LANE DETECTION USING DEEP LEARNING

PRESENTED TO
Aries

PRESENTED BY
Himank garg
Nitin Goyal
Priyanshu Maurya
Sagar Kumar

The Problem Statement

We are given an image of a road and we want to distinguish the Drivable Lane area in that image from the obstacles and the background

Input Image

Output Image

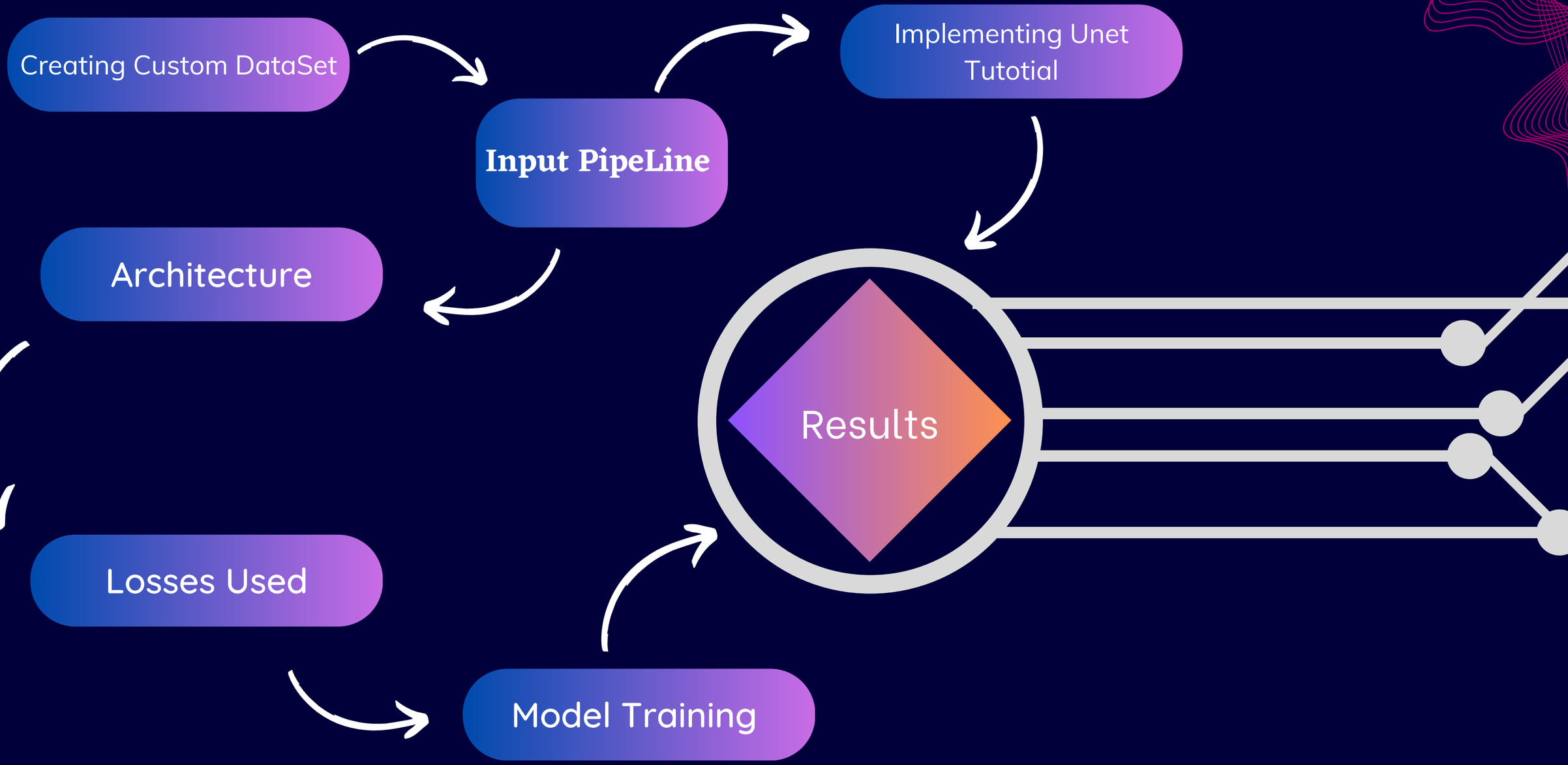

Creating Custom DataSet
Input PipeLine
Implementing Unet Tutotial
Architecture
Results
Losses Used
Model Training

Image Segmentation

Lane detection can be done using image segmentation,which segments an image into multiple areas by assigning a label to every pixel of the image.

