CV HW1

QUESTION 1 PART 1:

a. Since we are using a classifier or dealing with a classification type problem, MSE is not a good choice for loss function. MSE minimized the difference between predicted y values and the actual y values. MSE is better when our desired output values are continuous. For discrete classification problems, we are looking to fit a hyperplane or multiple hyperplanes across our data which MSE cannot capture.

Date / /

Q1) b) Papapa is sweet label = 1
Papaya not sweet label = 0

$$CE = -\sum_{i}^{c} t_i \log(f(s)_i)$$

for 1 sample:

$$CE = -[y \log(\hat{y}) + (1-y)\log(1-\hat{y})]$$

c) CE for $\hat{y} = 0.9$
$y = 0$

$$CE = -[0 \cdot \log(0.9) + 1 \cdot \log(1-0.9)]$$
$$= -\log_e(0.1)$$
$$= -(-2.3)$$
$$= 2.3$$

d) $\{y_1, y_2, y_3\} = \{1, 0, 0\} = Y$

$\hat{Y} = \{\hat{y_1}, \hat{y_2}, \hat{y_3}\} = \{0.1, 0.2, 0.7\}$

CE for $y_1 = -[1 \cdot \log_e(0.1)] = 2.3025$

CE for $y_2 = -[1 \cdot \log_e(0.8)] = 0.22$

CE for $y_3 = -[1 \cdot \log_e(0.3)] = 1.203$

Total loss $= 2.30 + 0.22 + 1.203 = 3.723$

Avg total BCE $= 3.723 / 3 = 1.241$

(e) L2 regularization

W - gradient descent

α, λ, L_{BCE} M - no of samples

In L2 regularization

$$\mathcal{L}(w,b) = \frac{1}{M} \sum_{i=1}^{M} L_{BCE}(y, \hat{y}) + \frac{\lambda}{2M} \sum_{l=1}^{L} \|W^{l}\|^2$$

Update in gradient descent

$$W^{[l]} = W^{[l]} - \alpha\, \partial W^{[l]}$$

$$= W^{[l]} - \alpha \frac{\partial \mathcal{L}}{\partial W^{[l]}}$$

$$= W^{[l]} \underbrace{- \frac{\alpha \lambda}{M} W^{[l]}}_{\text{from } L_2 \text{ regularisation term}} \underbrace{- \alpha \frac{\partial L_{BCE}}{\partial W}}_{\text{from backprop}}$$

f) KL Divergence measures diffence between two ~~divergence~~ distribution

$$D_{KL}(P||Q) = \Sigma_x P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

Cross Entropy measures bits needed to represent P using encoding scheme Q

$$CE(P, Q) = -E_{x \sim p} \log Q(x)$$

$$CE(P, Q) = CE(P) + D_{KL}(P||Q)$$

Proof:

$$CE(P, Q) = -\Sigma_x P(x) \log(Q(x))$$

$$= -\Sigma_x P(x) \log(Q(x)) + \Sigma_x P(x) \log\left(\frac{P(x)}{P(x)}\right)$$

$$= -\Sigma_x P(x) \log\left(\frac{Q(x)}{P(x)}\right) + \Sigma_x P(x) \log\left(\frac{1}{P(x)}\right)$$

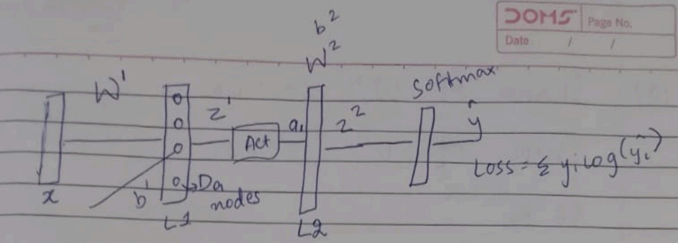
$$= -\Sigma_x P(x) \log\left(\frac{Q(x)}{P(x)}\right) + CE(P)$$

$$= D_{KL}(P||Q) + CE(P)$$

$$= CE(P, Q)$$

QUESTION 1 PART 2:

Q2.)



$$z^1 = W^1 x + b^1$$

$$a^1 = \text{Leaky ReLU}(z^1, \alpha = 0.01)$$

$$z^2 = W^2 a^1 + b^2$$

$$\hat{y} = \text{softmax}(z^2)$$

$$L = -\Sigma\, y_i \log(\hat{y}_i)$$

$$\underset{D_x \times 1}{x} \qquad y \in \{0,1\}^K$$

$$\underset{D_a \times 1}{z^1} = \underset{D_a \times D_x}{W^1}\ \underset{D_x \times 1}{x} + \underset{D_a \times 1}{b^1}$$

$$\underset{D_a \times 1}{a^1} = \text{Leaky ReLU}(\underset{D_a \times 1}{z^1}, \alpha = 0.01)$$

$$\underset{K \times 1}{z^2} = \underset{K \times D_a}{W^2}\ \underset{D_a \times 1}{a^1} + \underset{K \times 1}{b^2}$$

(a) Shape of W^2 $K \times D_a$

Shape of b^2 $K \times 1$

$X \in \mathbb{R}^{D_x \times m}$

Shape of output of hidden layer: $D_a \times m$

(b) $\frac{\partial \hat{y}_k}{\partial z_k^2}$

$$\text{Softmax}(y)_i = \frac{e^{y_i}}{\sum_{j=1}^{K} e^{y_j}}$$

$$\hat{y} \; \cancel{S(\hat{y})} = \frac{e^{y_i}}{\sum_{j=1}^{K} e^{y_j}}$$

$$\frac{\partial \hat{y}_k}{\partial z_k^2} = S' = \frac{e^{z_k^2} \sum e^{z_j^2} - e^{z_k^2} e^{z_k^2}}{\left(\sum e^{z_j^2}\right)^2}$$

$$= \cancel{S(\hat{y}_k) \times (1 - S(\hat{y}_k))}$$

$$= \hat{y}_k \times (1 - \hat{y}_k)$$

c) $\dfrac{\partial \hat{y}_k}{\partial z_i^2}$ for $i \neq k$ $\Rightarrow \dfrac{\partial \hat{y}_k}{\partial z_i^2} = \dfrac{e^{z_k^2}}{\sum_{j=1}^{k} e^{z_j^2}}$

$= \cancel{-(\hat{y}_k) \times s(\hat{y}_i)}$

$= -\hat{y}_k \times \hat{y}_i$

d) y $\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ 0 \end{bmatrix}$ ← k^{th} entry

$\dfrac{\partial L}{\partial z_i^2}$

Case 1: $i = k$

$\dfrac{\partial L}{\partial z_k^2}$

$L = -\sum y_i \log(\hat{y}_i) = -[y_1 \log(\hat{y}_1) + y_2 \log(\hat{y}_2) + \dots + y_n \log(\hat{y}_n)]$

$\dfrac{\partial L}{\partial \hat{y}_k} = -\left[y_k \cdot \dfrac{1}{\hat{y}_k} \right] = -\dfrac{y_k}{\hat{y}_k}$

$$\frac{\partial L}{\partial z_k^2} = \frac{\partial L}{\partial \hat{y}_k} \cdot \frac{\partial \hat{y}_k}{\partial z_k^2}$$

$$\hat{y}_k = s(z_k)$$

$$\frac{\partial L}{\partial z_k^2} = -\frac{y_k}{\hat{y}_k} \cdot \hat{y}_k \cdot (1 - \hat{y}_k)$$

$$= -y_k \cdot (1 - \hat{y}_k)$$

Case 2: $i \neq k$

$$\frac{\partial L}{\partial z_i^2} = -\frac{y_k}{\hat{y}_k} \cdot (-s(\hat{y}_k)) \cdot s(\hat{y_i})$$

$$= \frac{y_k}{\hat{y}_k} \cdot \hat{y}_k \cdot \hat{y}_i$$

$$= y_k \cdot \hat{y}_i$$

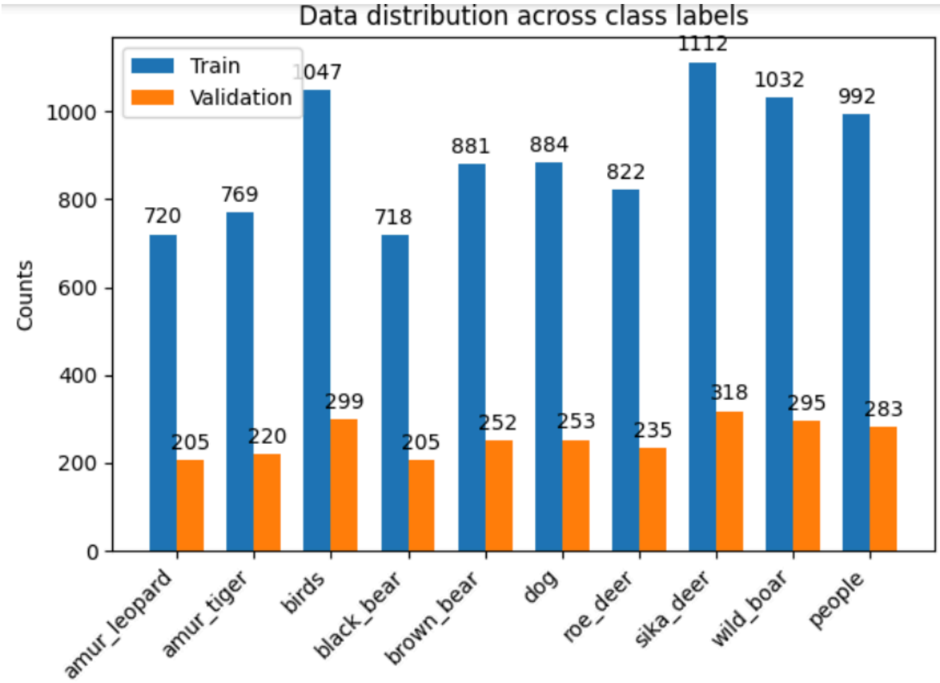
e)

e) The numerical stability problem in softmax function is due to the exponentiation of which can lead to very large values for not that large values of z_i.
This leads to precision problems of underflow or overflow when storing the values in memory.
This leads to inaccurate calculations due to floating point calculations.
Modified softmax implementation resolves this issue by subtracting the maximum of z from the exponent. This makes sure that the value of the numerator is always less than $e^0=1$
The modified equation:

