Assignment - 1
CSE344 : Computer Vision
(February 2024)

Aakarsh Jain
2021507

Feb 2024

1 Theory

1.1

(a) MSE is not necessarily a convex function over discrete inputs, which is the case here. Thus, we might not be able to optimise the error while using MSE.

(b)
$$-\frac{1}{N}\Sigma y \log \hat{y} + (1-y)\log(1-\hat{y})$$

(c)
$$-\frac{1}{N}\Sigma y \log \hat{y} + (1-y)\log(1-\hat{y})$$
$$= -\frac{1}{1}\Sigma 0 \log 0.9 + (1-0)\log(1-0.9) = 2.3025$$

(d)
$$= -\frac{1}{3}(\log 0.1 + \log 0.8 + \log 0.3) = 3.7297$$

(e) The rule for updating weights can be espressed as

$$W = W - \alpha(\nabla L_{BCE} + 2\lambda W)$$

Model A will have lower weights as compared to model B

1.2

(a) Dimensions of $W^{[2]}$ are $K \times D_a$ and Dimensions of $b^{[2]}$ are $K \times 1$

(b) $\frac{\partial \hat{y}_k}{\partial z_k^{[2]}} = \hat{y}_k(1 - \hat{y}_k)$

(c) $\frac{\partial \hat{y}_k}{\partial z_i^{[2]}} = -\hat{y}_i \hat{y}_k$

(d) We shall use the formula,

$$\frac{\partial L}{\partial z_i^{[2]}} = -\Sigma_{j=1}^{K} \frac{\partial y_j \log \hat{y}_j}{\partial \hat{z}_i^{[2]}}$$

Thus, for $i = k$,

$$= -\frac{\partial 1 \cdot \log \hat{y}_k}{\partial \hat{z}_i^{[2]}} = -\frac{1}{\hat{y}_k} \frac{\partial \hat{y}_k}{\partial \hat{z}_i^{[2]}} = \hat{y}_k - 1$$

And, for $i \neq k$,

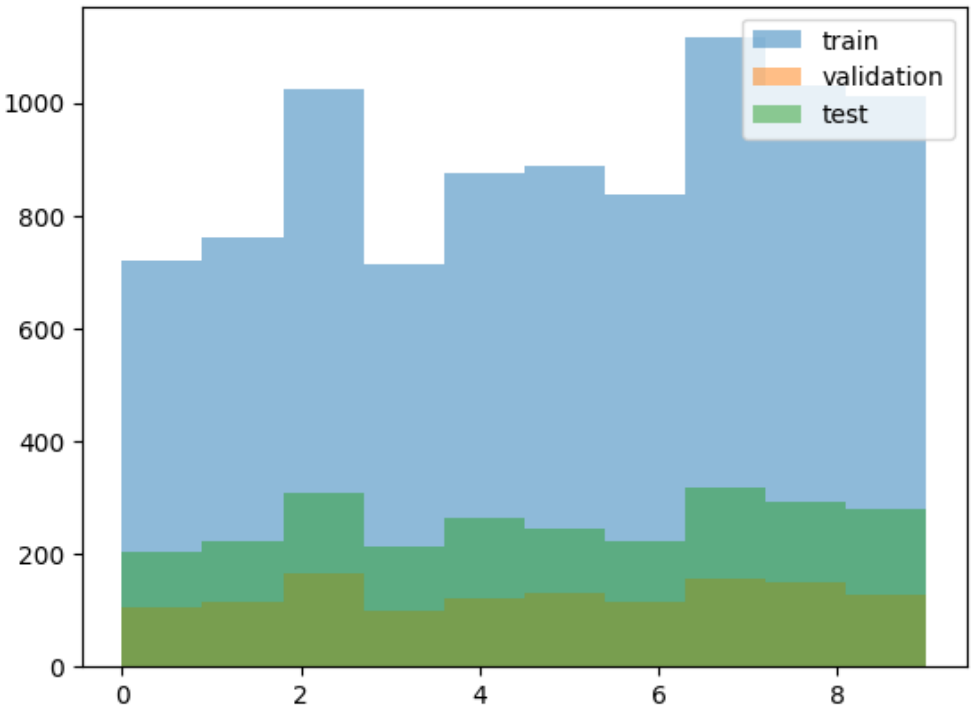
$$= -\frac{\partial 1 \cdot \log \hat{y}_k}{\partial \hat{z}_i^{[2]}} = -\frac{1}{\hat{y}_k} - \hat{y}_k \hat{y}_i = \hat{y}_i$$

(e) The problem stems from the fact that the output of softmax always stays betweem 0 and 1. Furthermore, Thus, even for very large numbers, the output is 1, but for small inputs, the value will diminish. We can resolve this issue by standardizing the outputs while subtracting the maximal value. This also ensures that the output values do not approach the extremeties.

2 Image Classification

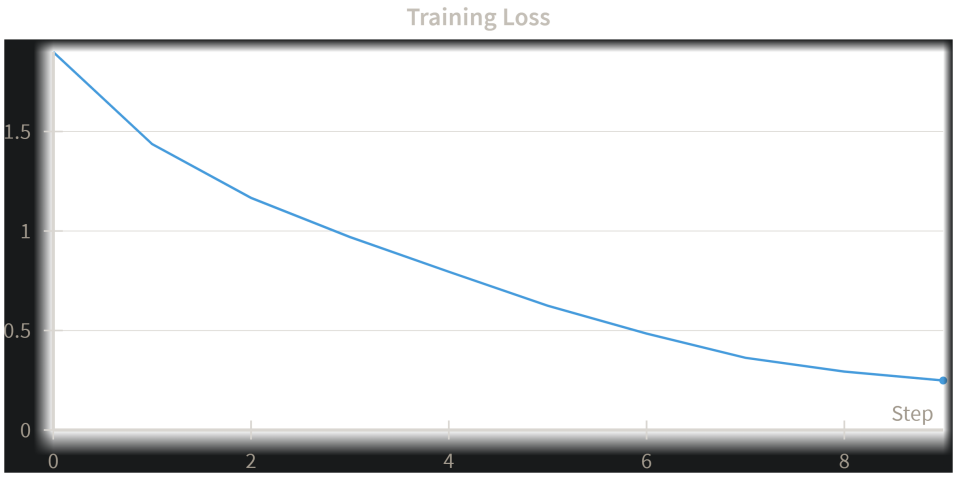
2.1 Downloading and Visualising the data