We just need to differentiate pixels of lane, objects and background. Hence all others pixels of trees,sky,buildings are part of background

We will be using the UNet Model for image segmentation

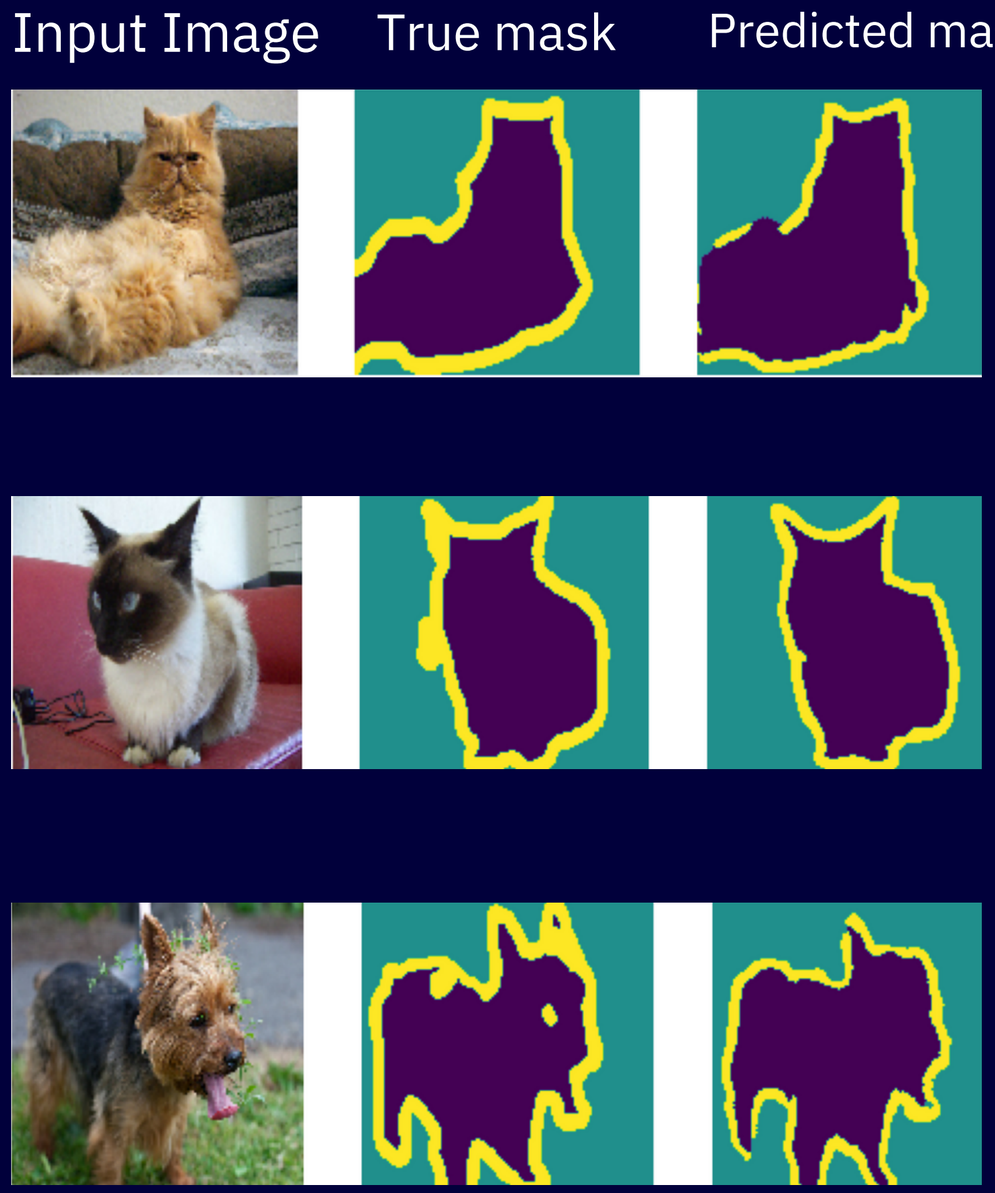
We first implemented a tutorial on UNet to see the results