$\text{softmax}(z_i) = e_i^{\,z} - \max / (\sum^{K}_{j=1} e_i^{\,z - \max})$

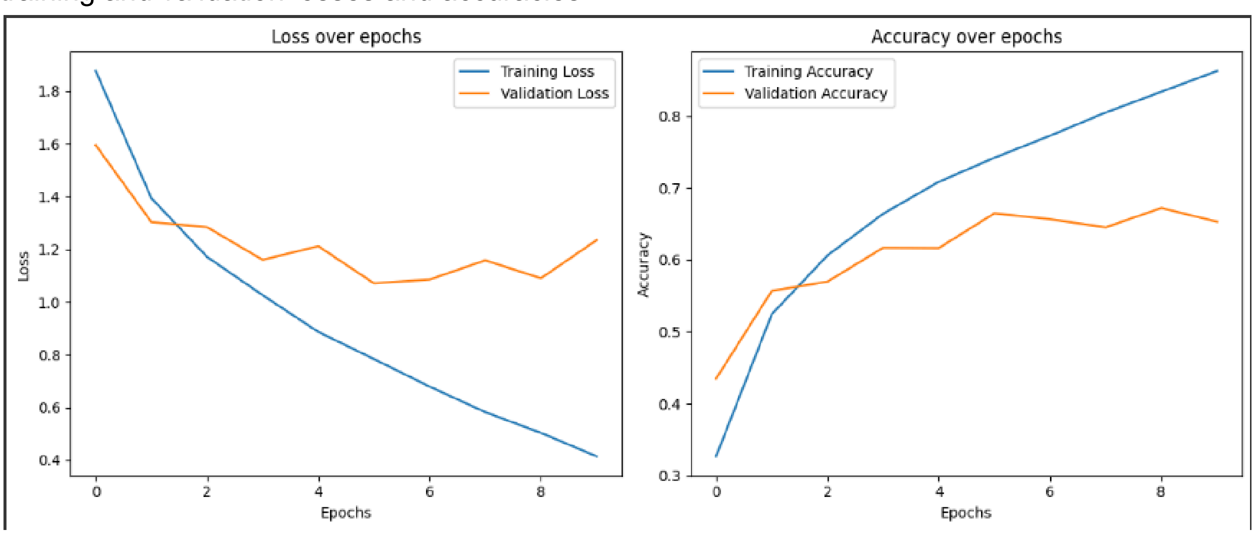
QUESTION 2:

1. Visualize the data distribution across class labels for training and validation sets.



CUSTOM CNN Architecture:

2. training and validation losses and accuracies.



3. Is the model overfitting?

   YES the model is overfitting as there is difference between the train and validation dataset. The model is performing well on training data but not that well on validation dataset.

4. Accuracy and F1-Score

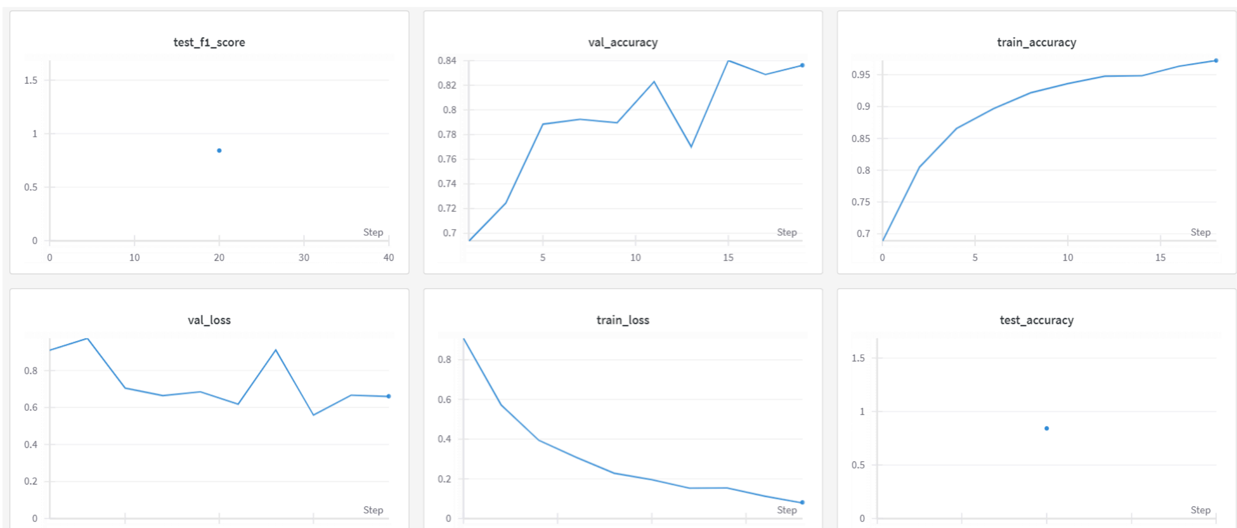
```
Accuracy: 0.6328916601714731
F1-Score: 0.6341451503964205
```

5. confusion matrix

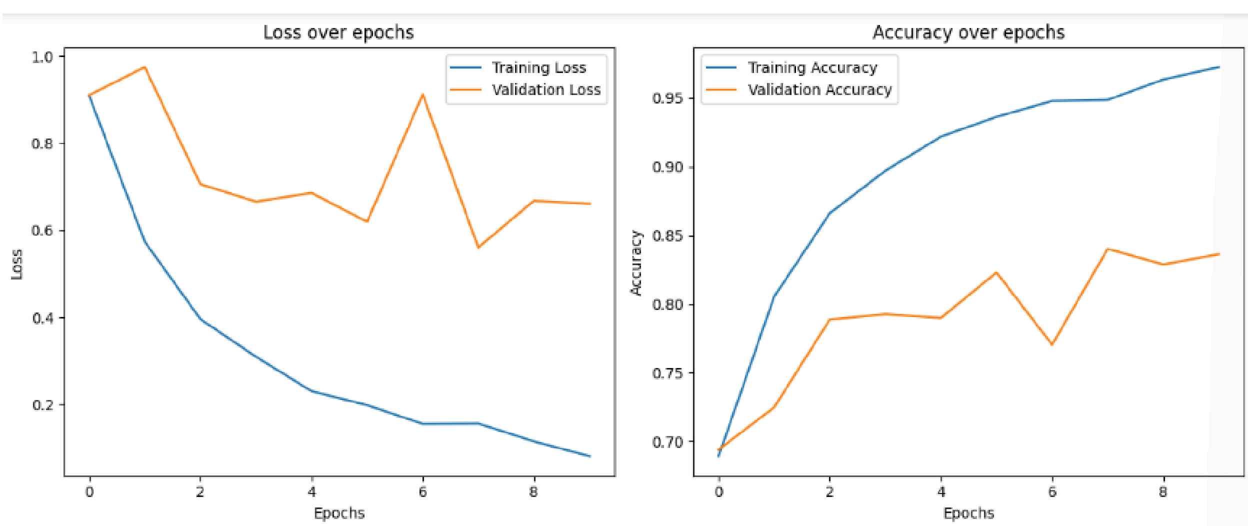
```
[[ 71  10   6   0   1   2   2   9   1   1]
 [ 15  59   7   0   2   6   4  11   3   3]
 [  2   1 110   3   4   3   2  15   6   4]
 [  5   2   0  51  13   4   2   5  20   0]
 [  7   0   0  18  75   4   3   6  13   0]
 [  4   3   8   1   7  68   6  22   6   1]
 [  1   4   1   0   1   8  69  24   8   1]
 [  2   5  12   1   4   7  14 104   9   1]
 [  2   5   4   2   8   1   6  11 108   1]
 [  6   1  18   1   1   6   2   9   1  97]]
```

FINE-TUNED RESENT 18:

6. training and validation losses and accuracies.



7.



8. Is the model overfitting?

   YES the model is overfitting as there is difference between the train and validation dataset. The model is performing well on training data but not that well on validation dataset.