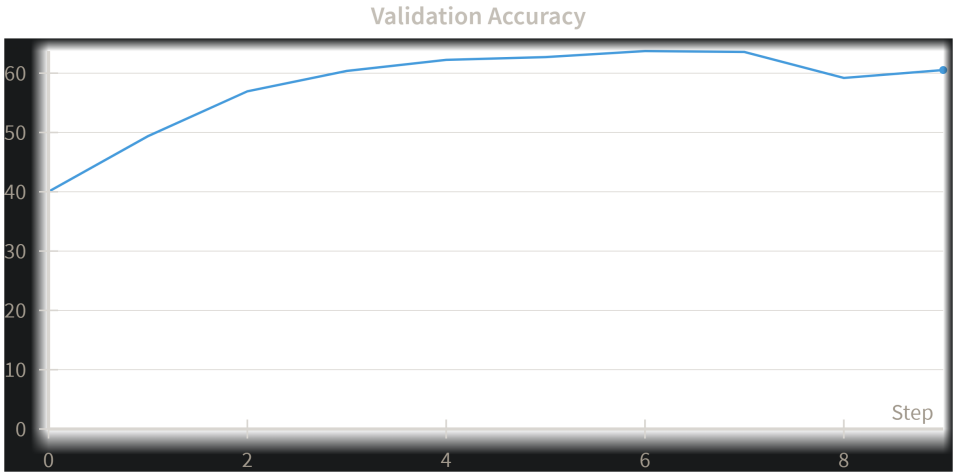
The dataloaders can be found in the submitted code, the distribution in the dataset is as follows -

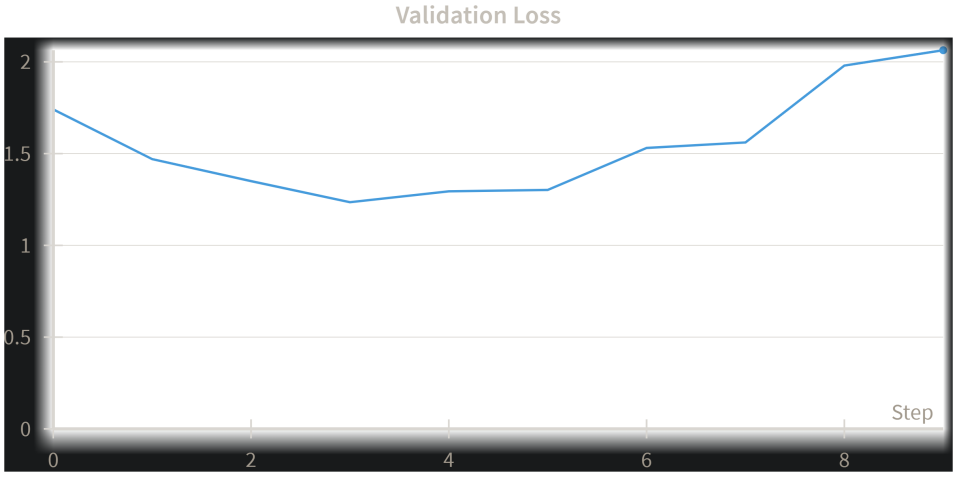


2.2 Training CNN from scratch

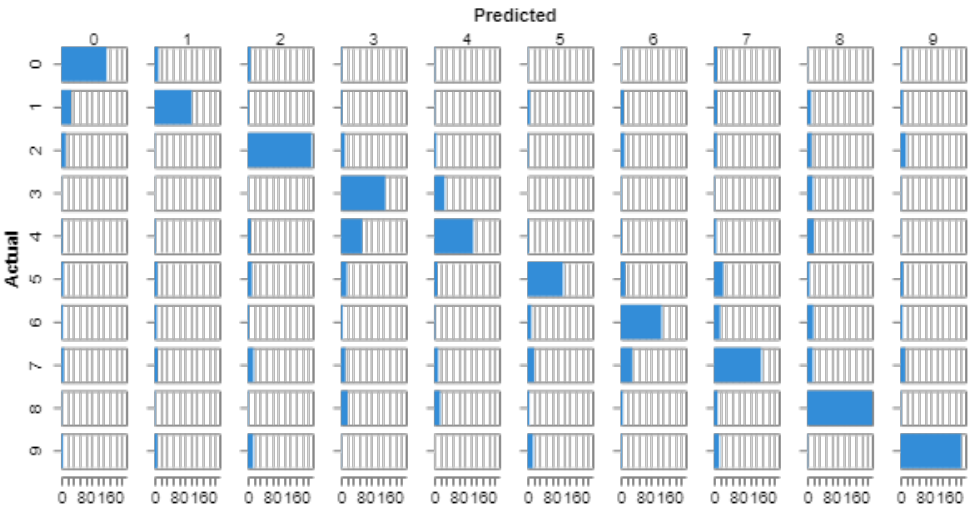
Following are the losses and accuracies for both training and validation -