Tutorial reference-https://pyimagesearch.com/2022/02/21/u-net-image-segmentation-in-keras/

Image Segmentation

We trained the model for on the Oxford IIIT pet Datset for 15 epochs,batch size 64 and usign cross entropy loss function.

The result was satisfactory as shown and we just need a labelled dataset of lanes which can be used to train the model .



Datset Ref-https://www.robots.ox.ac.uk/~vgg/data/pets/

DATASET

We used BDD100K dataset as our base
to create a custom dataset.

The label images of BDD100K dataset were modified in
such a way that, there are only three colors (pink,blue and black)
in the label representing lane area,obstacles and background
respectively.

Creating Custom Dataset

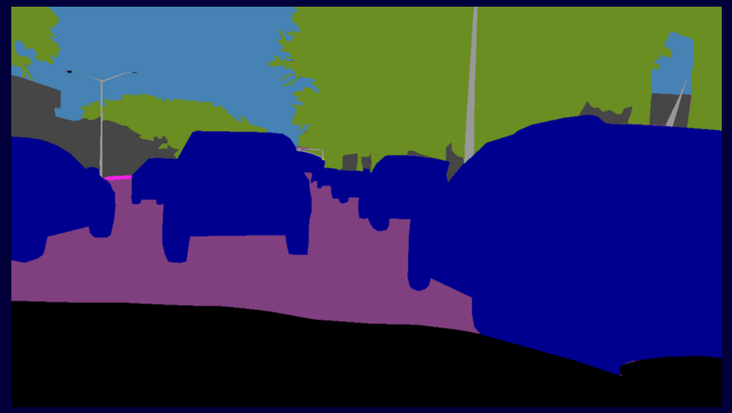
Code

```
# Loop through each pixel and change color if needed
for y in range(rgb_im.height):
    for x in range(rgb_im.width):
        pixel_color = rgb_im.getpixel((x, y))
        if pixel_color not in keep_colors:
            rgb_im.putpixel((x, y), (0,0,0))  # Change color to gray
```