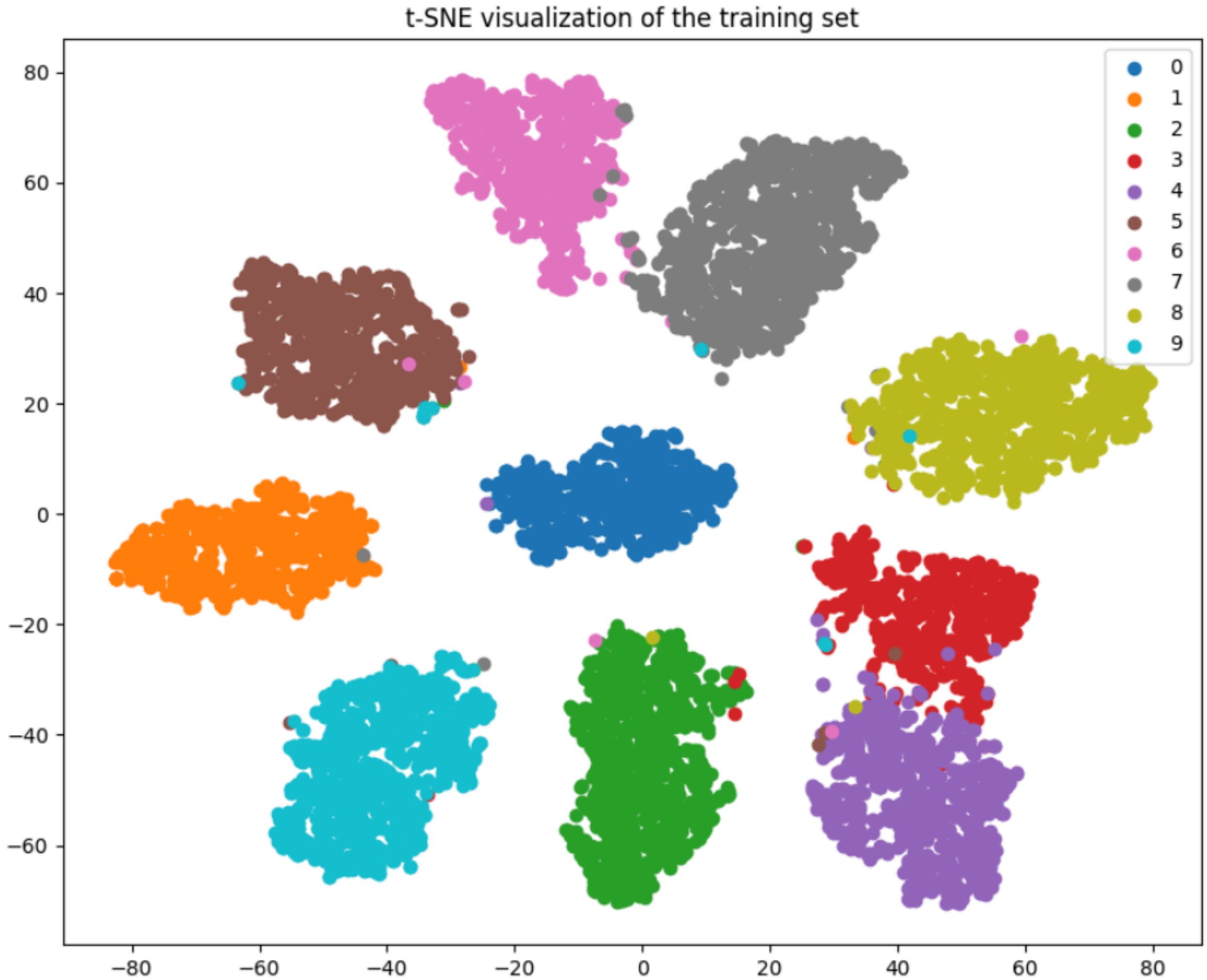
9. Accuracy and F1-Score

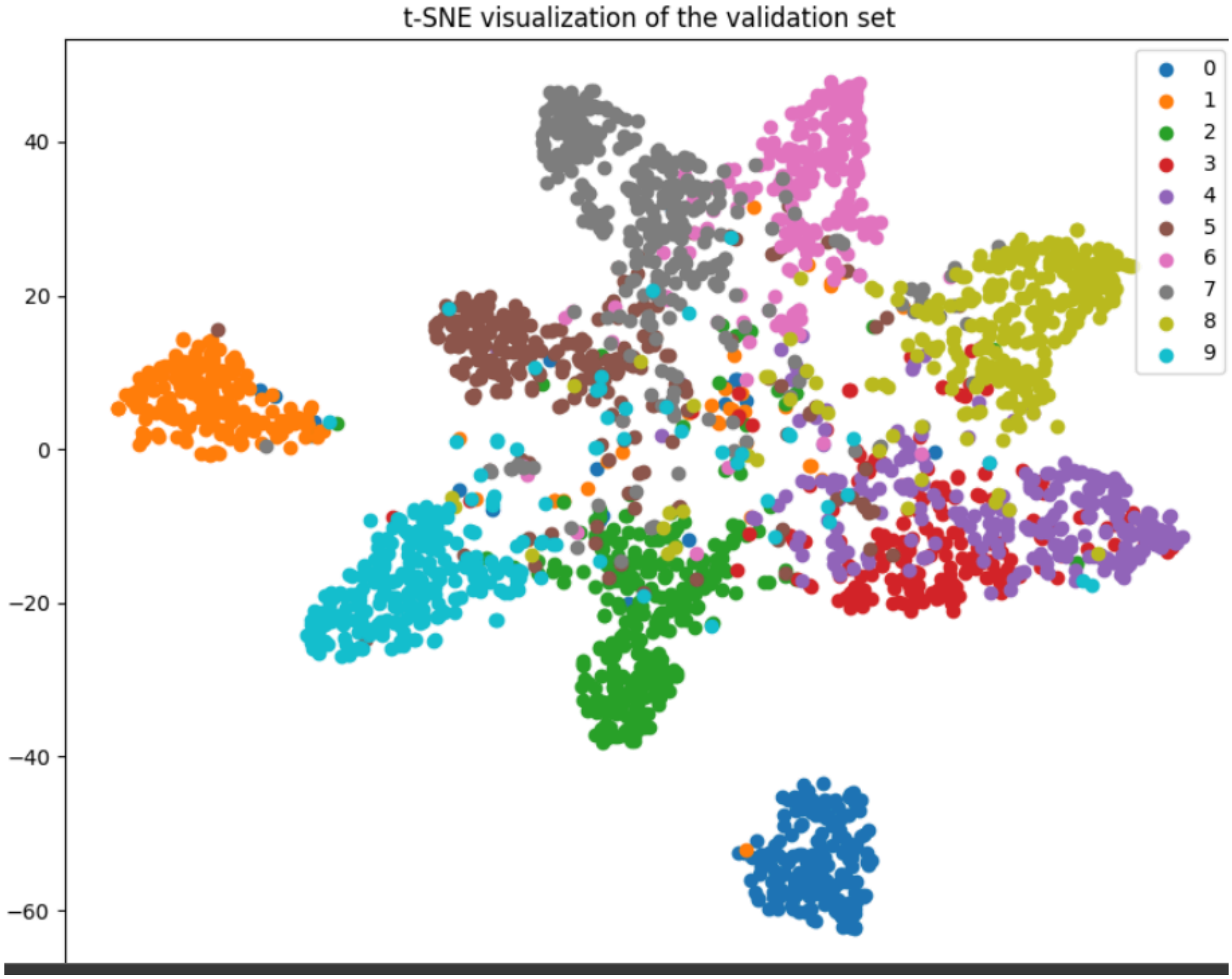
```
Accuracy: 0.8425565081839439
F1-Score: 0.8422363733672474
```

10. confusion matrix

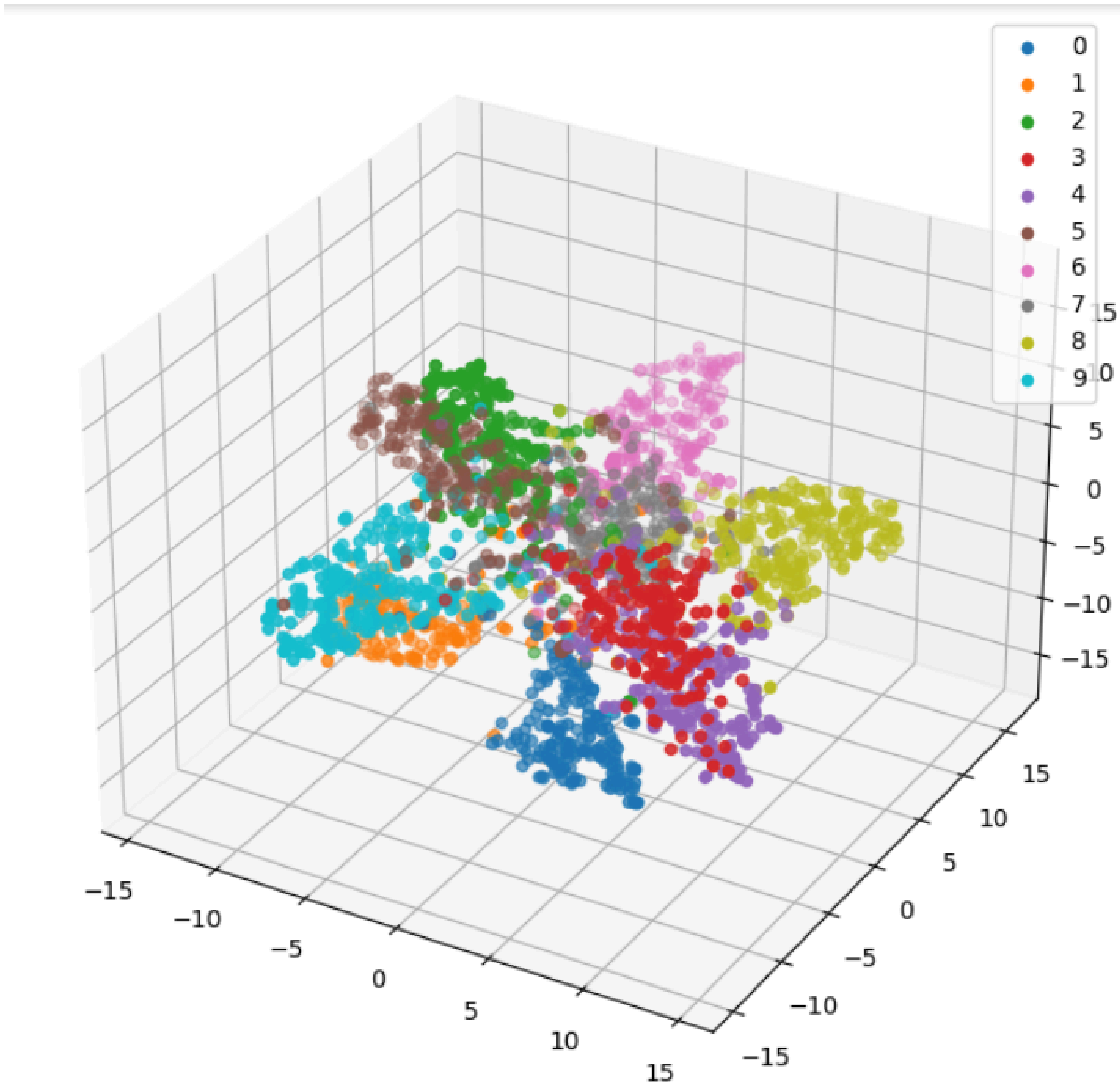
```
[[ 92   0   5   0   2   0   0   2   1   1]
 [  2  91   1   0   2   8   1   3   2   0]
 [  0   0 143   0   2   3   0   0   2   0]
 [  0   0   2  58  29   7   0   0   6   0]
 [  0   0   1   6 105   4   1   2   7   0]
 [  0   1   4   1   0 114   0   0   5   1]
 [  0   0   1   0   0   9  93   9   5   0]
 [  0   0   6   1   1   8   4 127  11   1]
 [  0   1   2   5   0   0   1   2 136   1]
 [  0   0   8   1   1   7   1   1   1 122]]
```