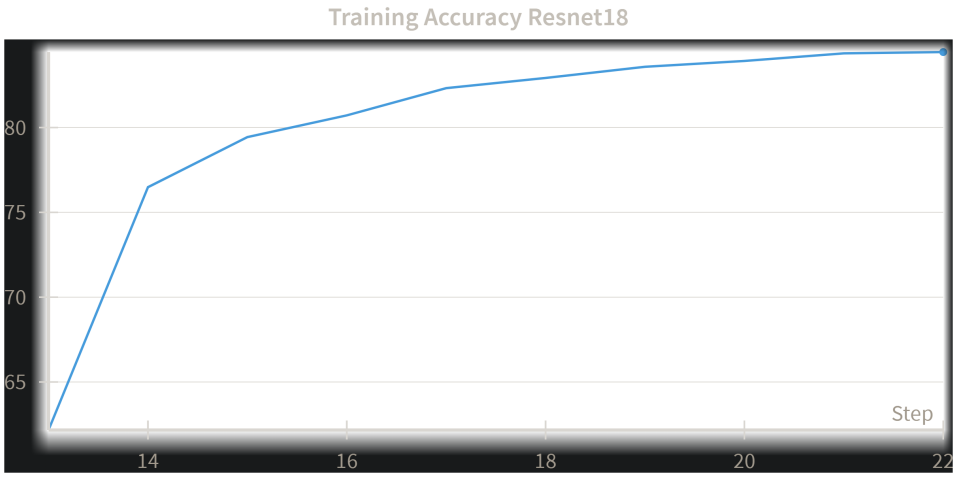


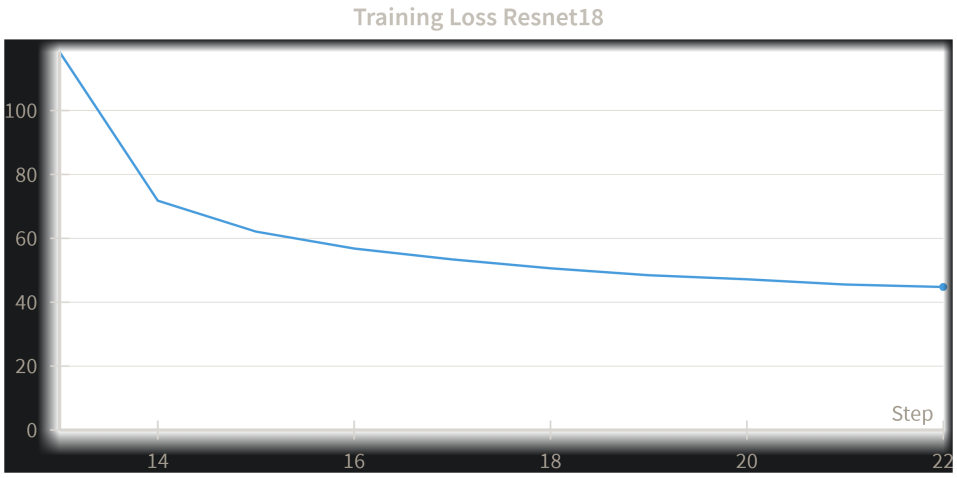
The model seems to be overfitting since the validation accuracy drops at 8 epoch.

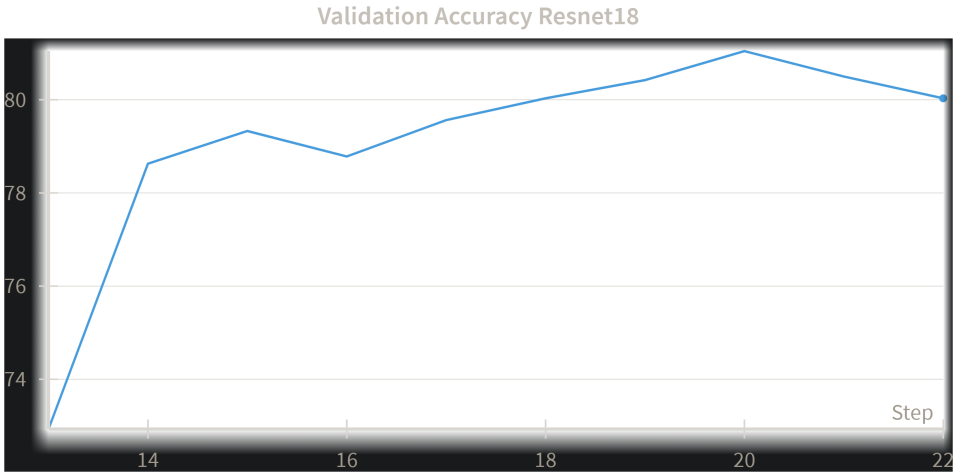


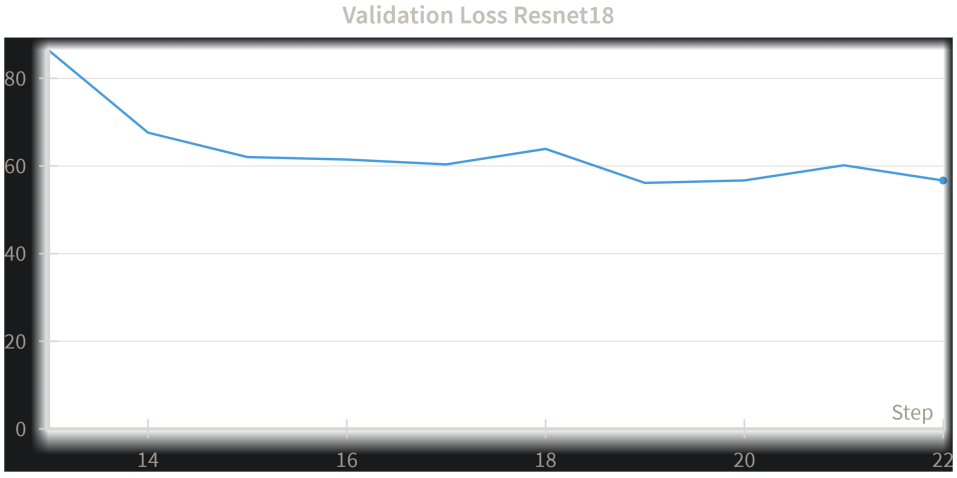
Testing Accuracy is 61.73 and the F1 score is 0.6152.

