DL Assignment-2 [700 Level]

DenseNet Fine Tuning, ResNet+SVM and CNN

By:- Kwanit Gupta
B19EE046

Question-1

Download a ResNet 50 trained on the ImageNet classification dataset:-

a. Use the features extracted from the last fully-connected layer and train a multiclass SVM classifier on STL-10 dataset. Report the following [20 marks]
   i. Accuracy, Confusion Matrix on test data. [15 marks]
   ii. ROC curve (assuming the chosen class as positive class and remaining classes as negative) [15 marks]

b. Fine-tune the ResNet 50 model (you may choose what layers to fine-tune) for the STL-10 dataset, and evaluate the classification performance on the test set before and after fine-tuning with respect to the following metrics,
   i. Class wise Accuracy [10 Marks]
   ii. Report Confusion Matrix. [15 Marks]

[Code for accuracy, ROC, Confusion Matrix should be done from scratch, SVM - you may use sklearn]

Soln-1

For the (a) subpart, pre-trained ResNet-50 on ImageNet was used alongside different variants of Support Vector Machines (SVMs), as follows:-

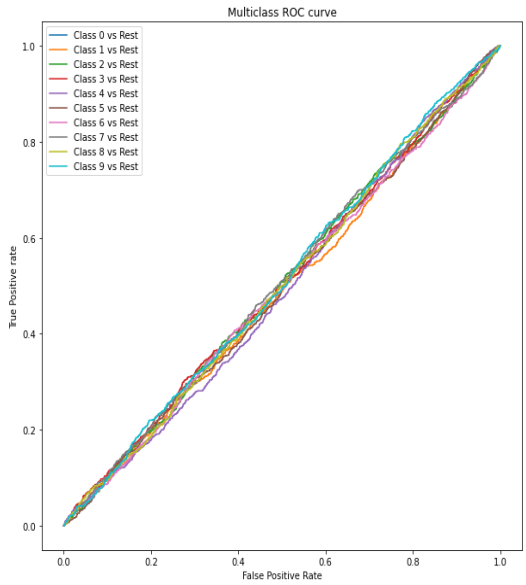
1. Radial Basis Function (RBF) Kernel
2. Linear Kernel
3. 3-Degree Polynomial Kernel

With different types of normalization methods, in order to understand the influence of changes in the distribution of extracted features, as follows:-

1. Default Configuration (No Normalization Method incorporated)
2. Standard Scalar (Plots to Normal Gaussian Distribution)
3. Min-Max Scalar (Plots to Quartile-based distribution)

As a matter of instruction, the Overall Accuracy and Confusion Matrix on Test Data was found out alongside the respective ROC Curve (one-v/s-rest configuration), as follows:-

RBF only



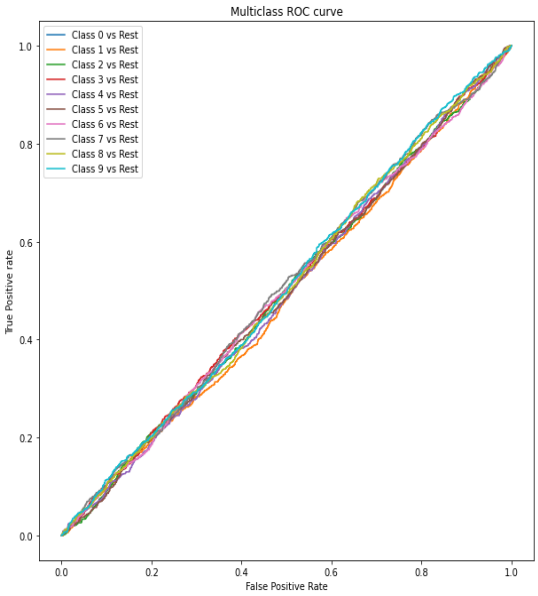
Confusion Matrix

[78, 83, 117, 86, 86, 83, 65, 89, 60, 53]
[77, 72, 109, 87, 83, 66, 68, 98, 66, 74]
[67, 92, 119, 79, 86, 88, 65, 77, 67, 60]
[77, 99, 113, 79, 83, 67, 70, 75, 76, 61]
[88, 72, 105, 92, 91, 77, 66, 82, 68, 59]
[91, 82, 94, 92, 79, 72, 69, 91, 70, 60]
[80, 88, 109, 85, 81, 81, 62, 88, 61, 65]
[60, 73, 117, 87, 93, 85, 72, 63, 71, 79]
[57, 66, 111, 88, 84, 77, 75, 90, 97, 55]
[71, 104, 134, 75, 62, 83, 74, 72, 72, 53]

