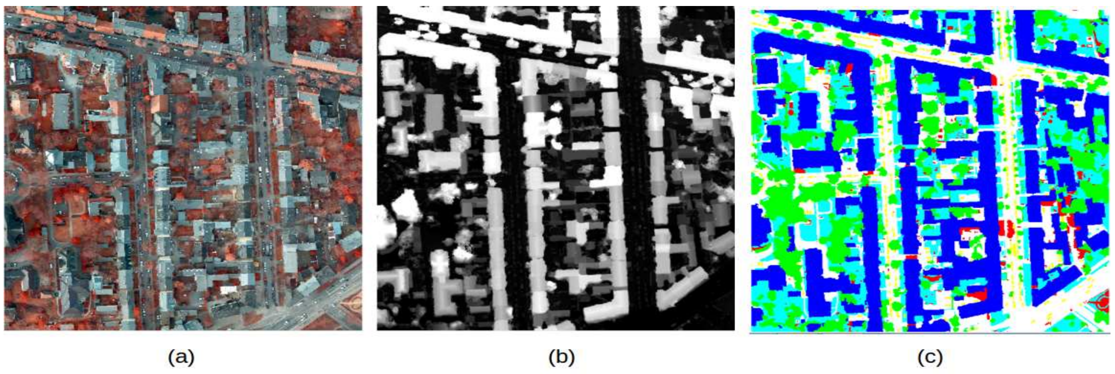


Figure 3.4: Example patches of the semantic object classification contest with (a) true orthophoto, (b) DSM, and (c) ground truth

3.2.2 MIT ADE20K

ADE20K is the largest open source dataset for semantic segmentation and scene parsing, released by MIT Computer Vision team. Zhou *et al.* (2018)

This dataset consist of a vast number of 150 classes almost coveting each and every object, scene and different types of backgrounds.It has about 20,210 train images along with the labels and 2000 validation images.

It has different levels of masking for using the dataset in various application. The following example as shown in fig 3.5 has two part levels. The first segmentation shows the object masks. The second segmentation corresponds to object parts (body parts, mug parts, table parts, ...). The third segmentation shows parts of the heads (eyes, mouth, nose, ...).

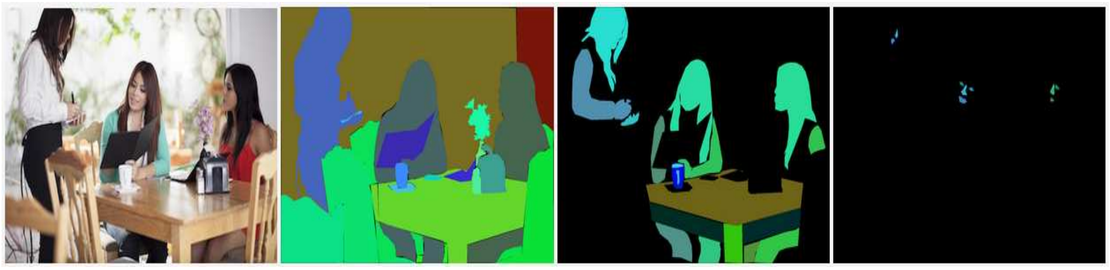


Figure 3.5: A Labelled image of mit ade dataset

CHAPTER 4

Aerial image segmentation and Results

Semantic segmentation can be applied on various scenes like aerial images, Autonomous Driving, Parking lot,Medical Imaging,Scene Understanding etc.

Through out the course project the main focus was on segmentation of aerial images,Autonomous Driving and Parking lot, the types of datasets, architecture used and problems faced during training and testing and is discussed in the below sections.

Availability of high-resolution remote sensing data has opened up the possibility for interesting applications, such as per-pixel classification of individual objects in greater detail. By the use of Convolution Neural Network (CNN), segmentation and classification of images has becomes very efficient and smart.

4.1 Using RefineNet

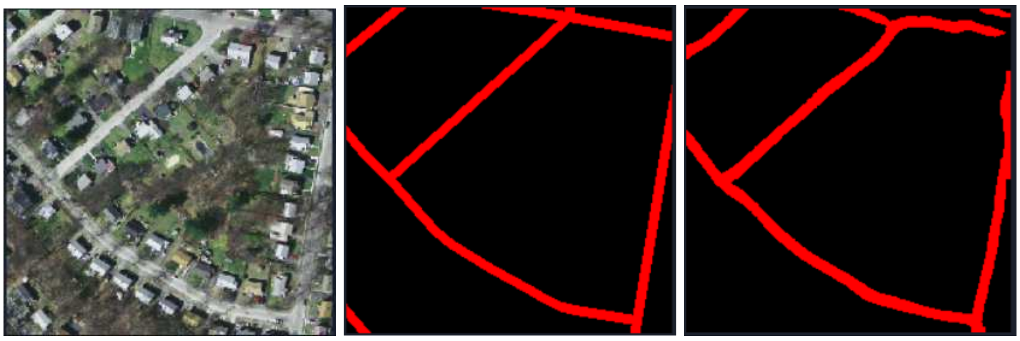
- As discussed in section 2.3.2 for the purpose of input image of the architecture, we crop the high resolution images from the ISPRS dataset. so that processing could be easily done in a moderate level computer with normal specifications. 3.2.1
- From each 6000X6000 high resolution image, 144 500X500 images are cropped in to input images and are resized to 244X244 before giving them as input to the **Refine-Net**.

RESULTS:

Initially at the start of project only 2 class images are taken instead of using all the classes like only road/buildings and background.

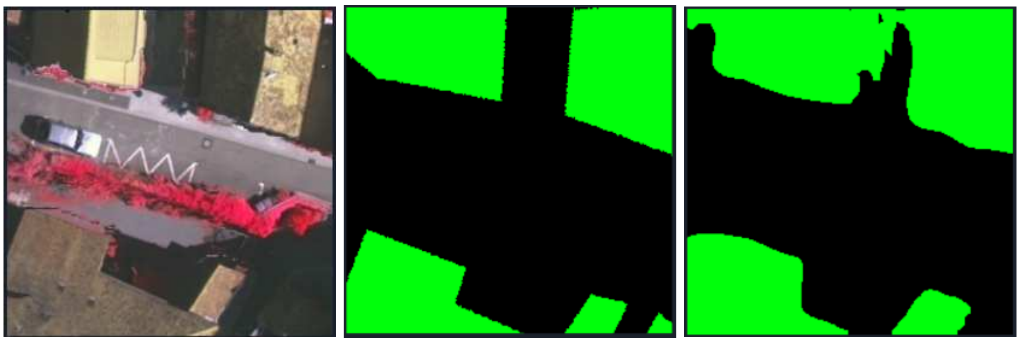
The output segmented images is shown in Fig 4.1 and Fig 4.2. We can see good amount of intersection between the two images.