BDD100K -https://bdd-data.berkeley.edu/

DATASET

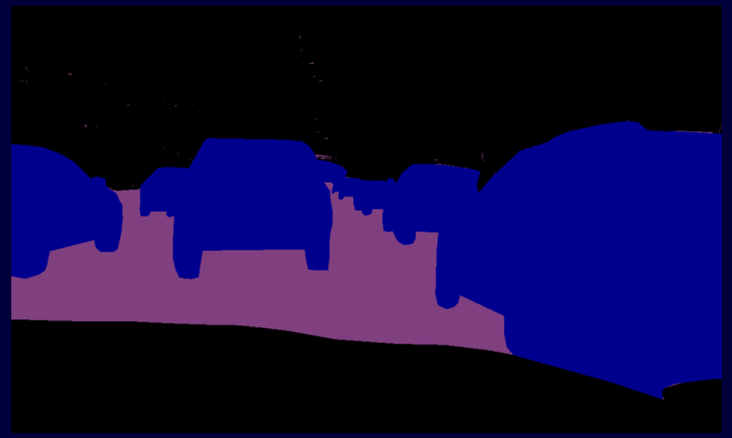
Raw image



BDD100K Label

All pixles other than cars and lane are colored black

Raw image



Custom Dataset Label

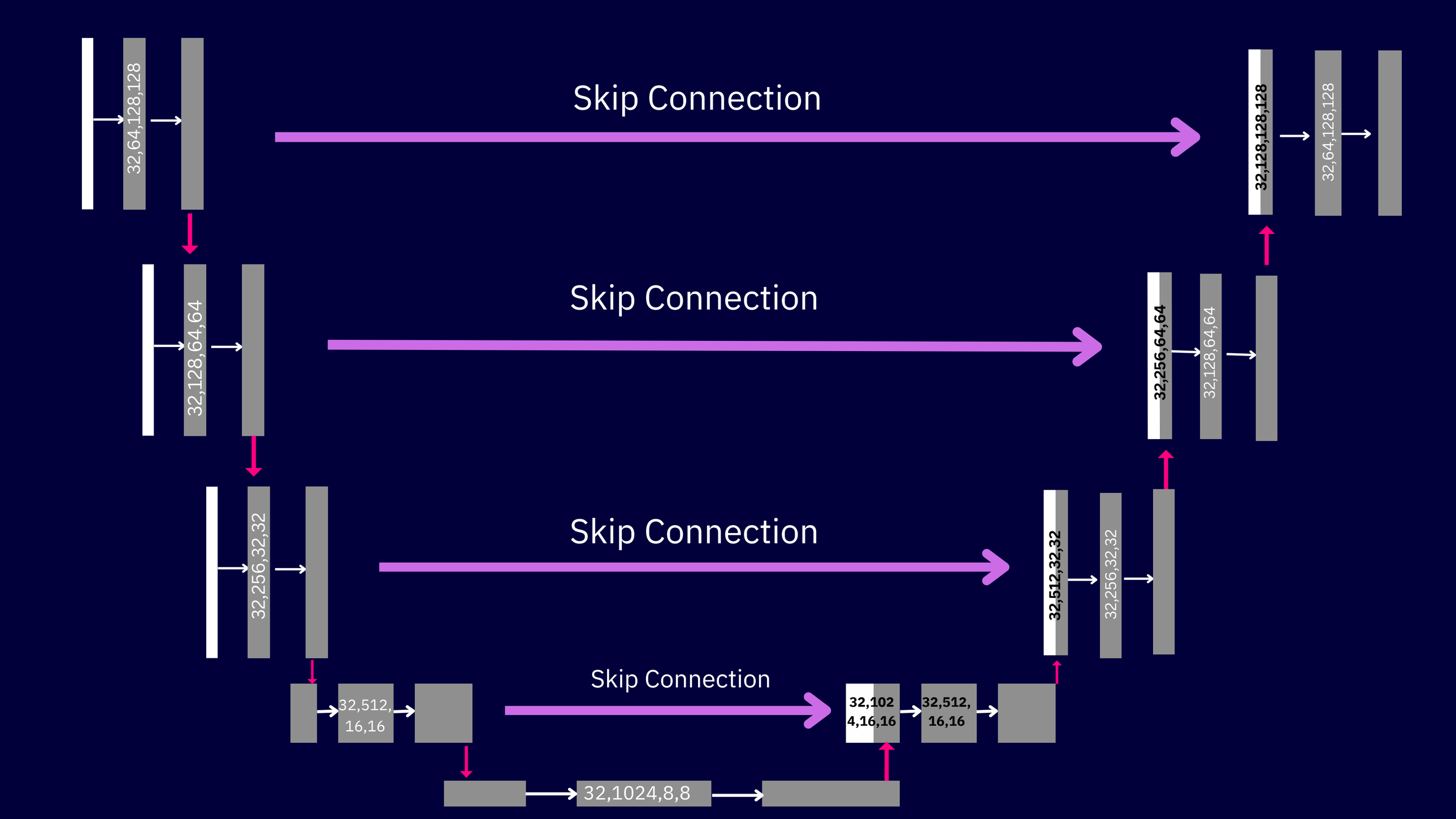
UNet Model

Input: N x 3 x H x W where N is the batch size, H and W are the height and width of the input image respectively.