Overall-Accuracy

9.825%

RBF with Standard Scalar



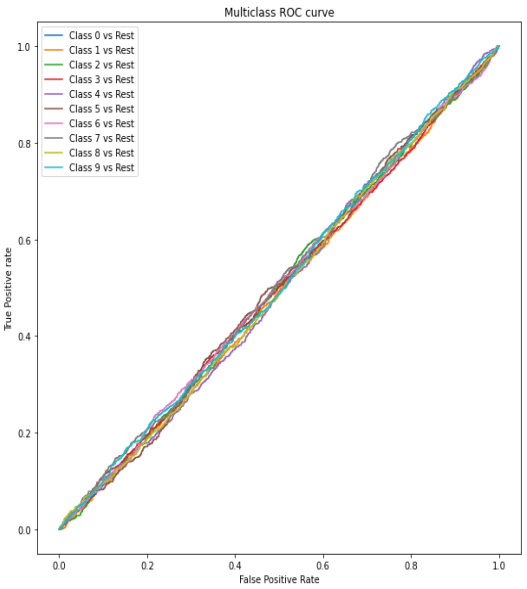
Confusion Matrix

[73, 80, 118, 92, 81, 83, 65, 92, 63, 53]
[77, 61, 109, 89, 87, 66, 72, 94, 69, 76]
[64, 89, 122, 88, 85, 92, 61, 79, 62, 58]
[79, 98, 106, 79, 88, 63, 74, 73, 81, 59]
[91, 76, 107, 95, 91, 75, 68, 75, 67, 55]
[88, 78, 94, 90, 82, 78, 65, 92, 71, 62]
[75, 87, 111, 84, 88, 76, 69, 84, 60, 66]
[61, 78, 112, 92, 96, 84, 70, 62, 73, 72]
[58, 60, 111, 91, 81, 78, 72, 89, 105, 55]
[68, 102, 139, 73, 59, 85, 75, 71, 77, 51]

Overall-Accuracy

9.8875%

RBF with Min-Max Scalar



Confusion Matrix

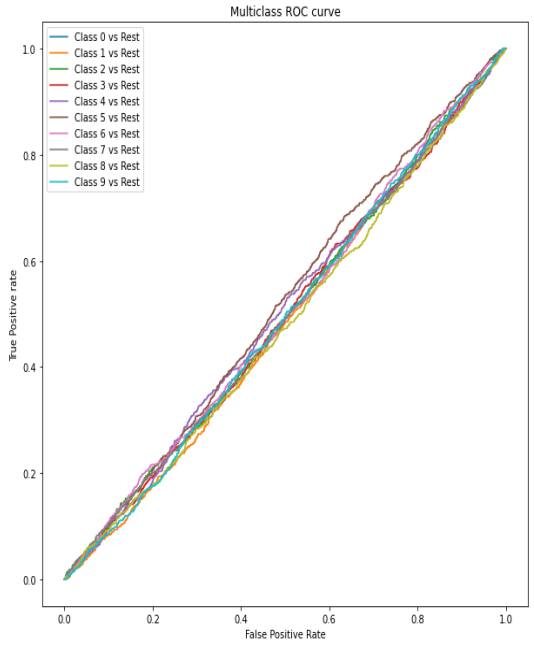
[72, 84, 107, 81, 94, 85, 65, 99, 59, 54]
[71, 66, 106, 90, 95, 63, 74, 99, 65, 71]
[61, 88, 111, 77, 94, 100, 60, 76, 68, 65]
[70, 99, 104, 73, 99, 69, 74, 82, 71, 59]
[87, 77, 100, 86, 106, 74, 70, 81, 67, 52]
[84, 85, 76, 93, 96, 76, 65, 93, 70, 62]
[77, 86, 105, 77, 93, 81, 69, 85, 59, 68]
[57, 74, 108, 77, 106, 84, 69, 66, 87, 72]
[64, 78, 104, 84, 86, 77, 77, 85, 89, 56]
[68, 96, 123, 76, 82, 88, 71, 74, 70, 52]