11. TSNE plots in 2D


t-SNE visualization of the training set
80
60
40
20
0
−20
−40
−60
−80
−60
−40
−20
0
20
40
60
80
0
1
2
3
4
5
6
7
8
9

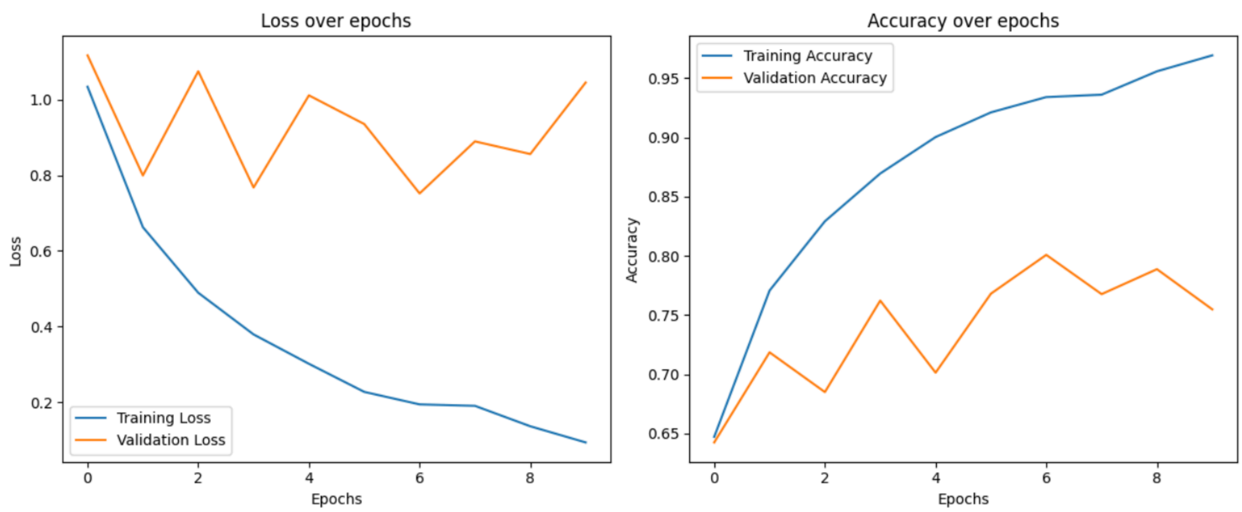


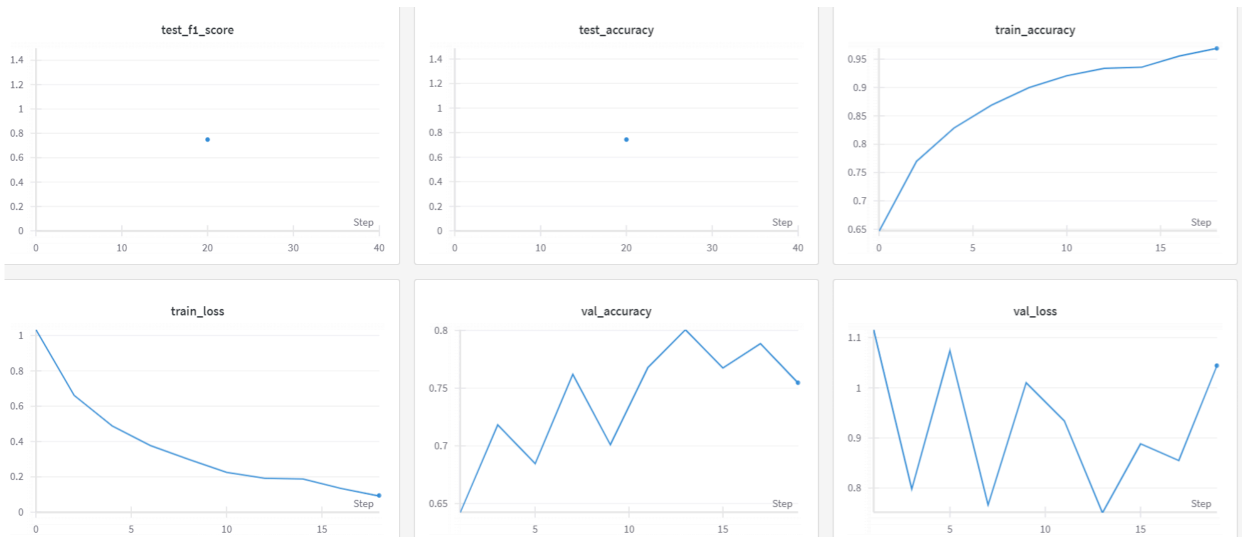
12. TSNE plot in 3D



DATA AUGMENTED RESENT 18:

13. training and validation losses and accuracies.





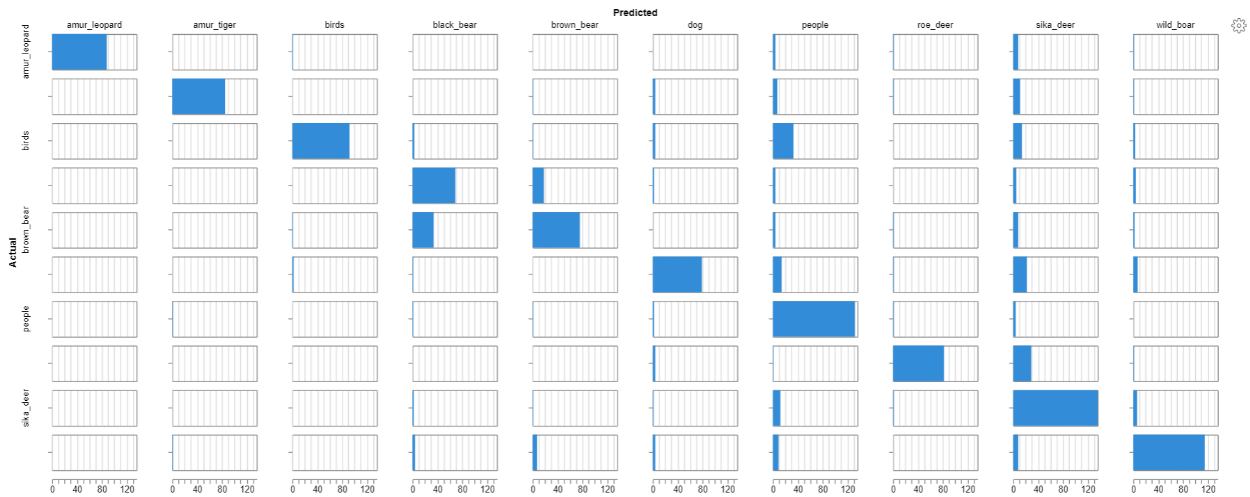
14. Is the model overfitting?

YES the model is still somewhat overfitting as there is difference between the train and validation dataset. The model is performing well on training data but not that well on validation dataset. But has reduced.