2.3 Fine-Tuning a pretrained model

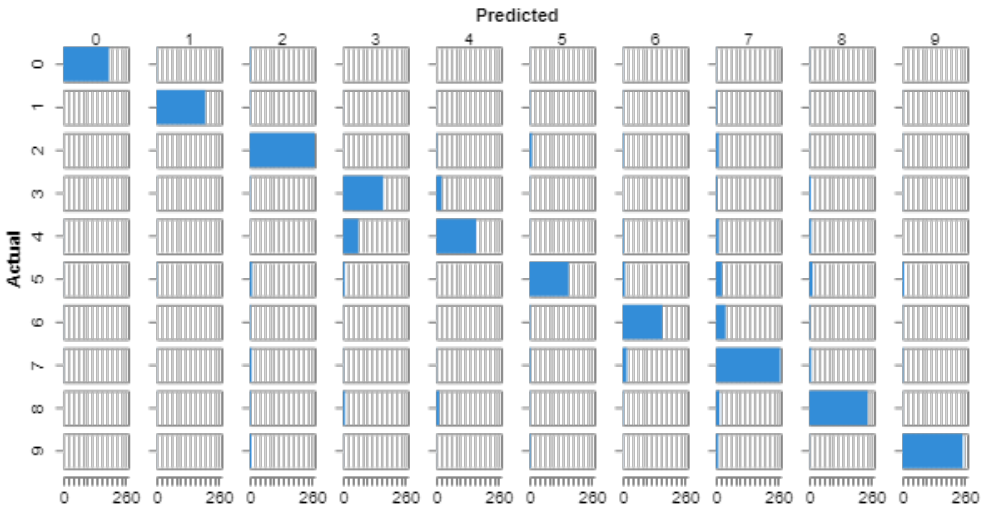






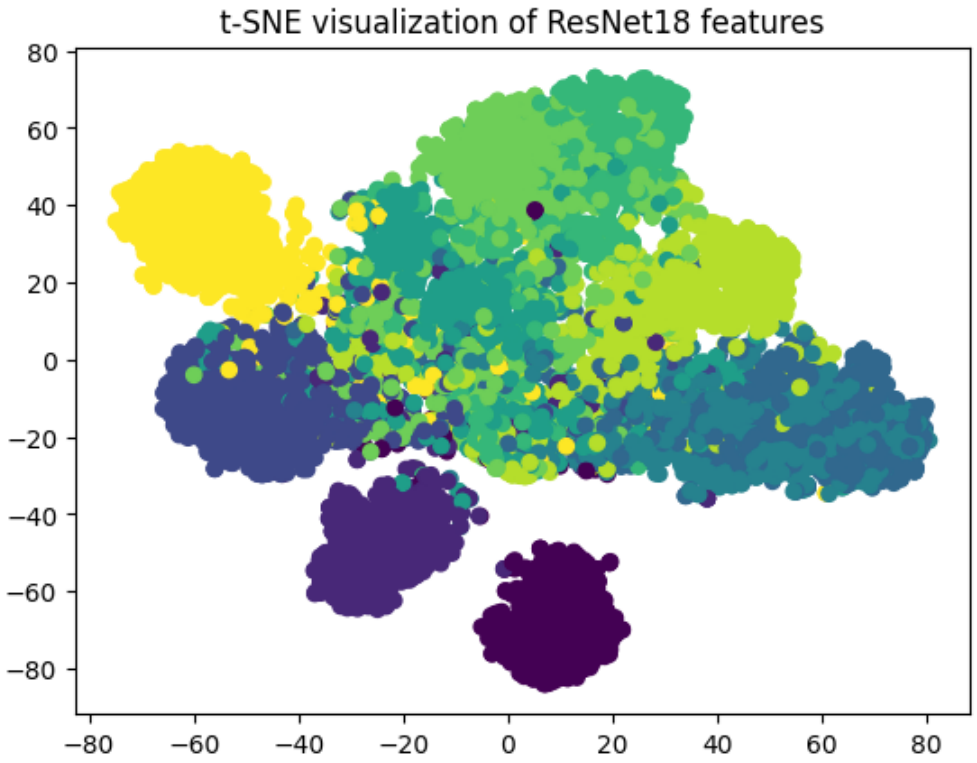


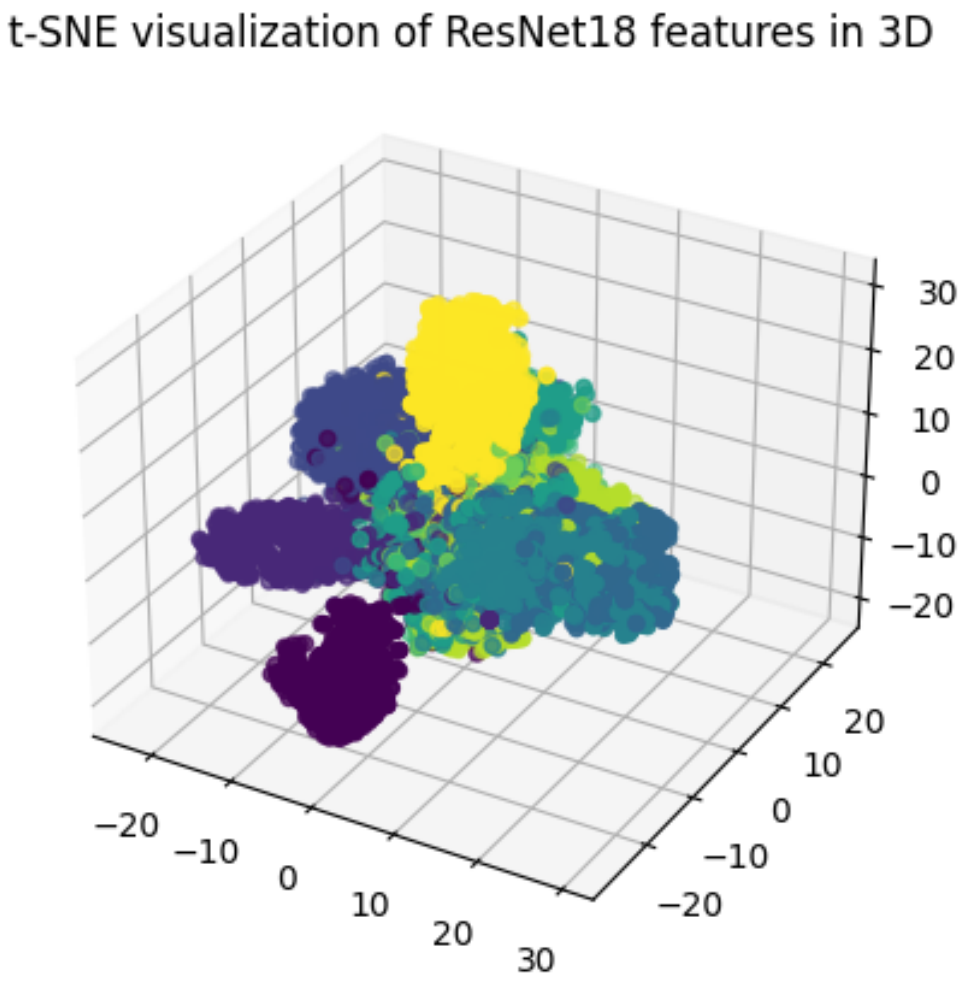
There is no overfitting as visible in the graph.