Overall-Accuracy

9.75%

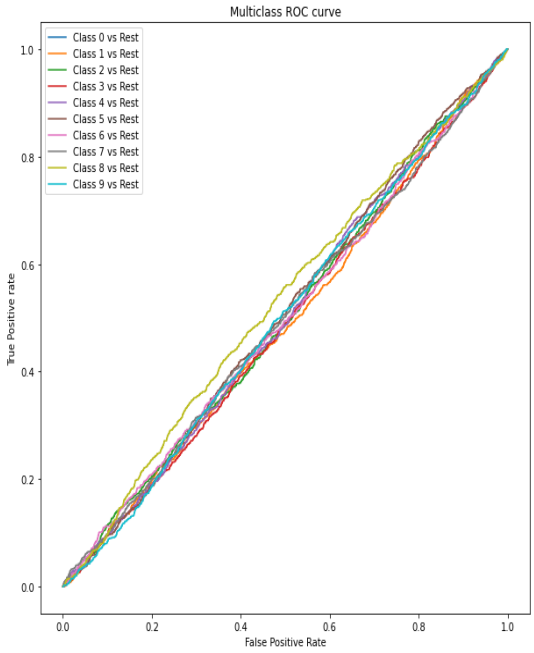
From the above ROC Curves, Confusion Matrix, and Overall Accuracy Scores, it can be clearly said that the change in orientation of feature maps, didn't affect the results in a drastic manner. In fact, degradation was observed too in one of the cases.

Linear only



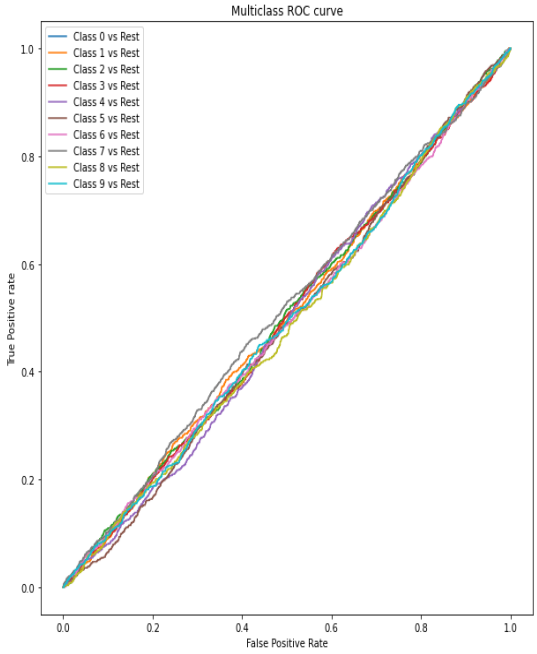

```
Confusion Matrix

[97, 96, 118, 74, 81, 75, 69, 70, 56, 64]
[77, 110, 98, 82, 97, 80, 62, 85, 56, 53]
[119, 110, 86, 88, 78, 72, 75, 71, 54, 47]
[99, 117, 102, 78, 79, 74, 83, 62, 47, 59]
[90, 94, 114, 95, 65, 68, 75, 75, 54, 70]
[95, 95, 85, 94, 96, 87, 73, 63, 46, 66]
[105, 111, 94, 87, 68, 78, 69, 73, 49, 66]
[106, 98, 92, 76, 80, 70, 77, 67, 61, 73]
[115, 103, 102, 90, 76, 71, 64, 62, 63, 54]
[91, 102, 102, 94, 76, 85, 47, 87, 60, 56]

Overall-Accuracy

9.725%
```

Linear with Standard Scalar




```
Confusion Matrix

[104, 99, 113, 80, 89, 65, 70, 68, 63, 49]
[73, 104, 111, 86, 96, 55, 76, 81, 64, 54]
[104, 113, 91, 81, 76, 79, 74, 74, 56, 52]
[85, 110, 103, 81, 77, 74, 76, 62, 63, 69]
[93, 87, 89, 101, 82, 67, 76, 67, 68, 70]
[91, 102, 80, 89, 91, 68, 69, 82, 58, 70]
[115, 103, 90, 97, 70, 71, 81, 58, 58, 57]
[98, 92, 105, 74, 76, 73, 79, 74, 56, 73]
[108, 96, 96, 91, 80, 72, 78, 60, 63, 56]
[103, 114, 101, 87, 67, 77, 62, 71, 58, 60]

Overall-Accuracy

10.1%
```