15. Accuracy and F1-Score

```
Accuracy: 0.7435697583787997
F1-Score: 0.7487109806541541
```

16. confusion matrix




```
[[ 88   0   1   0   0   0   1   8   1   4]
 [  0  85   0   0   1   4   1  11   1   7]
 [  0   0  92   3   1   4   0  14   3  33]
 [  0   0   0  69  18   2   0   5   4   4]
 [  0   0   1  34  76   0   1   8   2   4]
 [  0   0   2   1   0  79   1  22   7  14]
 [  0   0   0   0   0   4  82  29   1   1]
 [  0   0   0   2   1   1   1 136   6  12]
 [  0   1   0   4   7   4   0   8 115   9]
 [  0   1   0   1   1   2   1   4   0 132]]
```

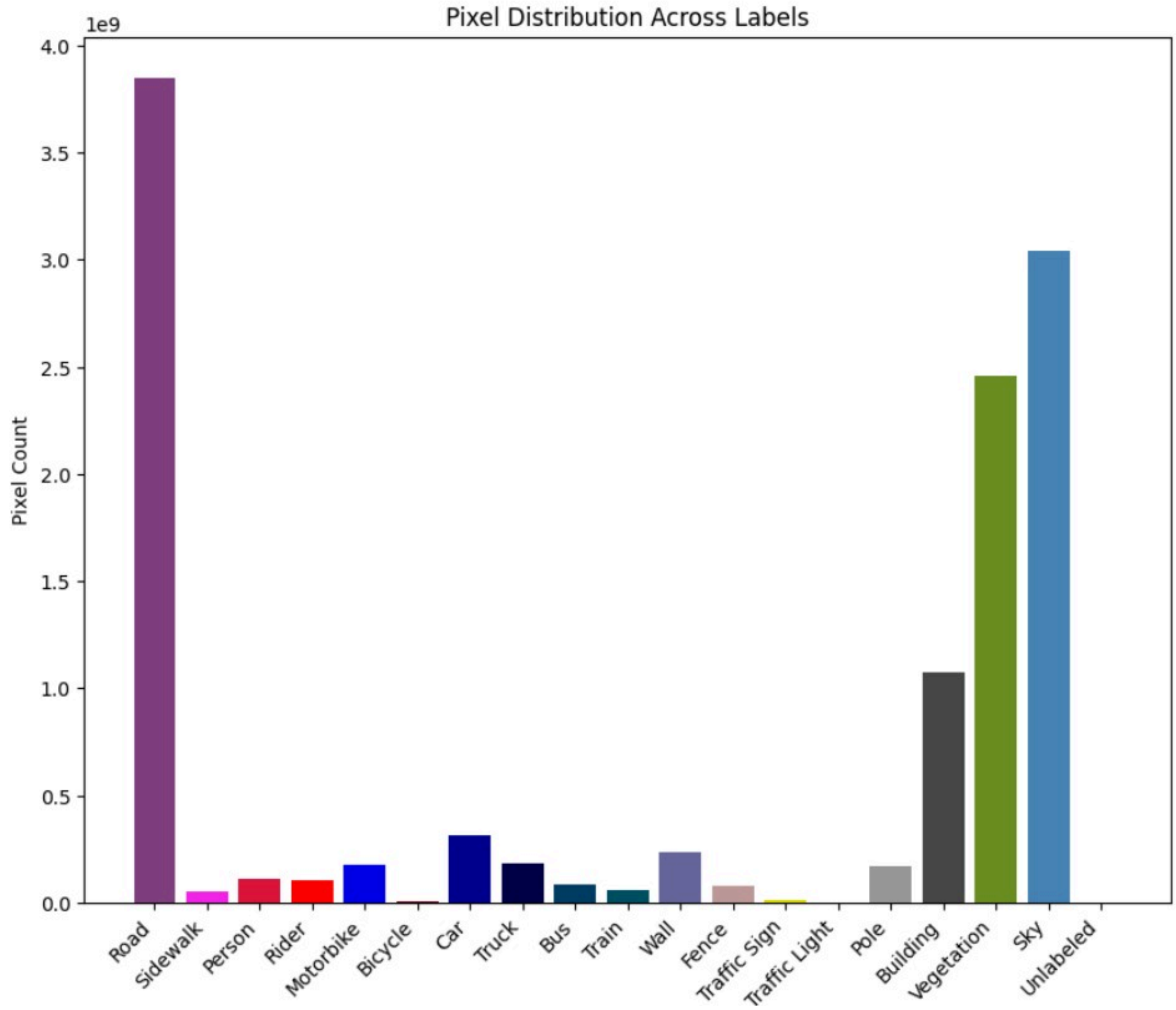
COMMENT AND CONTRAST

The accuracy is the best in the second case when using the fine tuned training model.
The least performing model on train, validation and test set is the custom cnn architecture.
We see that in the 3rd case when we use data augmentation with a fine tuned training model, we reduce the overfitting.

QUESTION 3:

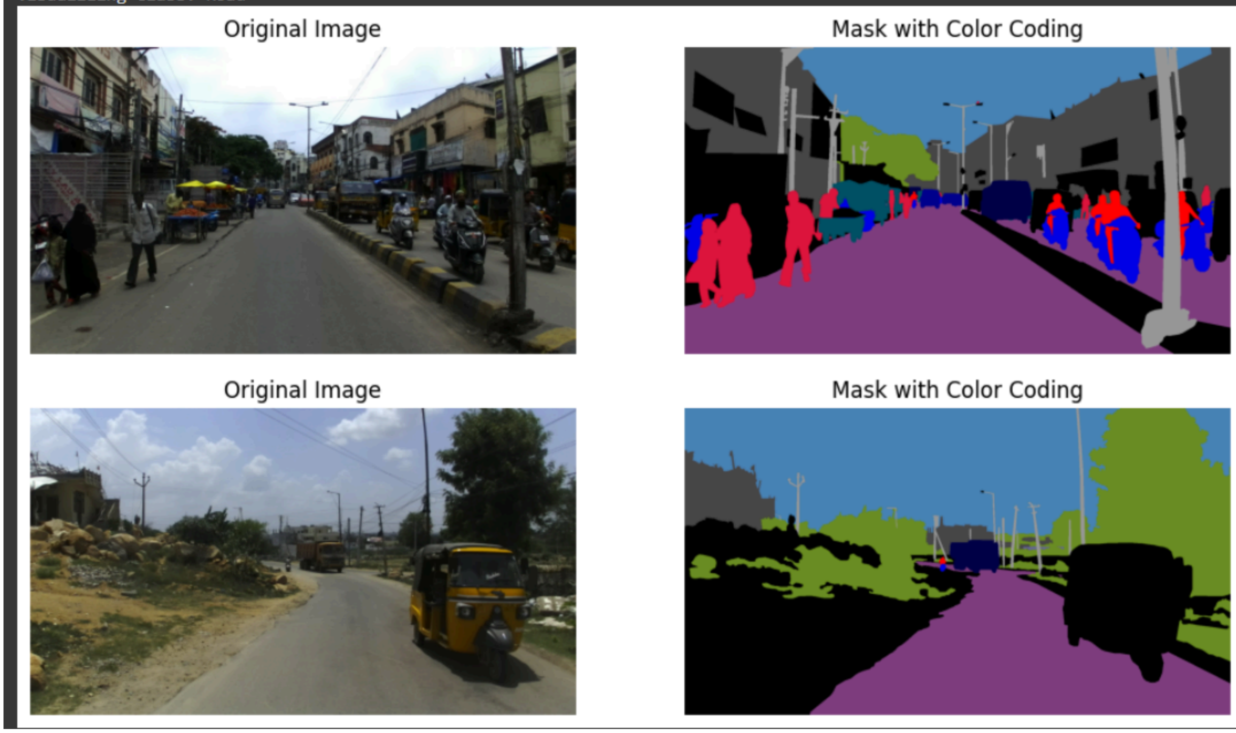
PART 1:

b)

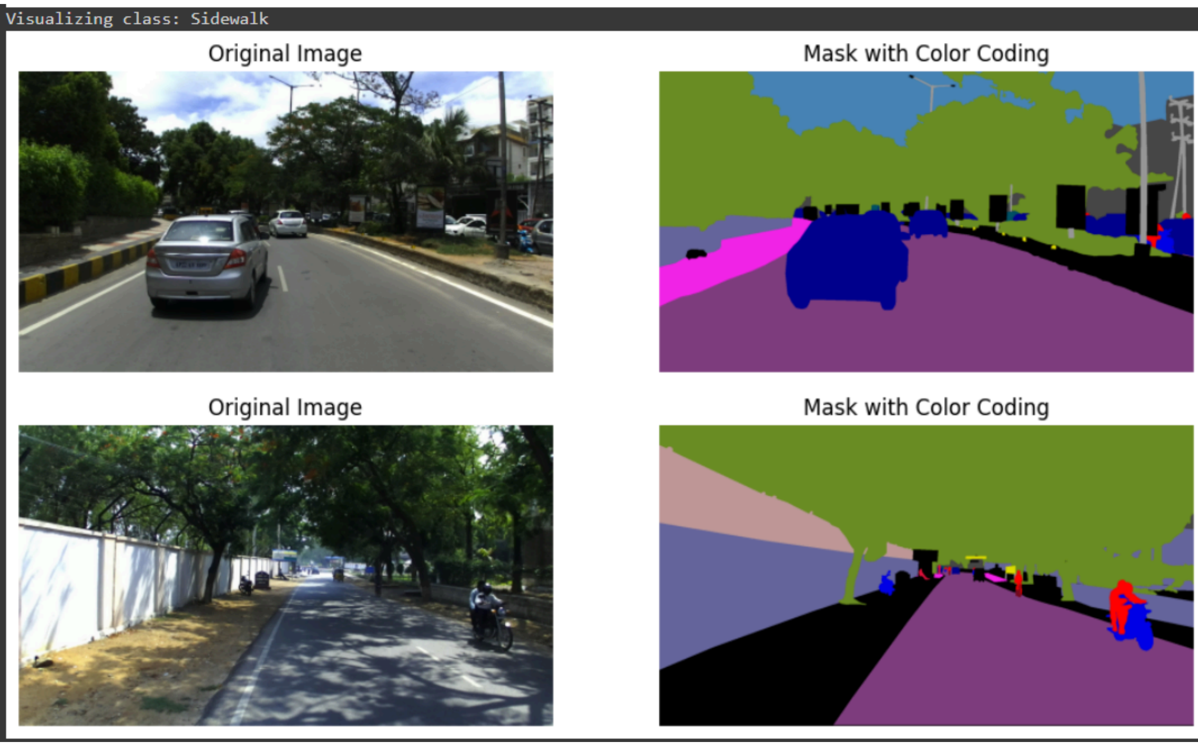

Pixel Distribution Across Labels
1e9
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Pixel Count
Road
Sidewalk
Person
Rider
Motorbike
Bicycle
Car
Truck
Bus
Train
Wall
Fence
Traffic Sign
Traffic Light
Pole
Building
Vegetation
Sky
Unlabeled

c)