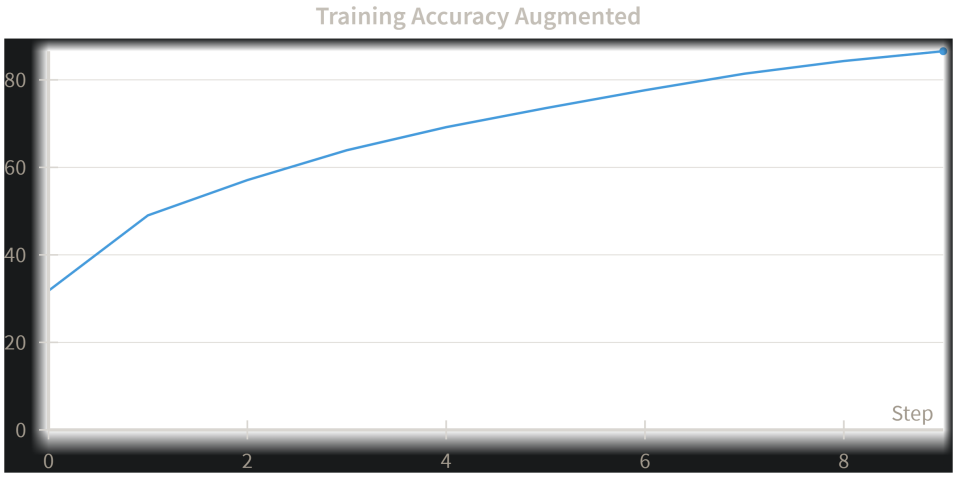
Testing Accuracy is 81.608 and the F1 score is 0.8164.