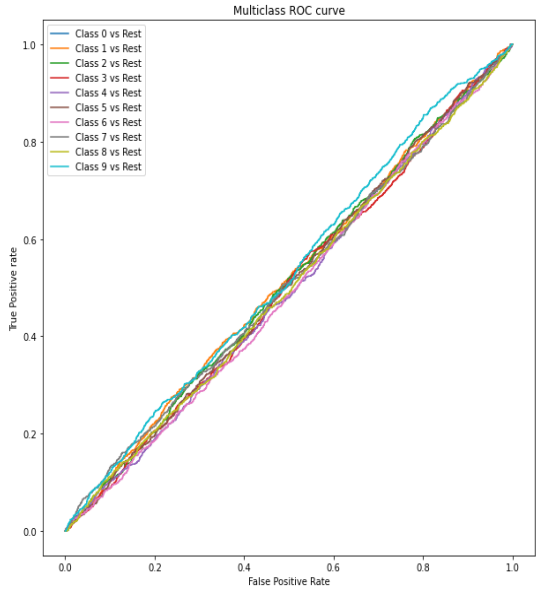
Linear with Min-Max Scalar




```
Confusion Matrix

[87, 105, 89, 98, 83, 76, 76, 69, 62, 55]
[95, 114, 89, 89, 86, 55, 67, 80, 49, 76]
[95, 112, 94, 74, 82, 64, 74, 78, 50, 77]
[99, 103, 97, 73, 88, 69, 72, 79, 54, 66]
[101, 119, 90, 89, 83, 66, 70, 60, 63, 59]
[105, 98, 85, 88, 110, 61, 75, 58, 49, 71]
[122, 98, 78, 85, 79, 70, 76, 69, 46, 77]
[76, 101, 99, 73, 90, 62, 83, 63, 70, 83]
[80, 98, 92, 80, 81, 86, 82, 61, 59, 81]
[91, 128, 93, 72, 74, 87, 69, 73, 44, 69]

Overall-Accuracy

9.7375%
```

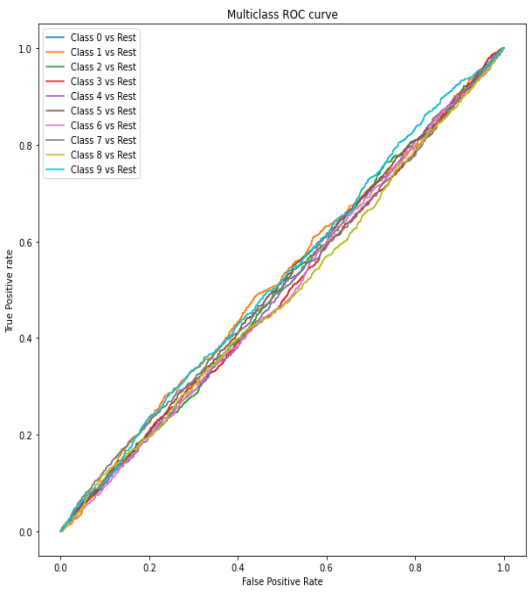
From the above ROC Curves, Confusion Matrix, and Overall Accuracy Scores, it can be clearly said that the change in orientation of feature maps, didn't affect the results in a drastic manner. In fact, degradation was observed too in one of the cases.

Cubic only



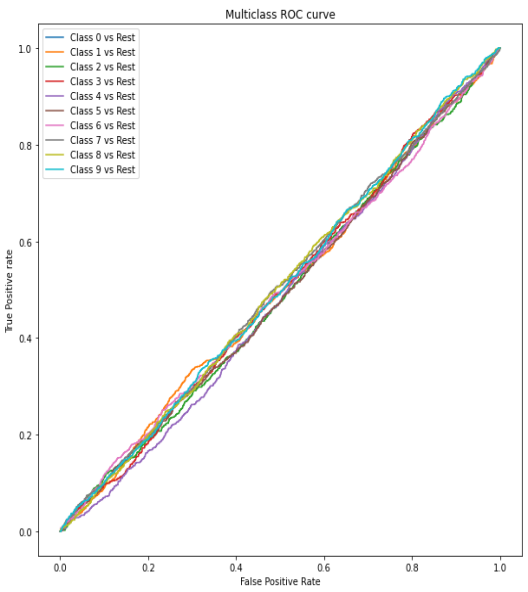

```
Confusion Matrix

[33, 59, 60, 38, 283, 39, 40, 147, 37, 64]
[43, 53, 56, 39, 285, 37, 38, 142, 40, 67]
[37, 46, 66, 47, 278, 63, 28, 140, 30, 65]
[36, 63, 52, 51, 297, 38, 38, 119, 37, 69]
[45, 51, 50, 33, 292, 38, 26, 144, 47, 74]
[40, 48, 54, 40, 294, 48, 30, 139, 35, 72]
[52, 56, 47, 43, 274, 43, 34, 135, 47, 69]
[40, 49, 50, 37, 282, 58, 26, 128, 57, 73]
[29, 48, 67, 42, 278, 41, 39, 150, 44, 62]
[38, 52, 66, 43, 268, 62, 38, 110, 44, 79]

Overall-Accuracy

10.35%
```