As in Fig 4.1a the Input image is segmented as 4.1c with approximately 90 percent intersection. Same is the case for image 4.2a, it is segmented as 4.2c.



(a) Input Image (b) Ground Truth (c) Predicted Image

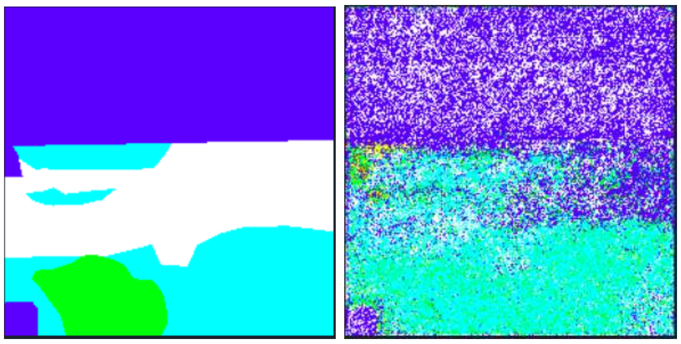
Figure 4.1: Segmented road and non-road class



(a) Input Image (b) Ground Truth (c) Predicted Image

Figure 4.2: Segmented Building and non-Building class

As we increase the number of classes the refine net doesn't seem to converge with some known learning rate parameters. The segmented output diverges from the ground truth as represented in the Fig 4.3b.



(a) Ground Truth (b) Segmented Output

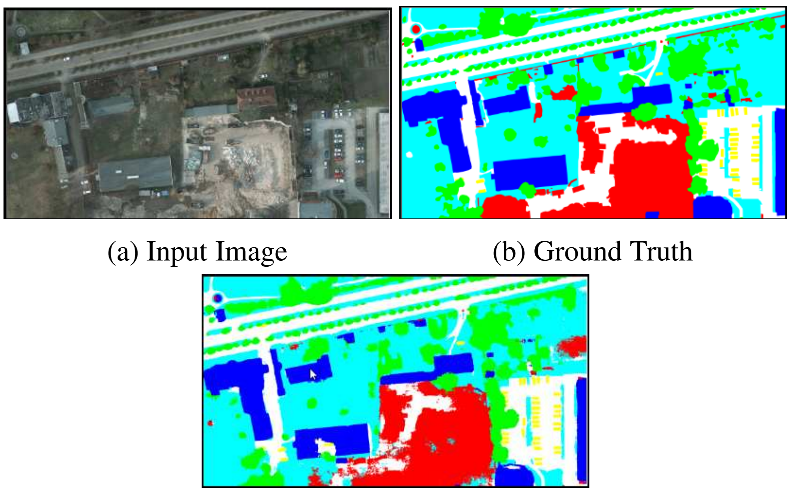
Figure 4.3: Segmented Building and non-Building class

4.2 Using SEGNET

When Encoder-Decoder segnet is used on the dataset which is made same as refinet dataset by cropping it produced significantly better results on those images.

Training a big network with encoders and decoder was not a easy task. The weights of network has been updated by training it from starch as most of the pre-trained models are not trained for aerial images they are trained for object recognition/person recognition etc.

Few of the difficulties faced while training this network from the scratch are listed below in the section 4.2.2.



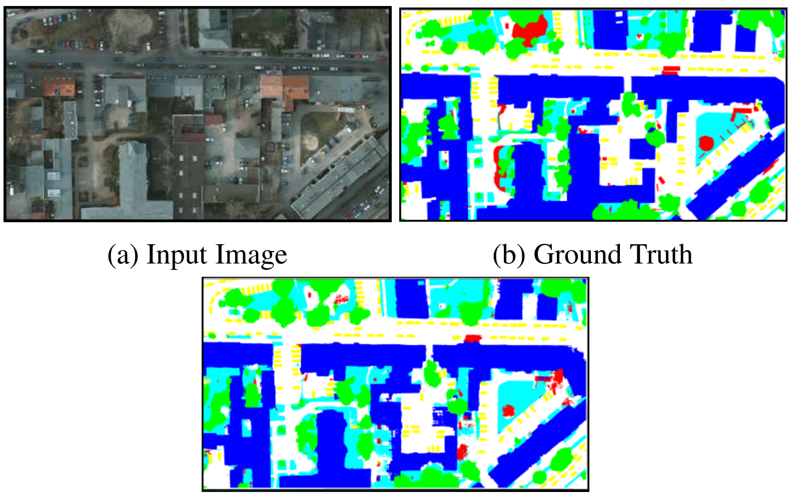
(a) Input Image (b) Ground Truth

(c) Predicted Image

Figure 4.4: Example 1 of segnet on aerial image

Label Name	Red	Green	Blue
Impervious surfaces	255	255	255
Buildings	0	255	255
Low vegetation	0	255	255
Trees	0	255	0
Cars	255	255	0
Clutter	255	0	0
Undefined	0	0	0

Table 4.1: ISPRS potsdam labels colour maps


(a) Input Image (b) Ground Truth

(c) Predicted Image

Figure 4.5: Example 2 of segnet on aerial image

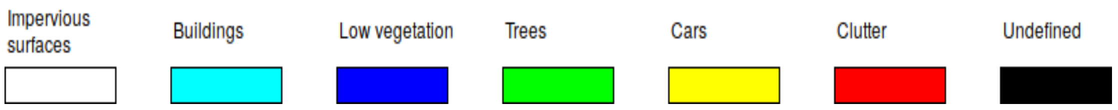


Figure 4.6: Color maps

4.2.1 Difficulties faced

- Network doesn't converge faster.
- Need to change continuously change learning rate.
- If one class converges other class diverges if appropriate learning rate is not given initially.
- As the image was 6000*6000 pixels many cropped 500*500 pixles had complete roads with no other classes in it. So even with correct learning rate, the model was classifying all the classes as road class because of dominance road class at the start. (Tackled it by randomly picking images at the start).

4.2.2 Configuration and Hyper-Parameters

- Train iteration : 250000 and Batch Size : 10.
- Optimizer used is SGD, Learning rate is 10^{-4} and weight decay: 0.0005.
- Multistep learning rate was used where the learning rate was reduced by 10 after 50 epochs and by another 10 and 150 epochs and another 10 after 250 epochs.

