# DeepLearning Final Report

*Lane Detection For Autonomous Vehicles*


**Azhar khan(B21CS087)**

**Deepanshu(B21CS021)**

## Problem Statement

Lane Detection is a Crucial part for building fully autonomous Systems as it provides the guide rails for the vehicles to maintain an order while traveling. So a robust and fast inference method is required for a better autonomous system.

## Solution strategy (with the motivation for the strategy)

Various solutions exist to this problem from traditional Computer vision approaches to latest Deep-Learning approaches. The possible Solution that we thought of were using semantic segmentation and creating a probability mask of the lanes as these methods are robust in nature. For simple inference we initially used UNet which is a Simple yet powerful Segmentation approach and we used binary mask to achieve this in Later Stages we have also looked at instance segmentation approach through Enet. And later we have explored the possibility of a custom loss function.

## Dataset

As Discussed in the Intermediate Reports we have used TUSimple as our data set as we had the flexibility of various benchmarks available on the dataset.Some other dataset that we explored were CULane and Comma2k19.

## Major innovations/ contributions

We thought of three possible ways to introduce innovation in the already existing solution.

1. Custom Loss function.
2. Determining Driveable Area with training any segmentation model.
3. Introducing an Conv-LSTM model for more robust inference of the lanes.

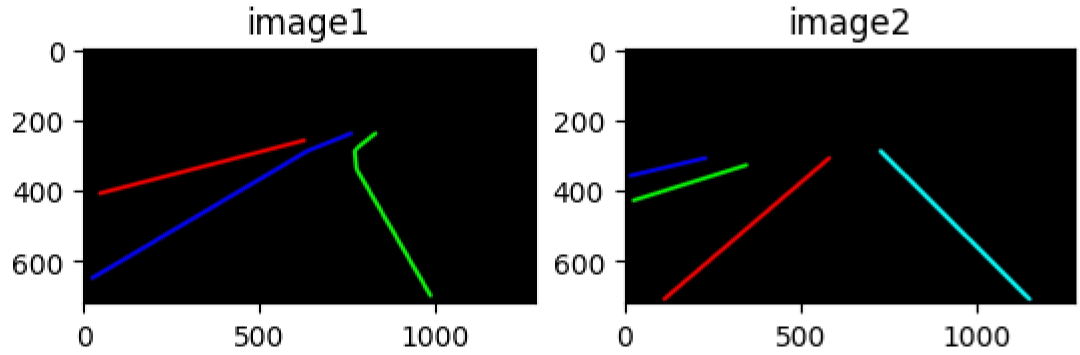
**Custom Loss Function:**

**Intuition:** Lane detection is similar to a line Fitting problem like a regression problem. We use CrossEntropyLoss for better Image generation But don't emphasize the lane lines much more. Hence we have devised a loss function which gives extra weightage to lane fitting.

**Algorithm:** Our Algorithm will take predicted and ground truth then divide the Images into different images equal to no of lanes in each image and then we will take XOR of different lane wise images and then calculate distance between points that have value set to high and the total distance will be our loss when we will minimize this loss Our lanes will fit on each other.

1. There are 5 Max lanes possible. We are segregating the Image and dividing them according to the number of lanes present.(eg: 4 lanes then 8 images 4 for prediction and 4 for ground truth which contains individual lane mask.
2. We will take XOR of the respective images and will calculate the distance between the pixels which are high in the resultant. We have used XOR as many edge cases and exceptions are neglected. Another approach could be to sample equi-distance points and calculating the euclidean distance or any other distance metric could be implementation.
3. We have used a HyperParameter alpha with which we could set how much emphasis we want on our loss.

Code implementation of the loss function have been provided.



LOSS between img1 and img2 with alpha=0.6 is 56.14473587772026.

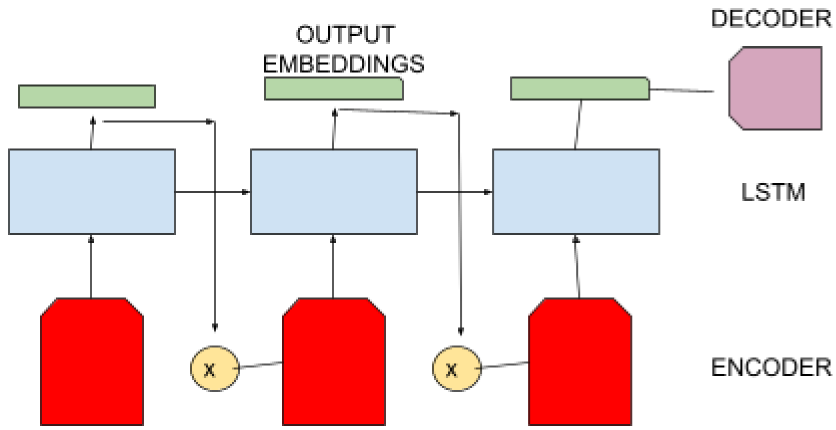
Loss between img1 and img1 with alpha = 0.6 is 0.0

**Determining Driveable Area with training any segmentation model:**

For determining the drivable area we need another segmentation model to predict it. But we have devised an algorithm to do so. Since we know the camera is center aligned to the car the center two lanes detected are of the current lanes and could be marked by coloring the respective polygon.