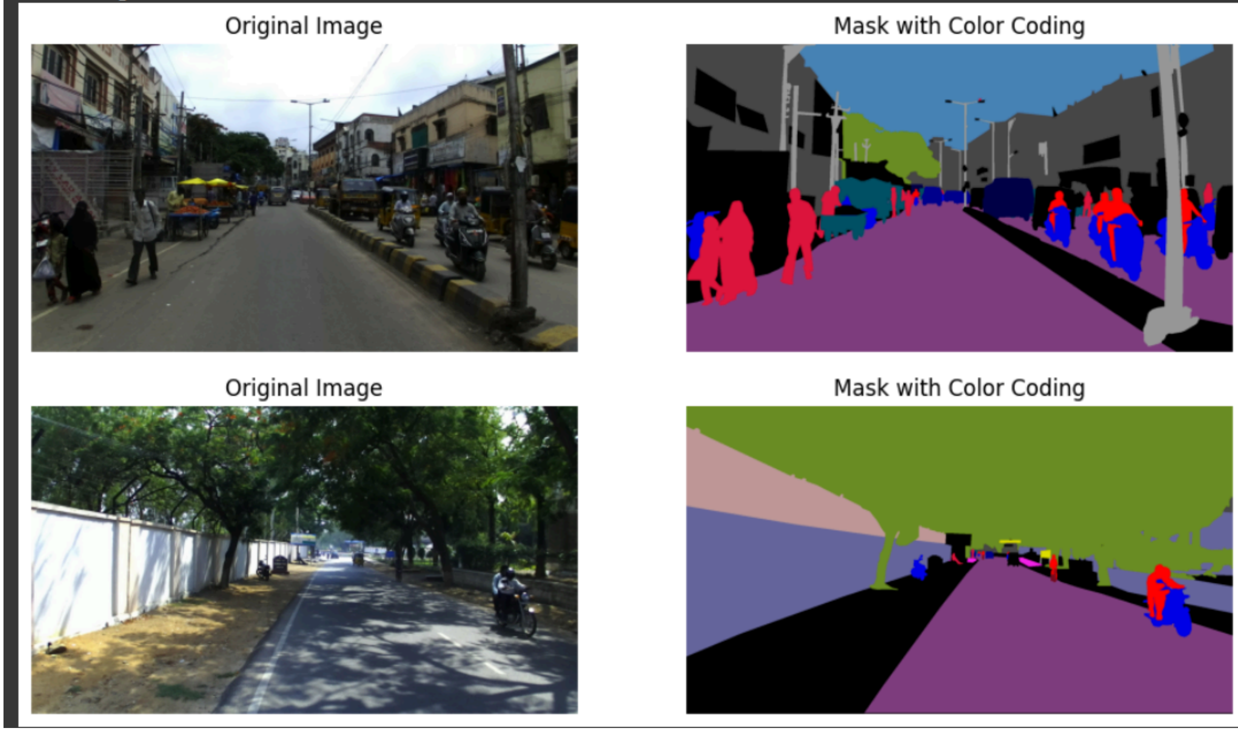
```
Visualizing class: Road
```



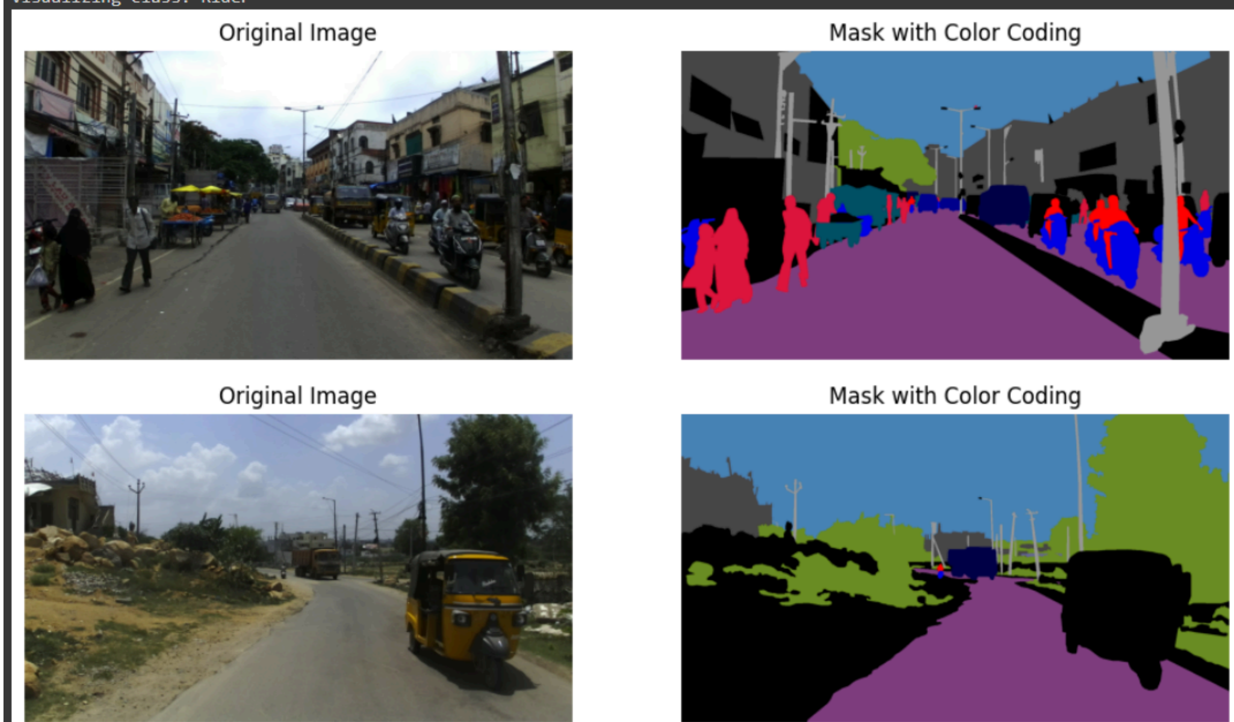

```
Visualizing class: Sidewalk
```



Visualizing class: Person



Visualizing class: Rider



Visualizing class: Motorbike

Visualizing class: Motorbike

Original Image

Mask with Color Coding

Original Image

Mask with Color Coding

Visualizing class: Bicycle

Original Image

Mask with Color Coding

Original Image

Mask with Color Coding

```
Visualizing class: Car
```

Original Image

Mask with Color Coding