CHAPTER 5

College Parking & Autonomous Driving segmentation and Results

Both College parking and autonomous driving need very high intersection score between ground truth and prediction.

Also the general scene might contain very high number of objects / classes hence in this situation a dataset with large number of classes and high train images dataset is need. So for this purpose MITADE20K as said in 3.2.2 is taken which contains about 20K train images and 150 classes.

For this problem statement we need to use the state of art architecture for accurate and high segmentation intersection score.So PSPNET which is currently one of the state-of-art in semantic segmentation is used as Decoder and ResNet100 is used as encoder. with some slight modifications. Hence ADEK dataset is trained on the architecture and we obtained a model for applying it on various scene segmentations.

Results

5.1 Segmentation Ignoring Few Classes

For instance when we consider image as shown in fig 5.1 when we want to segment out vehicles, persons, road, trees and ground surfaces it almost segments image up-to 90 IOU score. After passing the image through trained model.

As seen in Fig 5.1 the architecture segments out completely the vehicles, zero false negatives and it segments a part of slab as False positive because it has some depth and it is in rectangular shape which slight has the features of car but it shows almost zero false negatives which is quite useful for autonomous driving.

As seen in Fig 5.2 the architecture is able to segment accurately the person, vehicles, trees and also the ground part(Light grey colour) and road part(brown colour) as we

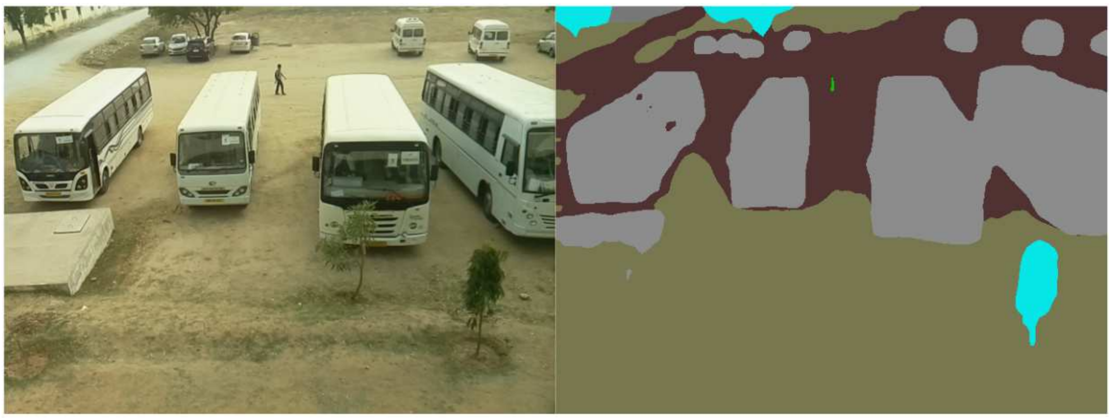


Figure 5.1: **Input Image(Left) & Segmented Image(Right)**

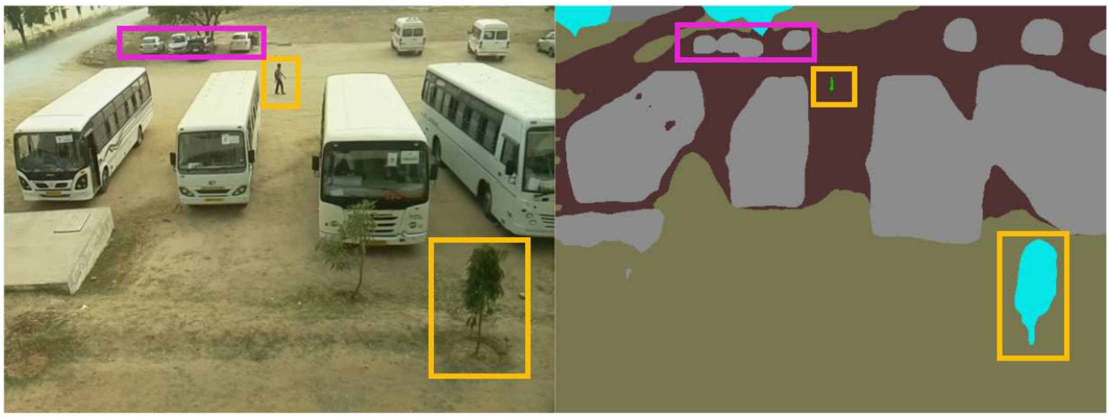


Figure 5.2: **Input Image(Left) & Segmented Image(Right)**

ignored around 144 classes(Initially the dataset contains of 150 classes of which we ignored 144 classes and considered).

5.2 Segmentation considering all the classes

But when we consider all the classes with out ignoring any of the classes unlike in the above method where we ignored class of vehicles (like we ignored cars,bus,vans and trucks and considered all of them as a single vehicle class). Results having multiple classes labelled for single object.

If we consider all the 150 classes then architecture is not able to predict the exact class of the vehicle like weather it is car / truck / bus / van. This issue arises because few buses look like trucks when are viewed from front, and if the image is captured from larger distances it might have the features of car. One more possible reason might be as this is US dataset few of the Trucks might look similar to cars, there are lot of reasons