Image Width-128

Image Height-128

Batch Size- 32


32,64,128,128
Skip Connection
32,128,128,128
32,64,128,128
32,128,64,64
Skip Connection
32,256,64,64
32,128,64,64
32,256,32,32
Skip Connection
32,512,32,32
32,256,32,32
32,512,
16,16
Skip Connection
32,102
4,16,16
32,512,
16,16
32,1024,8,8

Loss Functions

The Loss function used is combination of Cross Entropy loss and Mean Squared Loss(MSE).
The contribution of Cross Entroy Loss is 10 times that of MSE Loss

```
combined_loss = ce_loss + 0.1* mse_loss
```

Cross Entropy Loss

$$L(\hat{y}, y) = -\sum_{k}^{K} y^{(k)} \log \hat{y}^{(k)}$$

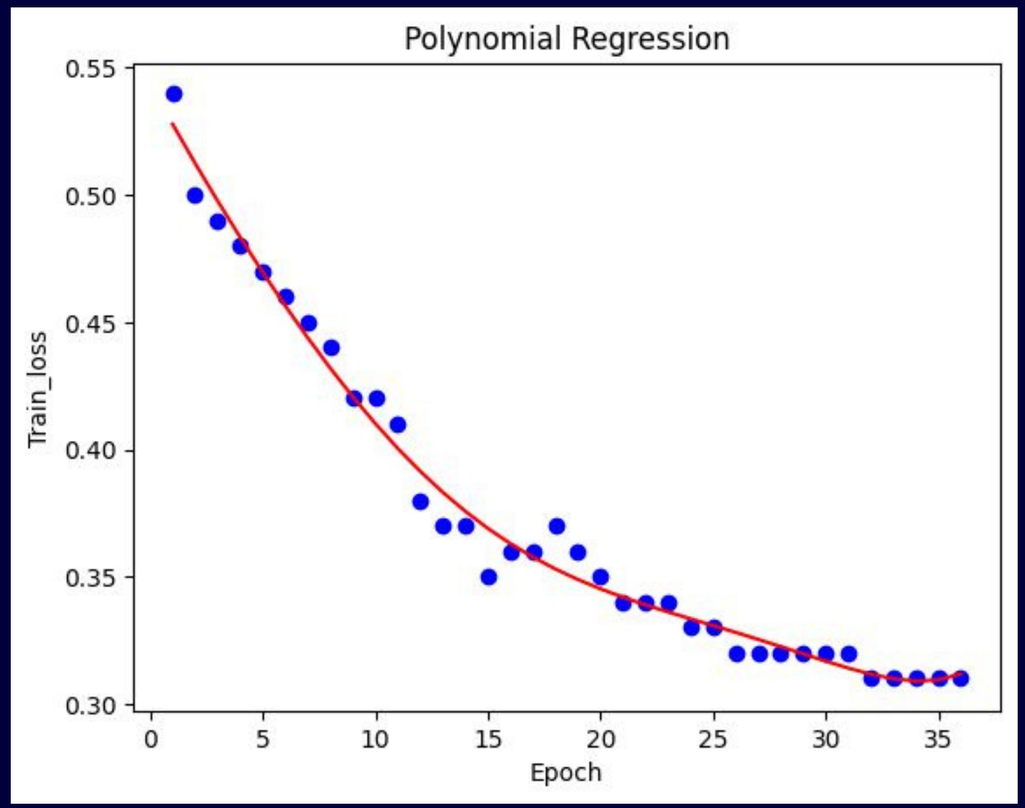
Mean Squared Error Loss

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Training Model

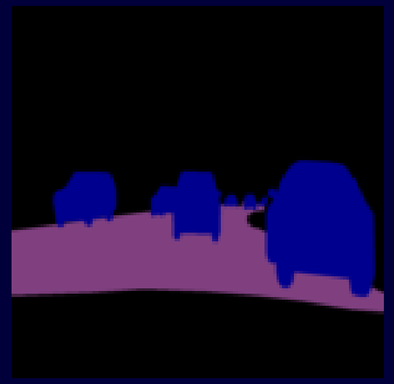
We train the model on the custom datasset with total epochs 100, bacth size 32 using the combinbed loss function