**Conv-LSTM**

Since the dataset is a continuous feed of images, lanes change gradually and don't change suddenly hence we could take advantage of this temporal nature of the data and integrate an LSTM layer. To predict the lane curve in the further layer rather that predicting frame by frame every time. After an CNN encoder layer a LSTM layer whose output is the mask and the mask is concatenated with the further embeddings and the last LSTM block could be attached with a CNN decoder to get the Output. Illustration given below.



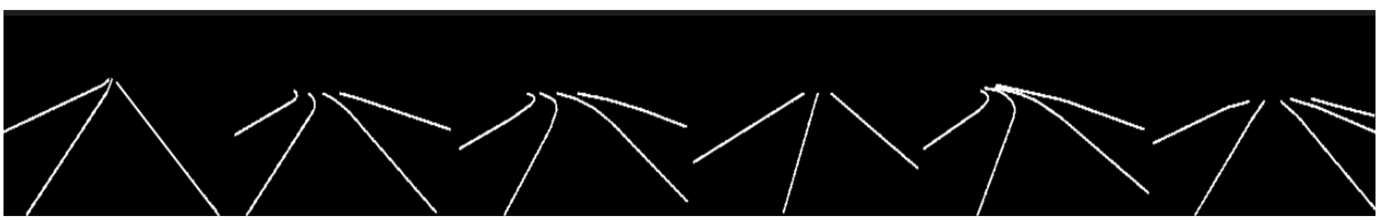

We tried implementing the Conv-LSTM model but weren't able to achieve good results due to low epochs and various errors and hence have not included the code implementation.

## Analysis of the solution with discussion on the possible weaknesses

SInce it is a Computer vision task change in illumination and camera can affect the prediction accuracy of the model. As in Low illumination condition model cant preform good. Heavy model size and low inference Speed as Car cant have heavy commutation power and plus drives fast hence low inference speed of model can affect the maneuverability of the vehicles. A work around of this can we Model compression with tricks like Quantization to reduce model size and well and increasing the inference speeds.

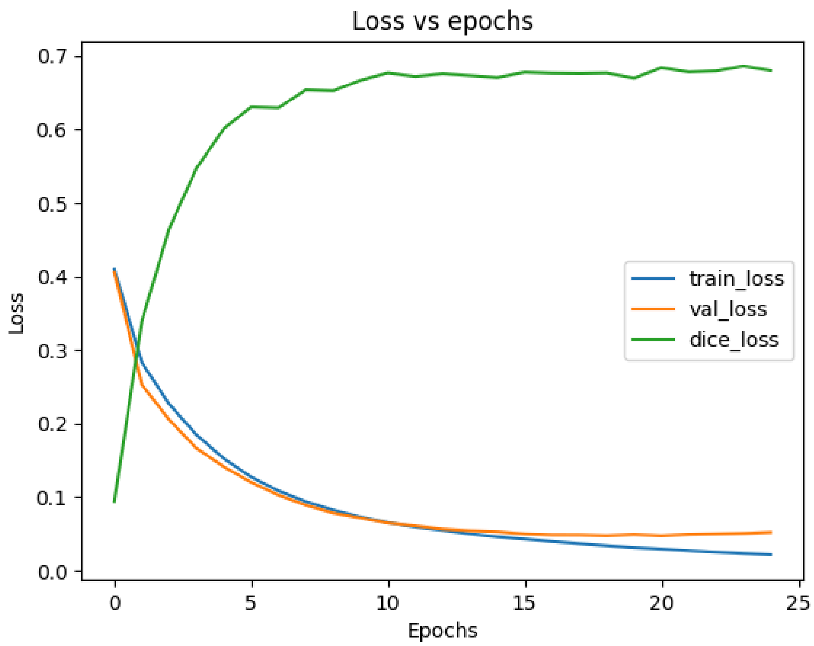
## Results

Simple Unet with Binary Mask. Initially it was giving random points all over the image but after applying the optimization



GROUND TRUTH

Prediction Masks




The Model was Ran for 25 Epochs in total where the testing metrics are provided Below.