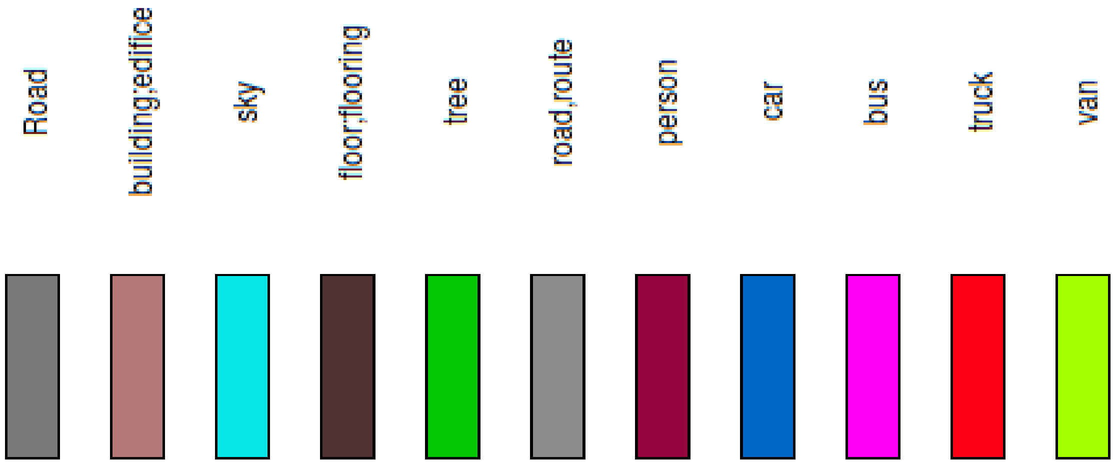


Figure 5.3: Colour map of dataset labels for Fig 4.4-4.11

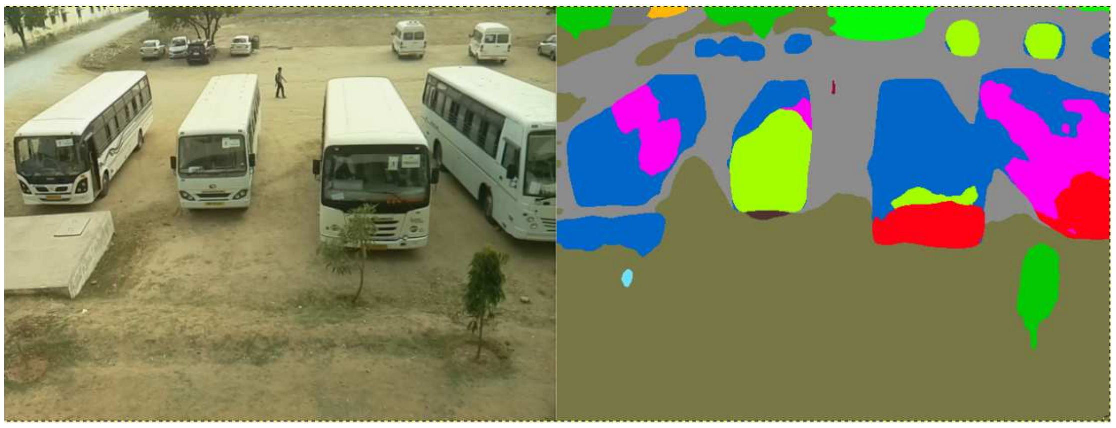


Figure 5.4: **Input Image(Left) & Segmented Image(Right)**

for this issue.

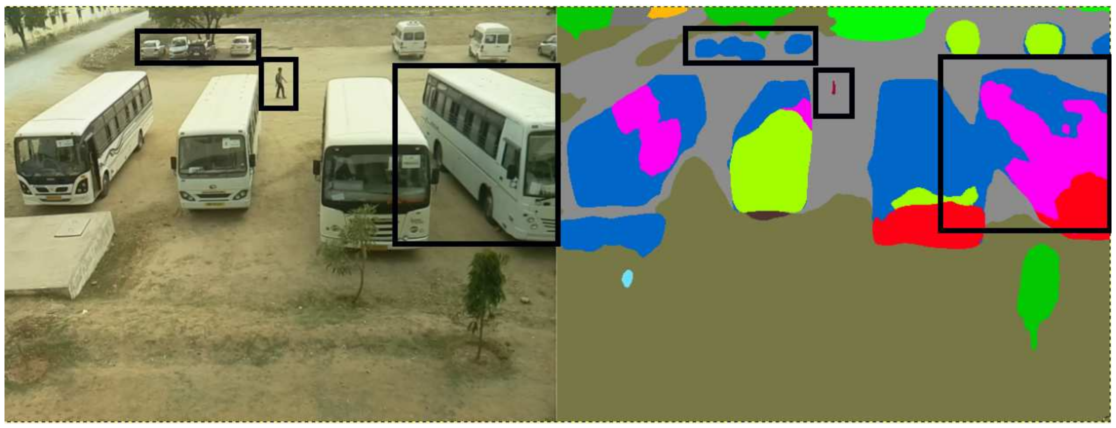


Figure 5.5: **Input Image(Left) & Segmented Image(Right)**

As seen in Fig 5.5 (the bottom right highlighted rectangle) instead of showing it as bus, network is labelling it as 3 classes car(blue), Bus(pink), truck(red) and also the bus to lower left it labelled as 2 classes car(blue), Bus(pink).

5.3 Solving the issue of multi-class labelling

As discussed in previous section multi-class labelling to a single object can arrive due to various reasons. This problem can be solved by using class-imbalance multiplying factor.

Class-Imbalance might used because in the training set of MITADE20K dataset the pixel ratio between buses and cars are in 1/10. So if a object is segmented as both car and bus we count the portion of bus in the object and multiply it with 10 and then compare it with the portion car to make it as one object.

Another issue in the method is getting the exact bounding boxes in a faster way without using large memory.

One quick method which satisfy above criterion is to use Yolo algorithm, we get the bounding boxes of the images in which ever box there is discrepancy among the vehicle classes we can crop that part of box and we again pass it through the network by ignoring all the classes and just keeping only the vehicle class (Car,Bus,Truck,Van). For the output segmented cropped image we apply class imbalance and finally segment it as a single class and put that cropped part back in the image.

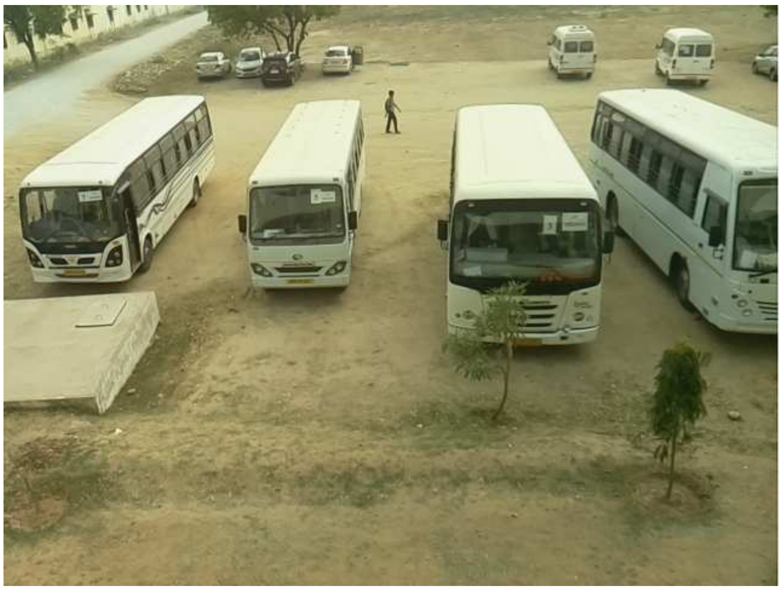


Figure 5.6: Input Image for yolo

As shown in Fig 5.7 we got the bounding box for the bus on the left half of the image which shows 2 classes. In Fig 5.8b we can clearly see it has 2 classes, now we take that bounding box and again pass it through segmentation architecture.