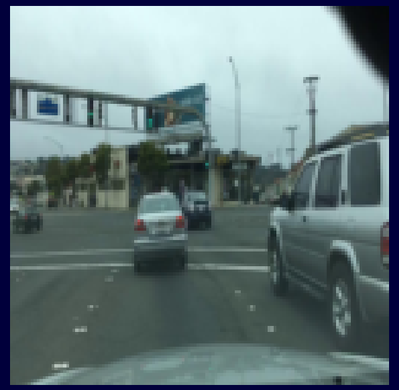
Loss VS Epochs



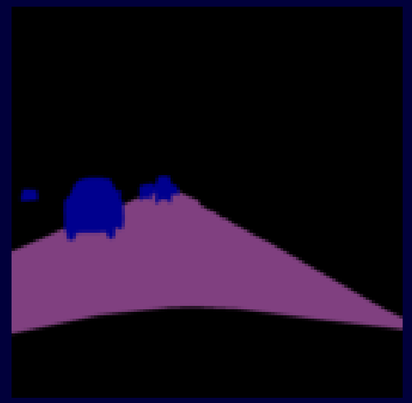
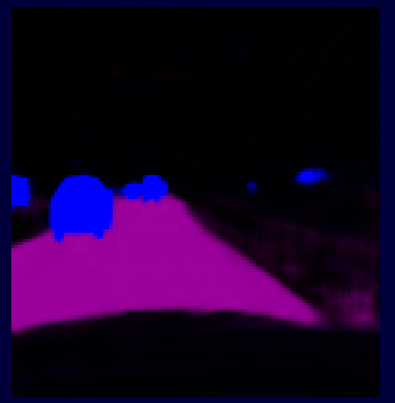
Testing

Input image | True Mask | Predicted Mask





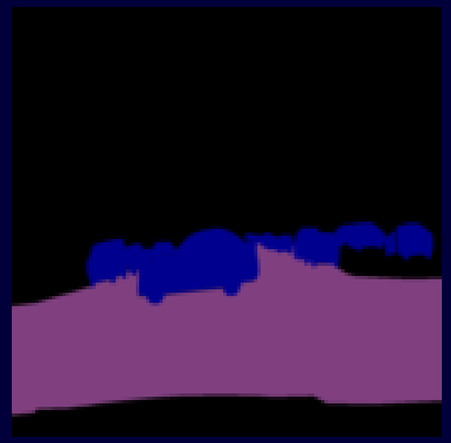



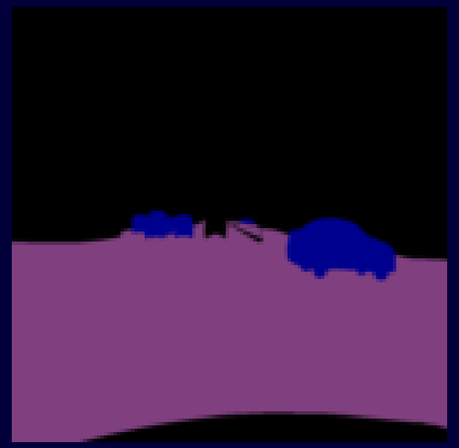
*images are resized to 128X128, hence appearing blur

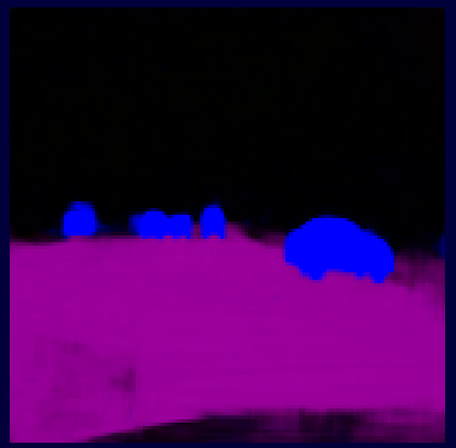
Input Image

True Mask

Predicteed Mask







*images are resized to 128X128, hence appearing blur

Result

The obtained results are very satisfactory as you can see

The Model could be improved if it can distinguish the pixels of sky, builidings, streetlights, grass etc as different instances rather than all as part of background. But it will require more computation power

Still the model performed very well given the less computation power we have.