Original Image

Mask with Color Coding

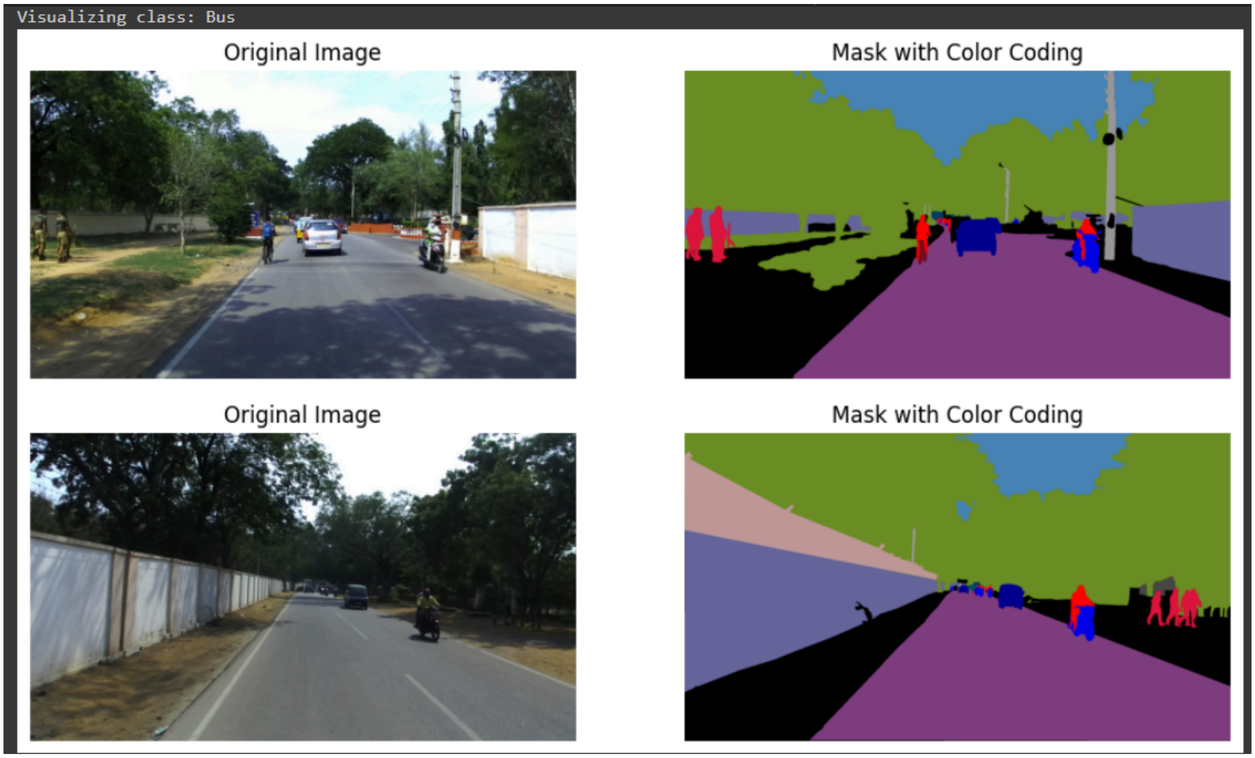
```
Visualizing class: Truck
```

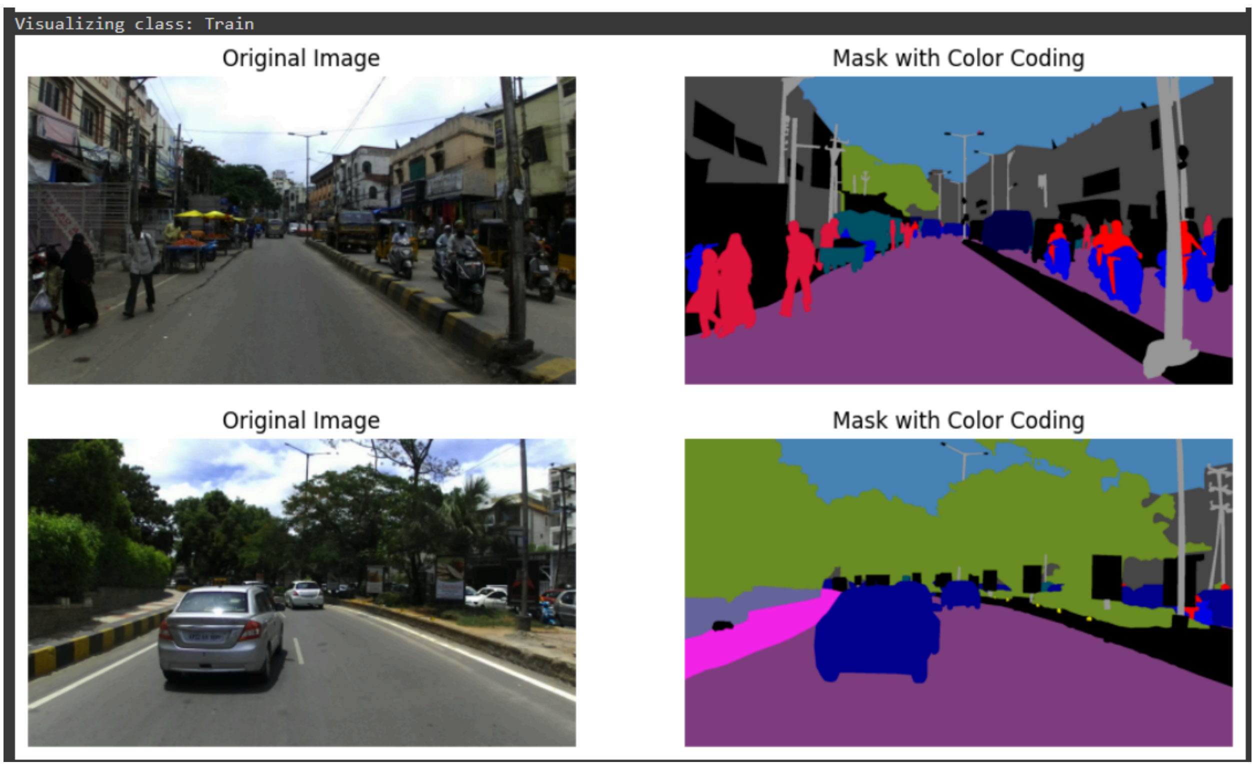
Original Image

Mask with Color Coding

Original Image

Mask with Color Coding

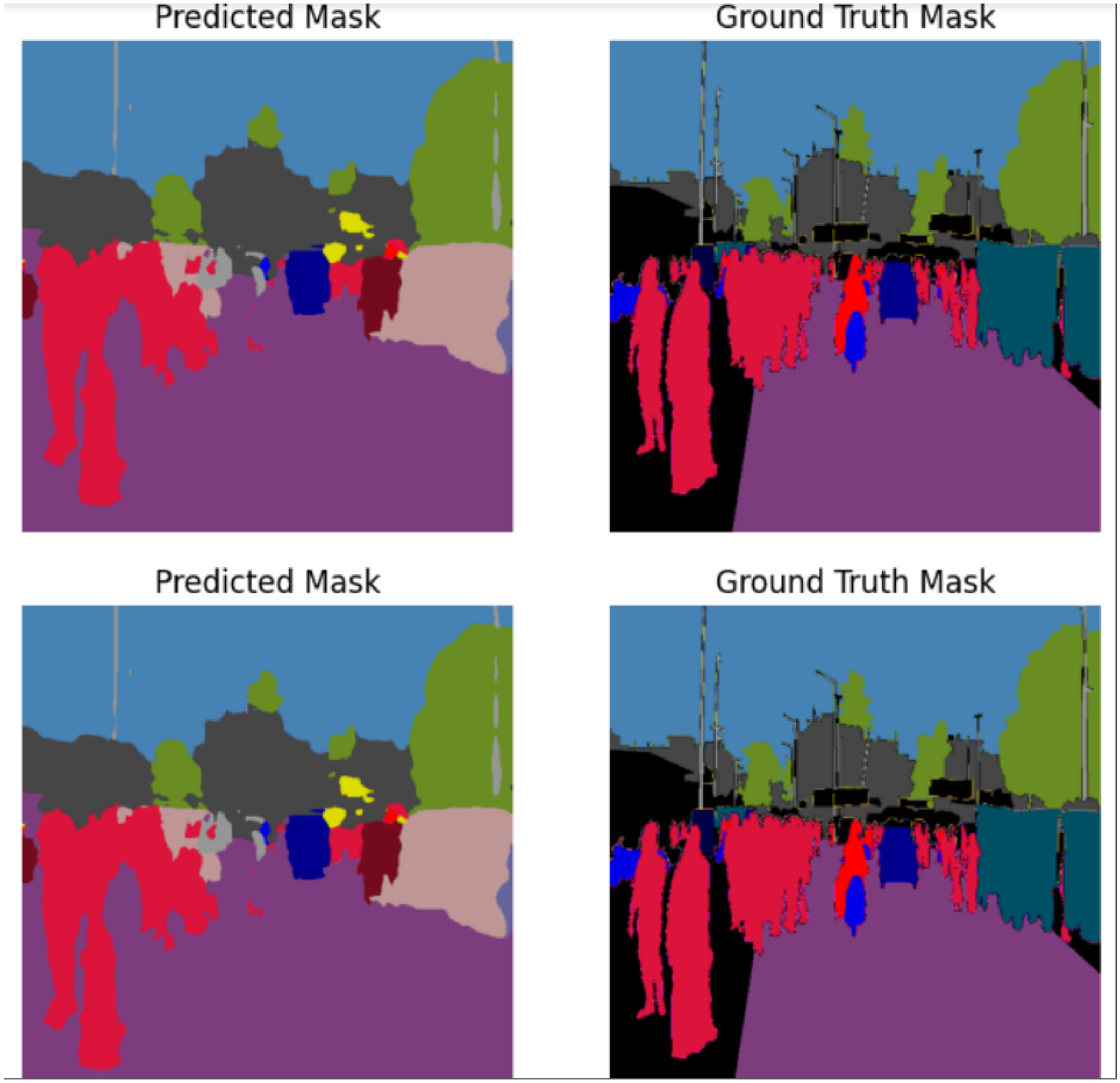

Visualizing class: Bus
Original Image
Mask with Color Coding
Original Image
Mask with Color Coding


Visualizing class: Train
Original Image
Mask with Color Coding
Original Image
Mask with Color Coding

```
Visualizing class: Sky
```