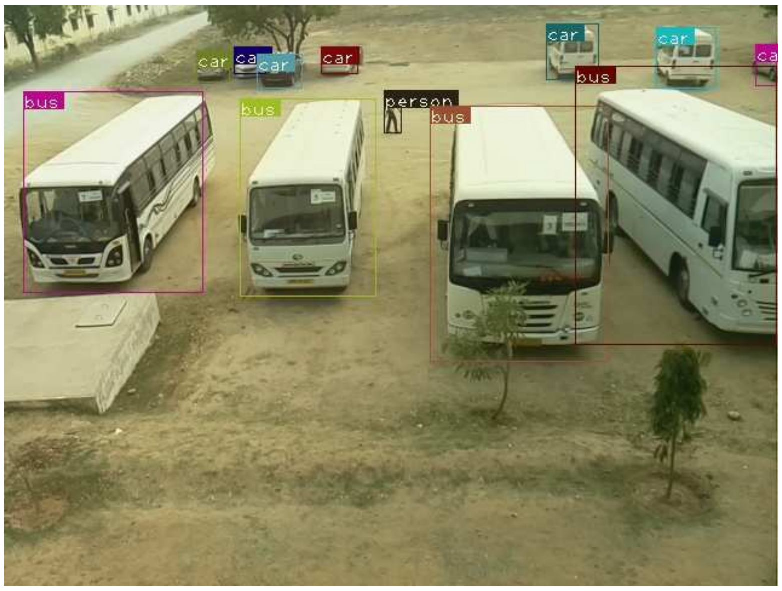
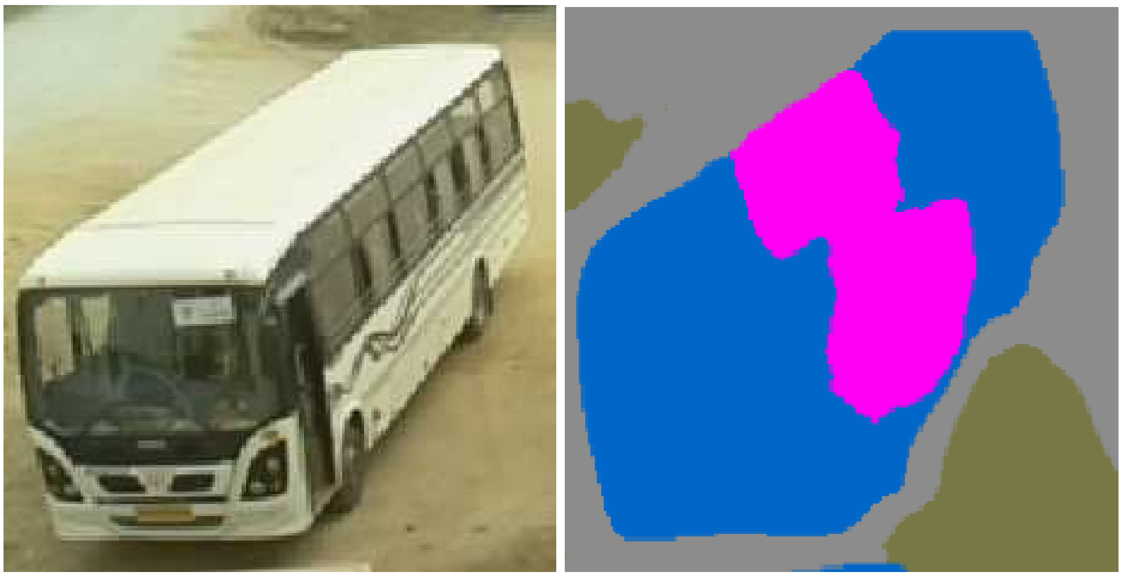


Figure 5.7: Output of yolo with bounding boxes



(a) Bounding box obtained after applying yolo (b) Segmented Output without class imbalance

Figure 5.8: Comparing the bounding box and segmented image without class imbalance.

Now we pass this cropped image through segmentation architecture by considering all the vehicles classes(car,van,bus,truck) and few basic classes like ground surface, sky,floor,grass we are considering 4 extra classes other than vehicle classes because, bounding box doesn't only contain vehicle but it also contains these extra classes, as we are later multiplying with class imbalance factor including these extra class will reduce error by a large margin Shah (2017).

We can clearly see in Fig 5.9 larger portion of the vehicle is classified as bus, so now even without multiplying applying the class imbalance factor we can directly say

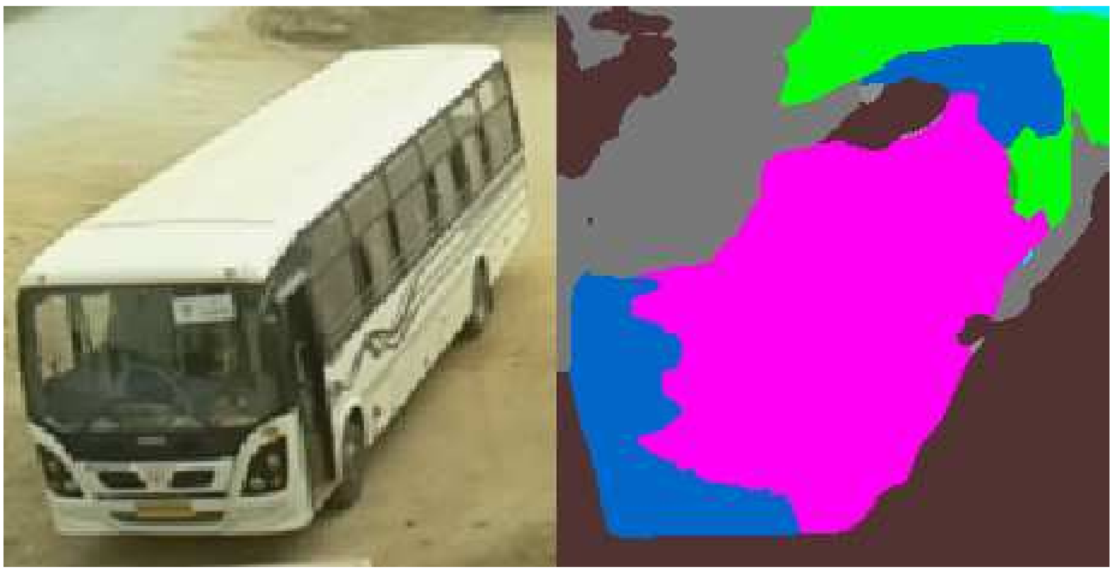


Figure 5.9: Segmentation of cropped image after passing considering vehicle+few general classes

it belongs to bus class.