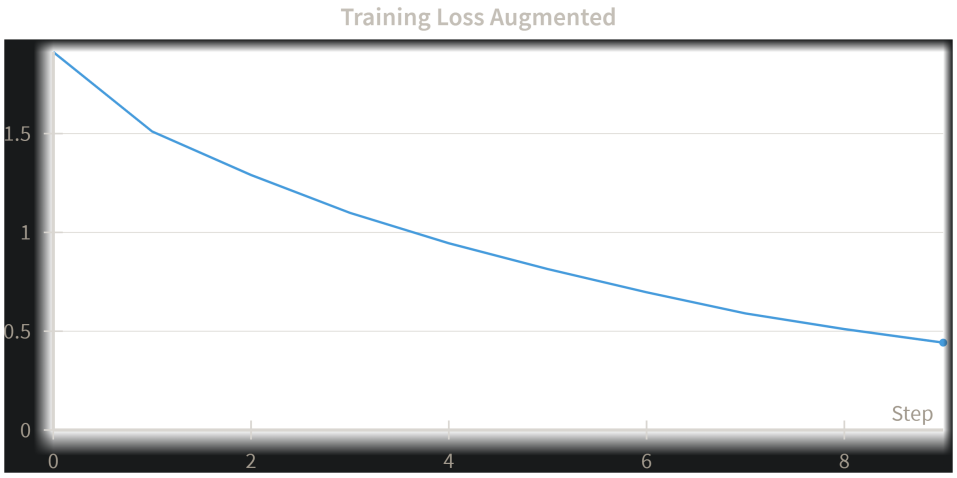
The t-SNE plots are as follows -

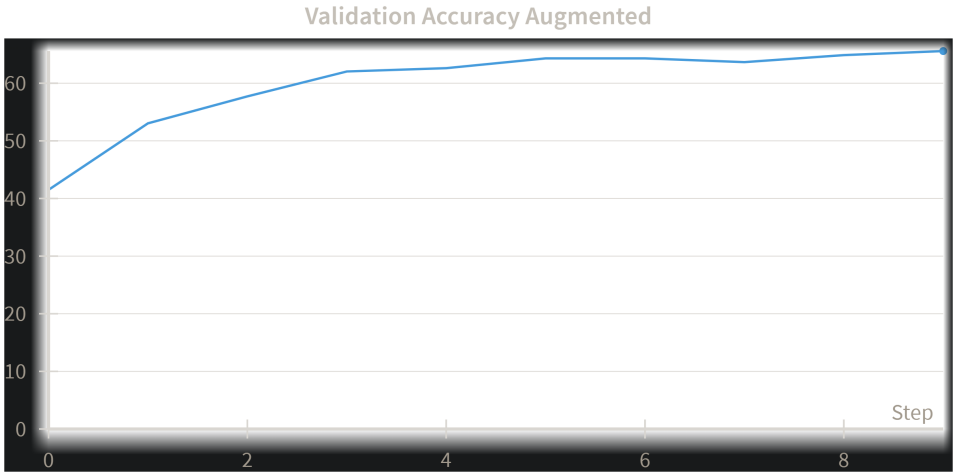


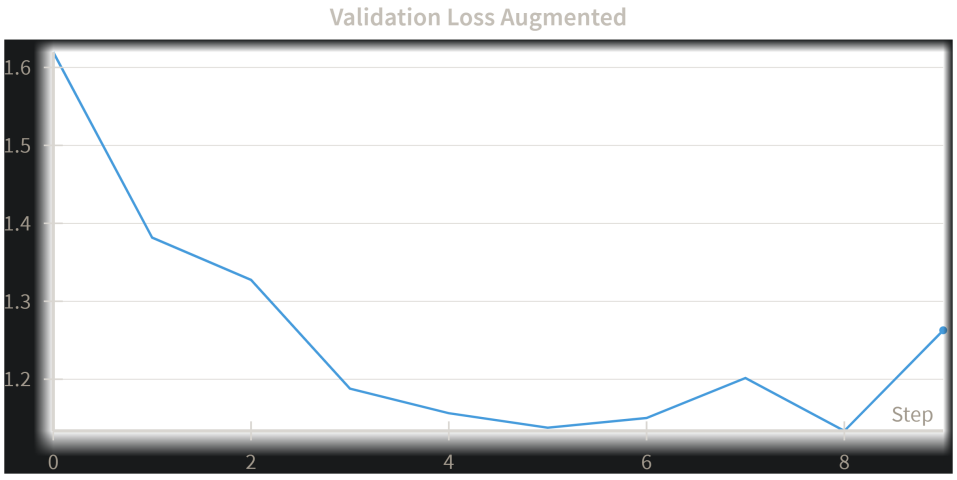


2.4 Data augmentation techniques

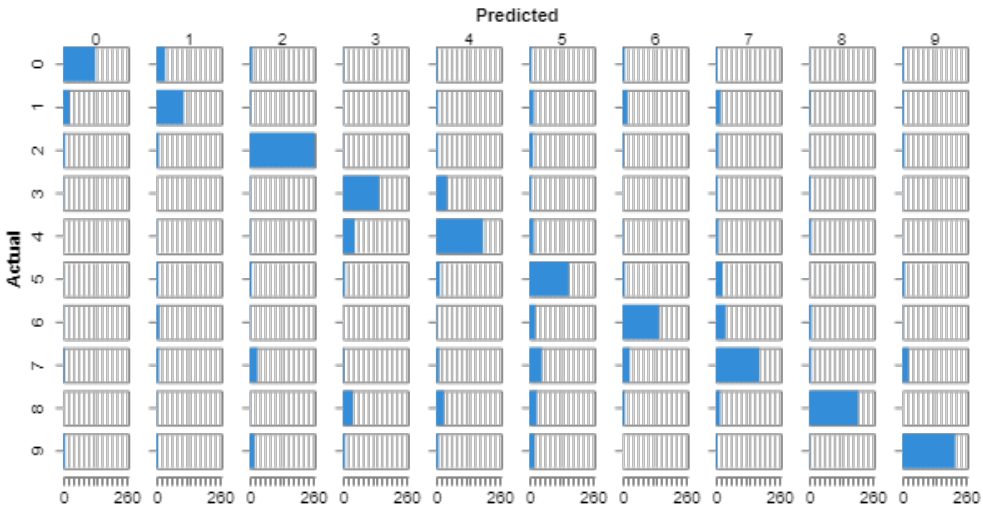








There is overfitting at the 8 epoch, when the validation loss increases suddenly.



Testing Accuracy is 62.513 and the F1 score is 0.6273.

2.5 Final Comparison

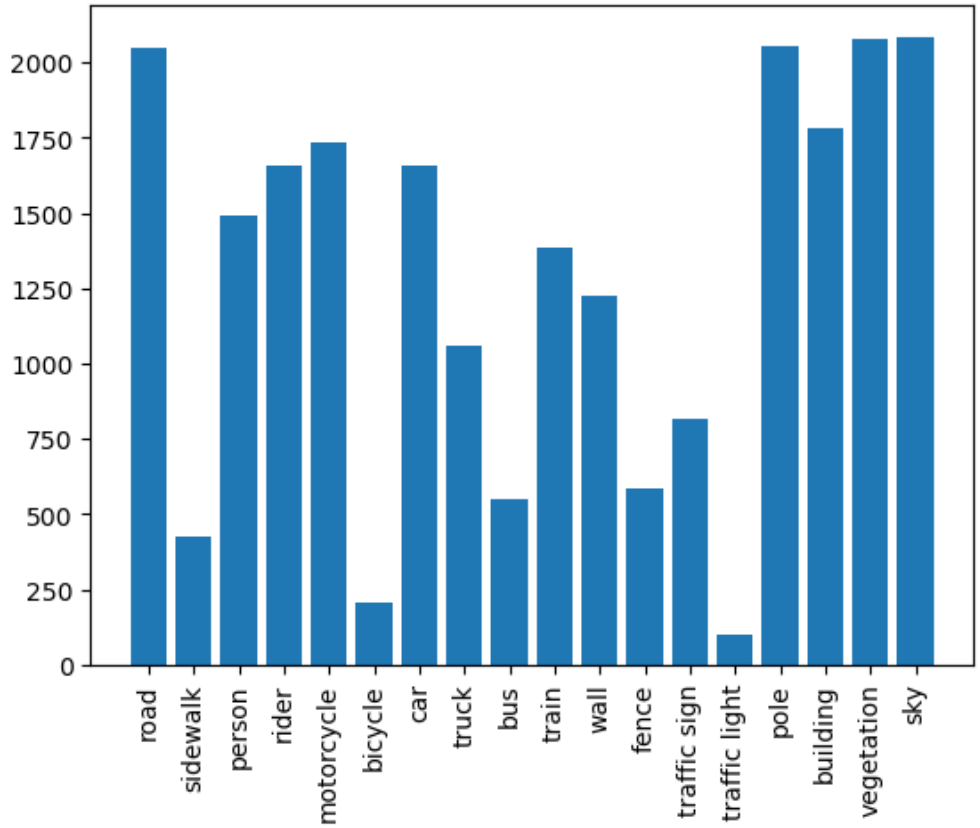
The pretrained Resnet model is clearly better by 20% margin, while the augemented CNN model is marginally better than CNN model trained in the first part.

Resnet model has been trained on other datasets, and then further fine-tuned for our dataset. Thus, we can expect its accuracy to be best among all.

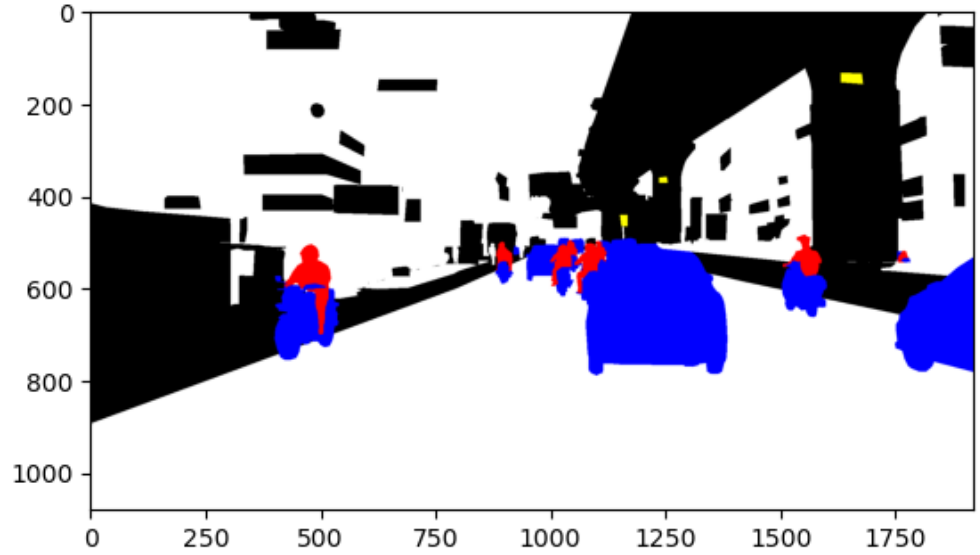
Further, in the augmented model, we add linearly transformed images back into the dataset to increase the diversity in the dataset. This results in marginally improved results.