Cubic with Standard Scalar




```
Confusion Matrix

[37, 64, 53, 43, 271, 36, 38, 150, 34, 74]
[53, 53, 65, 33, 276, 35, 34, 134, 46, 71]
[43, 58, 62, 49, 262, 45, 30, 134, 45, 72]
[41, 66, 55, 48, 271, 39, 40, 116, 46, 78]
[33, 66, 58, 29, 283, 46, 30, 132, 54, 69]
[39, 50, 44, 37, 290, 46, 38, 124, 57, 75]
[45, 55, 48, 48, 265, 37, 42, 142, 50, 68]
[43, 56, 57, 34, 283, 40, 35, 116, 54, 82]
[44, 58, 62, 47, 271, 34, 43, 134, 42, 65]
[26, 59, 68, 40, 253, 50, 46, 127, 48, 83]

Overall-Accuracy

10.15%
```

Cubic with Min-Max Scalar




```
Confusion Matrix

[105, 92, 95, 85, 73, 92, 64, 74, 62, 58]
[76, 97, 92, 100, 93, 65, 69, 80, 74, 54]
[100, 115, 96, 78, 63, 74, 65, 70, 62, 77]
[95, 94, 101, 89, 62, 75, 86, 62, 65, 71]
[98, 86, 95, 98, 74, 74, 80, 55, 72, 68]
[97, 90, 100, 99, 74, 70, 70, 73, 55, 72]
[95, 93, 86, 79, 70, 75, 80, 87, 67, 68]
[99, 92, 107, 87, 71, 69, 80, 63, 62, 70]
[92, 95, 100, 87, 71, 77, 74, 62, 78, 64]
[89, 91, 107, 89, 72, 75, 77, 58, 78, 64]

Overall-Accuracy

10.2%
```

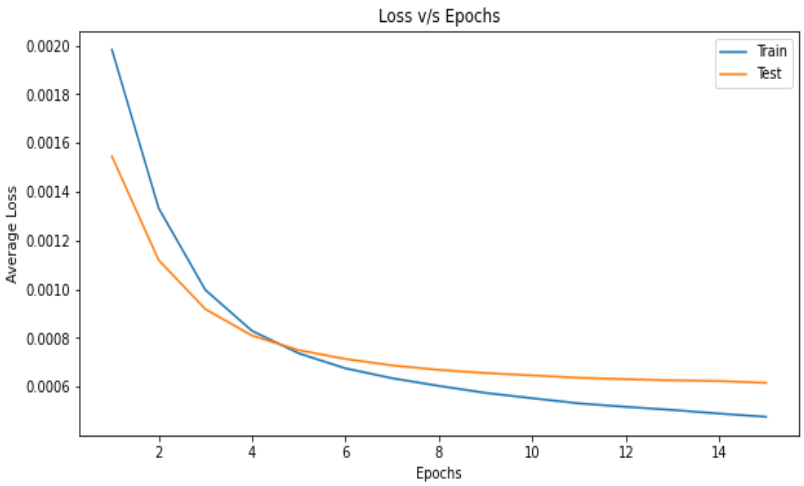
From the above ROC Curves, Confusion Matrix, and Overall Accuracy Scores, it can be clearly said that the change in orientation of feature maps, didn't affect the results in a drastic manner. In fact, degradation was observed too in one of the cases.