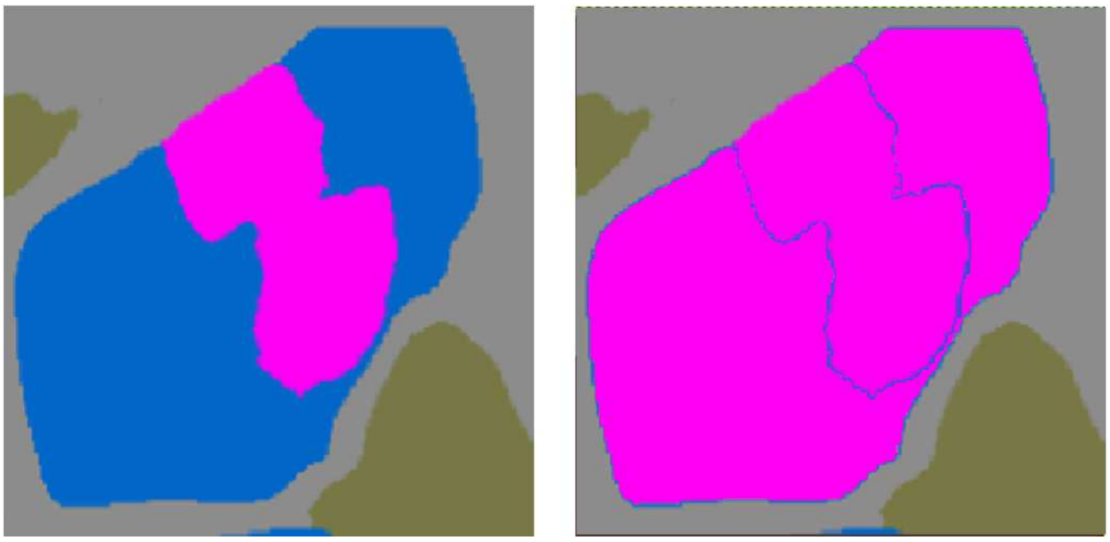
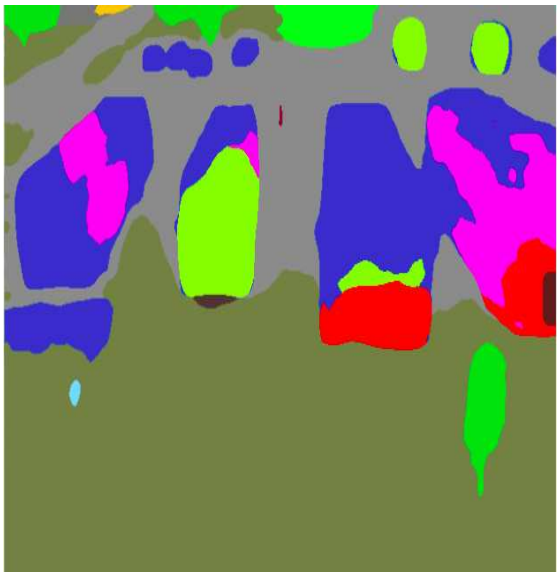


Figure 5.10: Left- Segmented output directly from all the classes
Right -Segmented output of a cropped image by ignoring classes and applying class imbalance in network.

As shown in Fig 5.11b is the final segmented image after applying class imbalance by following a sequence od steps of Yolo detection followed by cropping followed by segmentation and then finally multiplying with class imbalance factor to decide the class.

Another useful feature we get by applying Yolo and segmentation would be, when there are overlap between person and car it segments complete picture as car. but by using yolo we can discriminate person and car by applying architecture separately on cropped


(a) Segmented image without class imbalance.

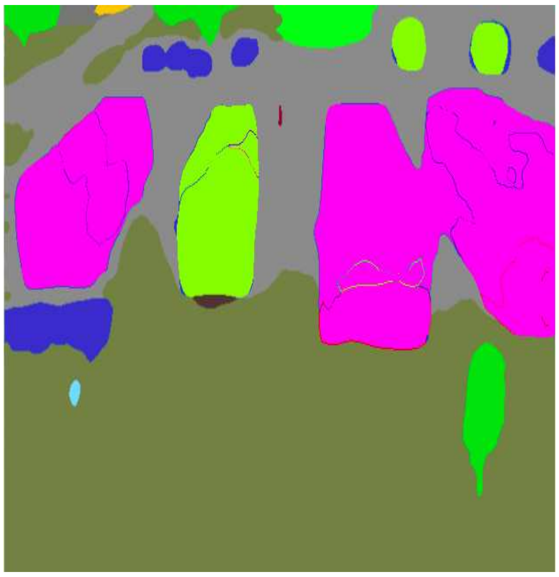
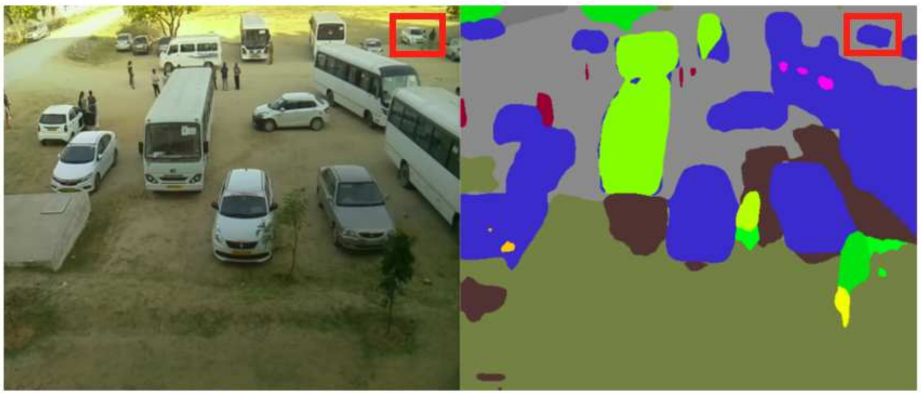

(b) **Final** Segmented output.

Figure 5.11: Comparing the bounding box and segmented image without class imbalance.

image as shown in Below figures.