**Testing data metrics**

**Accuracy:** 0.04740147062467676

**Dice score:** 0.6923

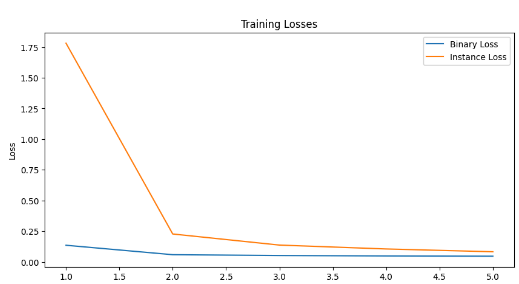
**Inference speed :** **`2.6962852478027344 seconds per frame`**

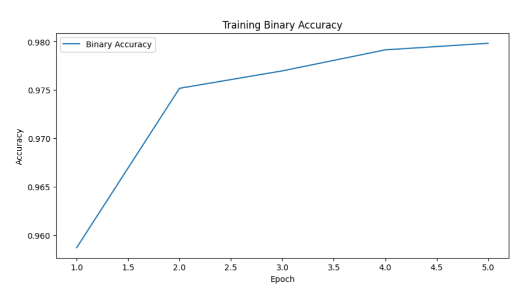
**`Total size of the model: 29.61 MB`**

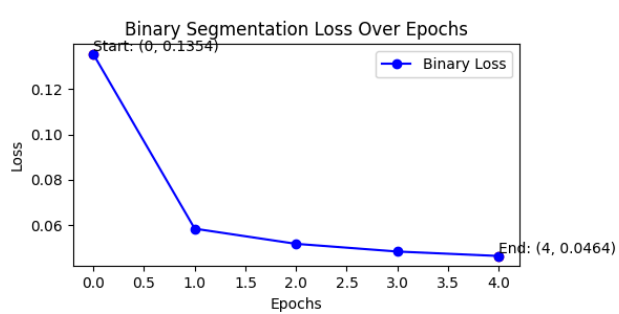
As we can See that the prediction boundary are well aligned and Dice Score Could be Increased by increasing the width of the the boundary of the mask

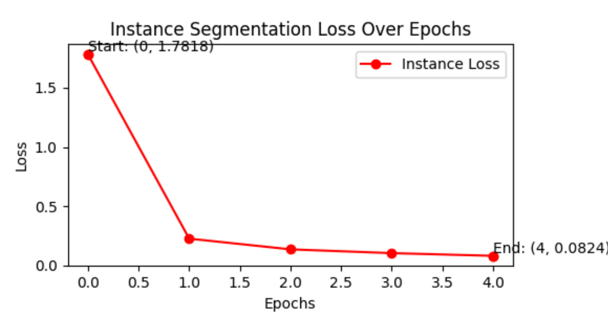
Next we have done the official paper implementation of ENet lane segmentation as its used for faster inferences and have compared the result with our Unet

Enet was ran for 5 epochs:

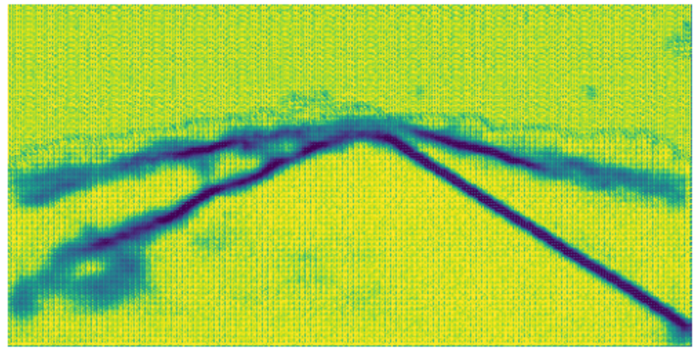


Original image



Predicted mask

On validation we got a Binary accuracy of 97.98 percentile which is comparable to the Unet But the lane line prediction is not even in the Given implementaion.

```
Epoch 4: Binary Loss = 0.0464, Instance Loss = 0.0824, Binary Accuracy = 0.9798
```

**Conclusion:** Deep learning is revolutionizing autonomous vehicle driving, particularly in tasks like lane detection and navigation but we can conclude that if increase the inference speed
(time to calculate one frame) by increasing the inference speed our robustness will increase. By this we can conclude that Deep learning based approaches are robust and adaptable. If we add more sensors like lidar and low light camera prediction accuracy can be increased and better models can be formed by introducing new methods of attention. As for Our model we stand at 18th position in the benchmark. Introduction of Custom loss function can alleviate the prediction accuracy.