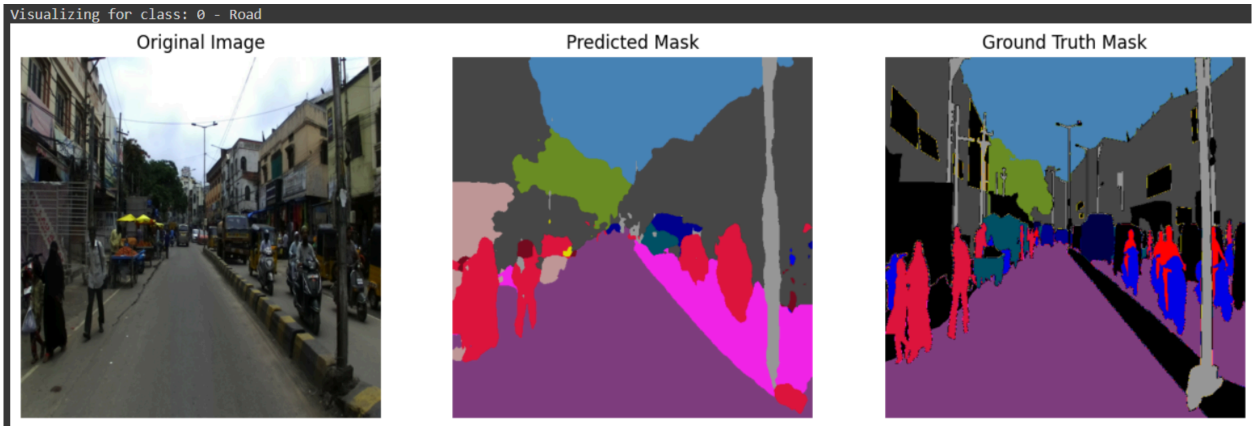
PART 2:

```
accuracy: {0: 0.8102181500770014, 2: 0.8700833267355911, 4: 0.9598400873292886, 5: 0.9733676986479494, 6: 0.978980223034155, 7: 0.9850608868901982, 9: 0.9757232369668548, 10: 0.983468272174738, 11: 0.9676768356847906, 12: 0.953981679
dice_coefficient: {0: 0.8599099835536917, 2: 0.0, 4: 0.1771374256739133, 5: 0.0, 6: 0.42766302091783104, 7: 0.32107454832183213, 9: 0.31165140983102, 10: 0.061529953105728834, 11: 0.0, 12: 0.6433130553205688, 14: 0.33148794895952854,
iou: {0: 0.7542474463850273, 2: 0.0, 4: 0.09717541419127611, 5: 0.0, 6: 0.2719919626691434, 7: 0.19123812078785418, 9: 0.18458949274203387, 10: 0.031741503153122914, 11: 0.0, 12: 0.47417943973255816, 14: 0.19867279277534447, 15: 0.33
precision: {0: 0.7687637766736989, 2: 0, 4: 0.10724281901300234, 5: 0, 6: 0.3290745599057535, 7: 0.6151001513698494, 9: 0.606244131455399, 10: 0.03473643555317879, 11: 0, 12: 0.49784973523576026, 14: 0.6607059315044945, 15: 0.6406846
recall: {0: 0.9755763073814775, 2: 0.0, 4: 0.5086379155952531, 5: 0.0, 6: 0.6105922269152593, 7: 0.21723397705328906, 9: 0.2097348011728436, 10: 0.2690864050816181, 11: 0.0, 12: 0.9088694129365281, 14: 0.2212453411736417, 15: 0.41232
```


Predicted Mask
Ground Truth Mask
Predicted Mask
Ground Truth Mask

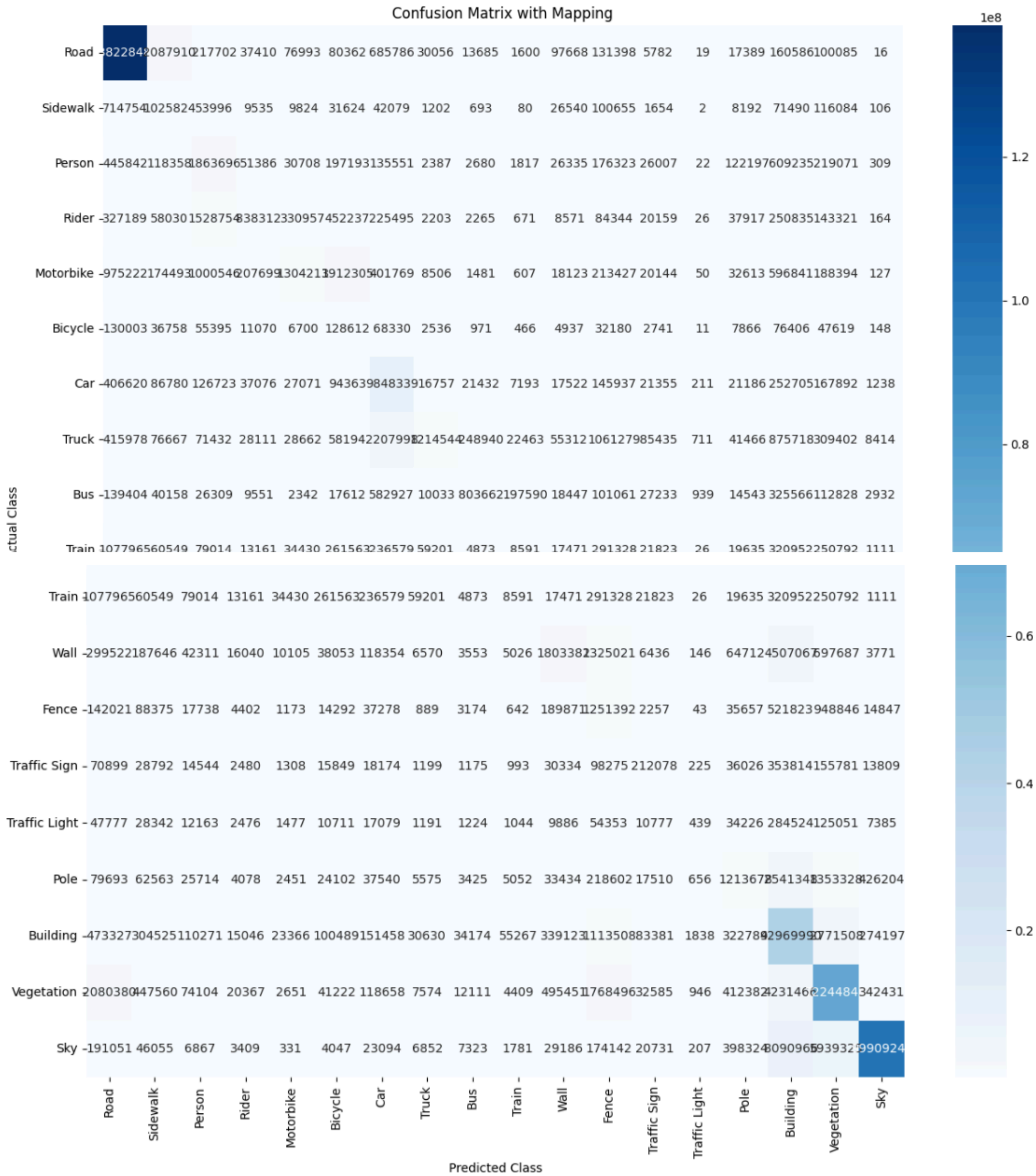
a)

b)



This might be because of the image size and instances of classes being less in the dataset. The objects belonging to those classes might be misclassified because of overlap, hindrance from other objects.

PART 3:



The model correctly identifies the road and sky the most.

```
Class: Road
  Precision: 0.7415688352961356
  Recall: 0.9733343535981208
  F1 Score: 0.8417902860357704

Class: Sidewalk
  Precision: 0.0860101745861727
  Recall: 0.45539635158723535
  F1 Score: 0.14469245497561212

Class: Person
  Precision: 0.2971849254189731
  Recall: 0.460575224770929
  F1 Score: 0.361264745286679

Class: Rider
  Precision: 0.5790051310599379
  Recall: 0.19390706148845305
  F1 Score: 0.29051989251299054

Class: Motorbike
  Precision: 0.5828244134946504
  Recall: 0.1845462650837334
  F1 Score: 0.28032884683401044

Class: Bicycle
  Precision: 0.030589880070792552
  Recall: 0.20705399000566688
  F1 Score: 0.05330460844973287

Class: Car
  Precision: 0.47893001730128626
  Recall: 0.8702860648826829
  F1 Score: 0.6178493209725898
```

```
Class: Truck
  Precision: 0.6211128794046533
  Recall: 0.17782629613762685
  F1 Score: 0.2764921441058085

Class: Bus
  Precision: 0.6007641285667619
  Recall: 0.32893948938089573
  F1 Score: 0.42511407264467627

Class: Train
  Precision: 0.018822905620360552
  Recall: 0.0030993058579442123
  F1 Score: 0.0053222679391126595

Class: Wall
  Precision: 0.3106251250935726
  Recall: 0.19680368162097833
  F1 Score: 0.24094874951307732

Class: Fence
  Precision: 0.08847100357399972
  Recall: 0.37827558553136514
  F1 Score: 0.14340295766766675
```

```
Class: Fence
  Precision: 0.08847100357399972
  Recall: 0.37827558553136514
  F1 Score: 0.14340295766766675

Class: Traffic Sign
  Precision: 0.15973674220741269
  Recall: 0.1976678183728042
  F1 Score: 0.17668948205282656

Class: Traffic Light
  Precision: 0.029574238749663164
  Recall: 0.0006634456556399681
  F1 Score: 0.0012977779617199844

Class: Pole
  Precision: 0.32413600445257756
  Recall: 0.1996899870693577
  F1 Score: 0.24713059521842398

Class: Building
  Precision: 0.4888808590697916
  Recall: 0.8729580430913074
  F1 Score: 0.626759123066786

Class: Vegetation
  Precision: 0.7723483120194709
  Recall: 0.858820290660744
  F1 Score: 0.8132922626514487

Class: Sky
  Precision: 0.9798044124600327
  Recall: 0.8695689099830981
  F1 Score: 0.921401260199805
```

The model performs well on road and sky and needs improvement in bicycle, train and fence.