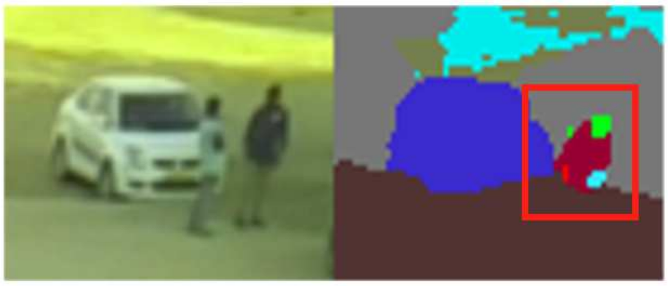
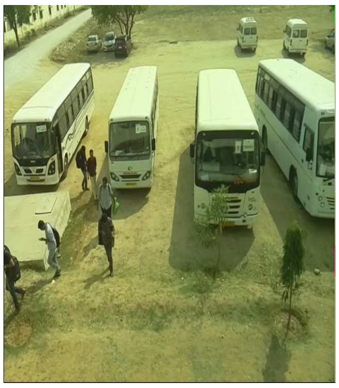
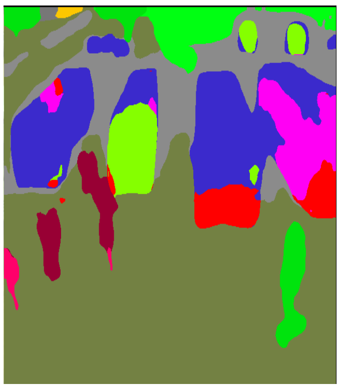


Figure 5.12: Top - Segemtation without cropping and without ignoring classes
Bottom - Segemtation of the cropped without ignoring classes.

5.3.1 Other results and issues


(a) Input image


(b) Without class Imbalance.

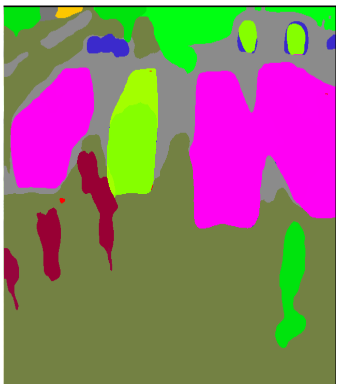
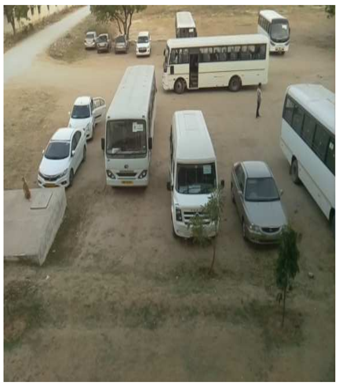
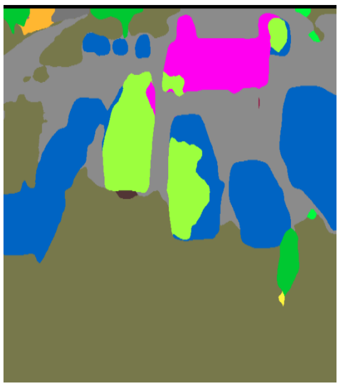

(c) **Final** Segmented output.

Figure 5.13: Output of image after class-imbalance and ignoring few classes


(a) Input image


(b) Without class Imbalance.

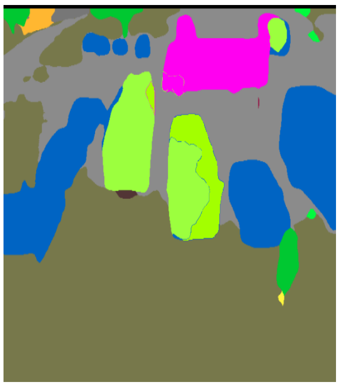

(c) **Final** Segmented output.

Figure 5.14: Output of image after class-imbalance and ignoring few classes

Fig 5.13 and 5.14 shows that when input image (a) is passed through the network without any ignoring any classes and without removing class imbalance it outputs (b) as shown in sub-figures and when we ignore unnecessary classes we get final segmented image as shown in sub-figures (c).

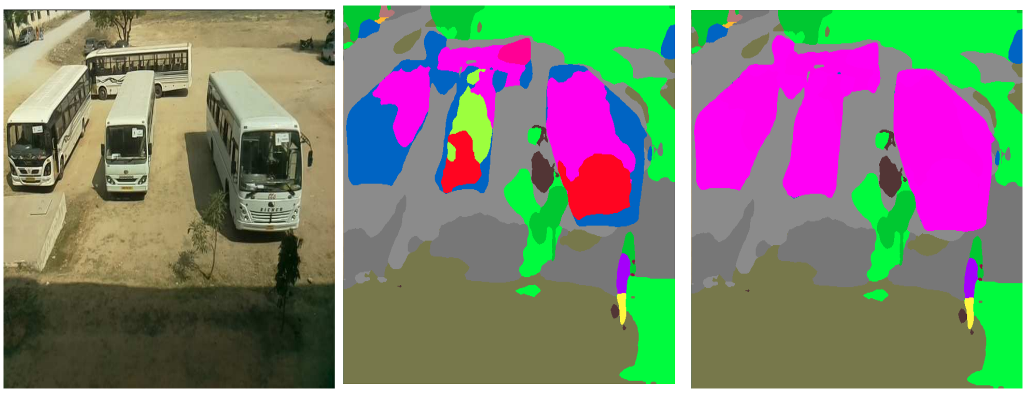

(a) Input image (b) Without class Imbalance. (c) **Final** Segmented output.