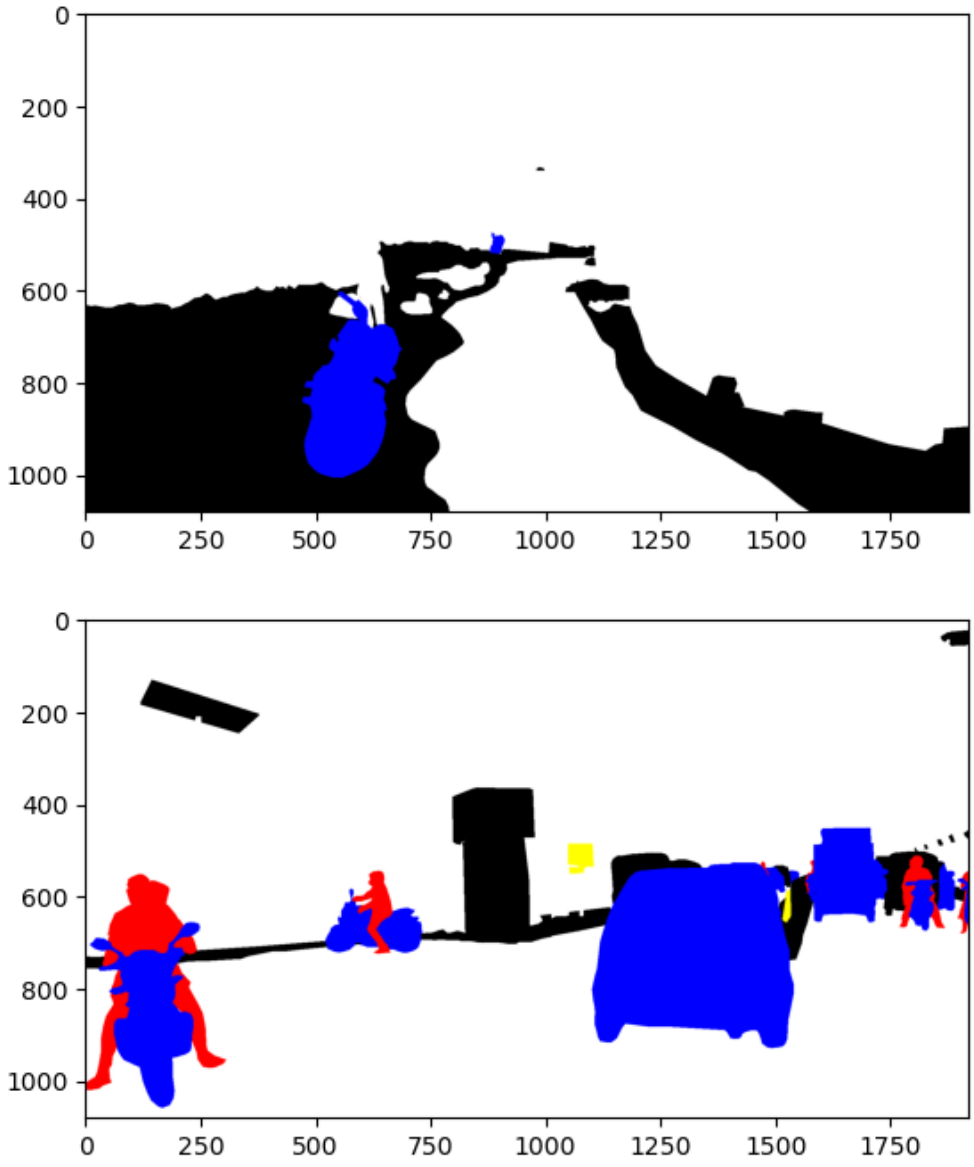
3 Image Segmentation

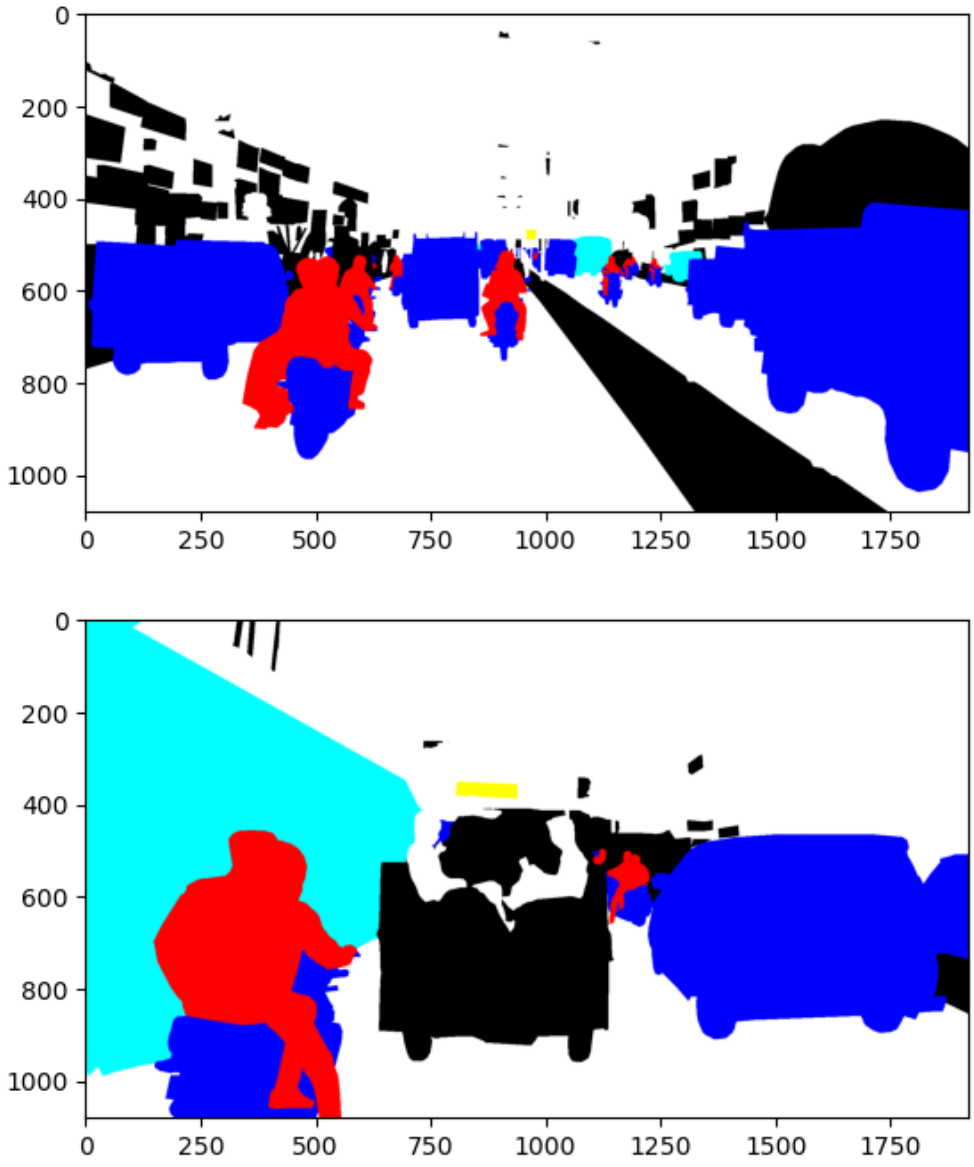
3.1 Downloading and Visualising the data