Figure 5.15: Results when there is overlap issue

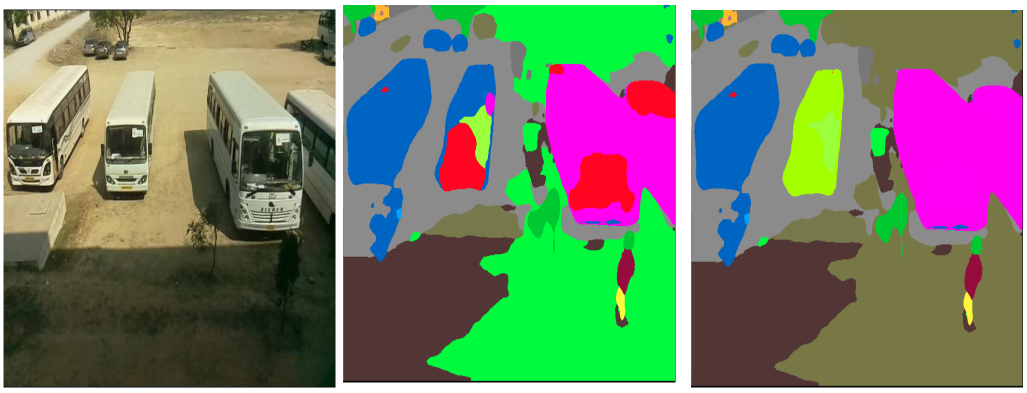
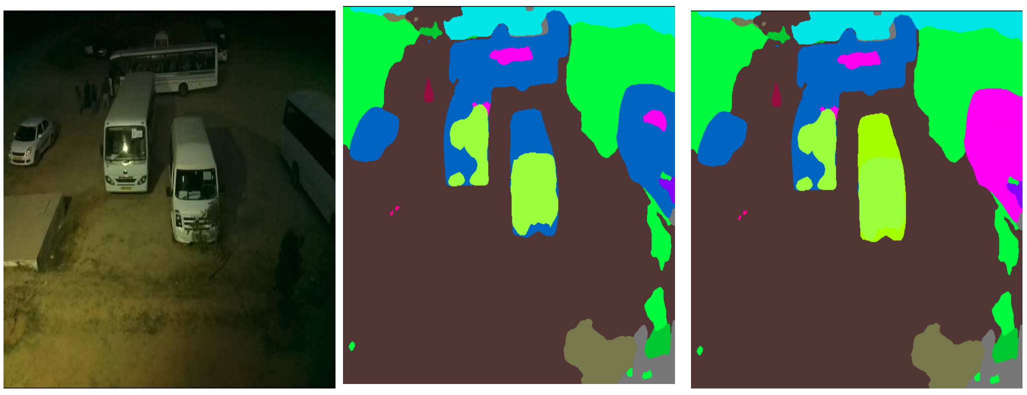

(a) Input image (b) Without class Imbalance. (c) **Final** Segmented output.

Figure 5.16: Results when there is overlap issue

When there is significant amount of overlap between the vehicles of different classes (like bus and car overlapping) as shown in fig 5.15 (a) and 5.16 (a), it is difficult to decide where to draw bounding box and where is apply that bounding box on that image. So this method shows some in-consistency with this situation, with properly designing decision system this problem can be tackled and this problem is marked as **Future work**. As shown in the Final segmented outputs of image it considers everything as a bus class as bus has high class im-balance.



(a) Input image (b) Without class Imbalance. (c) **Final** Segmented output.

Figure 5.17: Application in dark-background

When there is low-light or very high shadowed image segmentation doesn't work properly, the part of image which has good amount of exposure to light segments properly, other objects are difficult to segment. But still it is able to classify them under vehicle classes but not type of vehicle. (Type of vehicle is not a major concern in autonomous driving problem only vehicle and width of vehicle matters most of the time).

CHAPTER 6

SUMMARY

Semantic segmentation is one most important as well challenging task in the field of deep-learning. Unlike labelling complete image as one label in segmentation we need to assign a label to each of the pixel in the image. Aerial image segmentation and road scene image segmentation have been taken as two main problem statements throughout the course of the project. Various deep-learning architectures like refinet, segnet, pspnet have been used and modified accordingly to get desired results. Modifications to existing methods is done by using the techniques of class imbalance. The part of the image which not segmented properly is cropped by using a fast object detection algorithm like yolo and that part of object is cropped and we pass cropped image through the segmentation network by ignoring unlabelled classes from and then finally by removing class imbalance.

REFERENCES

1. **Alom, M. Z.**, **T. M. Taha**, **C. Yakopcic**, **S. Westberg**, **M. Hasan**, **B. C. V. Esesn**, **A. A. S. Awwal**, and **V. K. Asari** (2018). The history began from alexnet: A comprehensive survey on deep learning approaches. *CoRR*, **abs/1803.01164**. URL http://arxiv.org/abs/1803.01164.

2. **Badrinarayanan, V.**, **A. Kendall**, and **R. Cipolla** (2015). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, **abs/1511.00561**. URL http://arxiv.org/abs/1511.00561.

3. **Girshick, R. B.**, **J. Donahue**, **T. Darrell**, and **J. Malik** (2013). Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, **abs/1311.2524**. URL http://arxiv.org/abs/1311.2524.

4. **He, K.**, **X. Zhang**, **S. Ren**, and **J. Sun** (2015). Deep residual learning for image recognition. *CoRR*, **abs/1512.03385**. URL http://arxiv.org/abs/1512.03385.

5. **Huang, G.**, **Z. Liu**, and **K. Q. Weinberger** (2016). Densely connected convolutional networks. *CoRR*, **abs/1608.06993**. URL http://arxiv.org/abs/1608.06993.

6. **image segmentation models., E.** (2018). jeremyjordan. URL https://www.jeremyjordan.me/evaluating-image-segmentation-models/.

7. **ISPRS** (). 2d semantic labeling contest - potsdam. URL http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html.

8. **Lin, G.**, **A. Milan**, **C. Shen**, and **I. D. Reid** (2016). Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *CoRR*, **abs/1611.06612**. URL http://arxiv.org/abs/1611.06612.

9. **Long, J.**, **E. Shelhamer**, and **T. Darrell** (2014). Fully convolutional networks for semantic segmentation. *CoRR*, **abs/1411.4038**. URL http://arxiv.org/abs/1411.4038.

10. **Redmon, J.**, **S. K. Divvala**, **R. B. Girshick**, and **A. Farhadi** (2015). You only look once: Unified, real-time object detection. *CoRR*, **abs/1506.02640**. URL http://arxiv.org/abs/1506.02640.

11. **Shah, M. P.** (2017). Semantic segmentation architectures implemented in pytorch. *https://github.com/meetshah1995/pytorch-semseg*.

12. **Simonyan, K.** and **A. Zisserman** (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, **abs/1409.1556**. URL http://arxiv.org/abs/1409.1556.

13. **Tsang, S.** (2018). Pspnet. URL https://bit.ly/2JEtFgK.

14. **wikipedia** (2019). Vanishing gradient problem. URL https://en.wikipedia.org/wiki/Vanishing_gradient_problem.

15. **Zhao, H.**, **J. Shi**, **X. Qi**, **X. Wang**, and **J. Jia** (2016). Pyramid scene parsing network. *CoRR*, **abs/1612.01105**. URL http://arxiv.org/abs/1612.01105.

16. **Zhou, B.**, **H. Zhao**, **X. Puig**, **T. Xiao**, **S. Fidler**, **A. Barriuso**, and **A. Torralba** (2018). Semantic understanding of scenes through the ade20k dataset. *International Journal on Computer Vision*.