The colour coded images for each class is available in the submitted .ipynb file, here are 5 example images for reference -

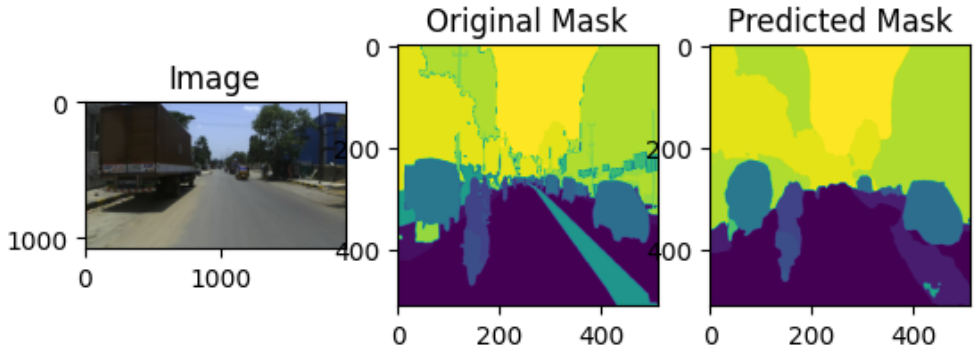



0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750


0
200
400
600
800
1000
0
250
500
750
1000
1250
1500
1750

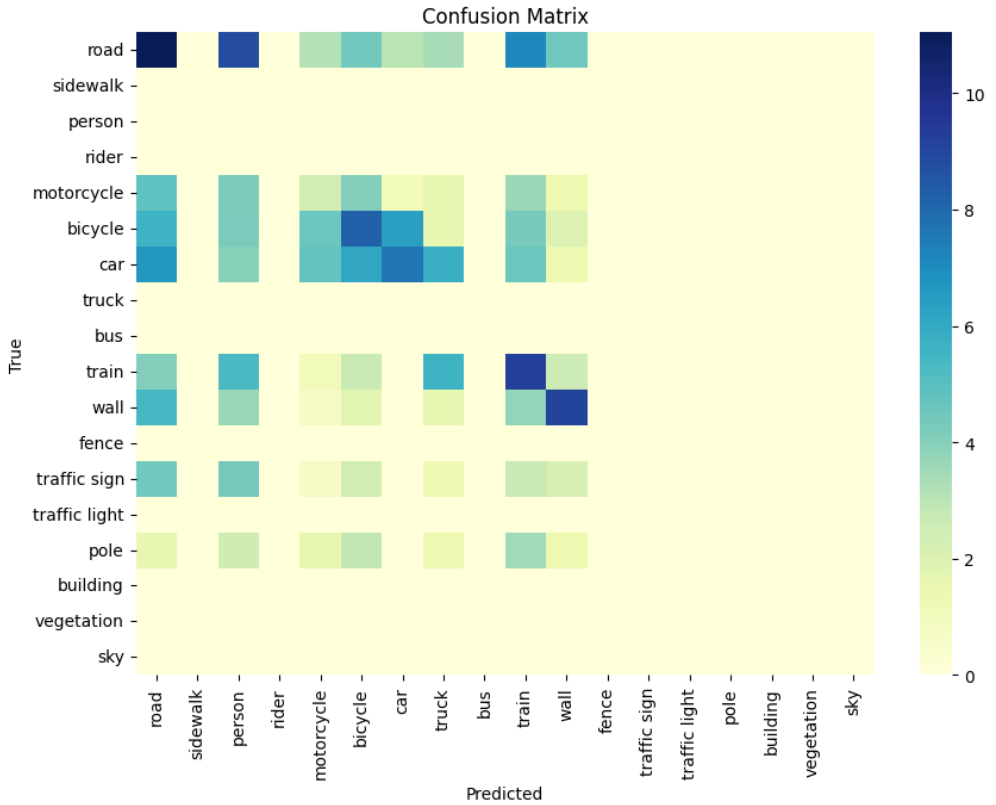
3.2 Evaluating the segment model

Metrics on different values of α						
Object v/s Perf	Pixel	dice	IoU	precision	recall	F1
road	0.97	0.87	0.77	0.98	1.00	0.99
sidewalk	0.61	0.00	0.00	0.22	0.97	0.35
person	0.52	0.03	0.02	0.83	0.82	0.82
rider	0.24	0.74	0.59	0.96	0.53	0.69
motorcycle	0.22	0.60	0.43	0.97	0.36	0.52
bicycle	0.45	0.00	0.00	0.13	0.90	0.23
car	0.89	0.88	0.79	0.88	0.90	0.89
truck	0.21	0.84	0.73	0.80	0.28	0.42
bus	0.52	0.00	0.00	0.57	0.36	0.44
train	0.00	0.00	0.00	0.80	0.11	0.19
wall	0.22	0.00	0.00	0.62	0.56	0.59
fence	0.35	0.00	0.00	0.30	0.88	0.45
traffic sign	0.26	0.00	0.00	0.44	0.70	0.55
traffic light	0.01	0.00	0.00	0.08	0.06	0.07
pole	0.21	0.00	0.00	0.99	0.90	0.94
building	0.87	0.82	0.69	0.87	0.98	0.92
vegetation	0.90	0.89	0.80	1.00	0.97	0.98
sky	0.89	0.97	0.93	1.00	0.97	0.98



The above image reported low IoU score in 9 classes - sidewalk, person, motorcycle, bicycle, train, wall, fence, traffic sign and pole. This is likely due to less amount of examples in the dataset.

3.3 Analysis



The precision, recall and F1 score is already present in the table in the above section. The model clearly is good at identifying classes like road, person, car, pole, building, vegetation and sky. On the other hand, the performance on classes like traffic light and train is abyssmally low.

This can be probably explained using the number of times these objects appear in the dataset itself, thus making an argument for better dataset.