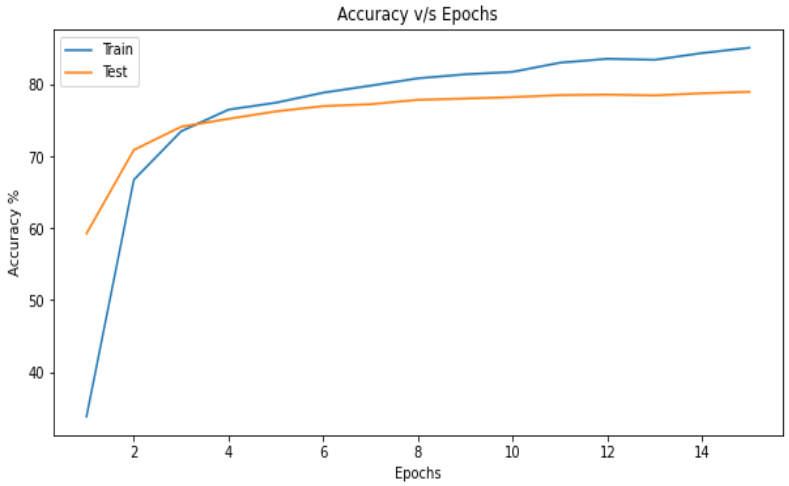
In regards to (b) subpart, ResNet-50 was fine-tuned at different layers in the context of implementation point-of-view, which can be seen as follows:-

1. Default Configuration (Base-Network Frozen and a classifier block added)
2. Layer-1 Trainable and rest Frozen
3. Layer-2 Trainable and rest Frozen
4. Layer-3 Trainable and rest Frozen
5. Layer-4 Trainable and rest Frozen
6. Convolution Layers Trainable and rest Frozen
7. Batch-Normalization Layers Trainable and rest Frozen
8. Downsampling Layers Trainable and rest Frozen

To analyze the trend of each and every configuration, respective Loss v/s Epoch and Accuracy v/s Epoch curves were plotted, as well as Class-Wise Accuracies and Confusion Matrices were found out as follows:-

1. Default Configuration (Base-Network Frozen and a classifier block added)



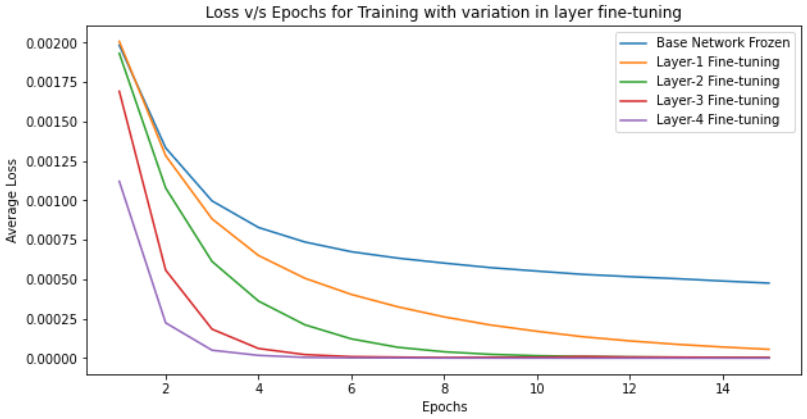


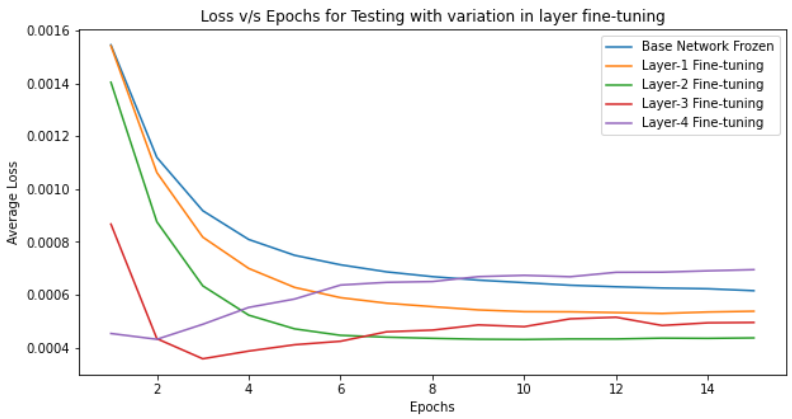
```
Confusion Matrix

[660, 1, 31, 22, 17, 7, 49, 3, 10]
[1, 705, 2, 1, 0, 4, 1, 8, 78]
[21, 0, 558, 78, 74, 14, 44, 6, 5]
[20, 1, 52, 627, 27, 46, 20, 2, 5]
[12, 1, 67, 31, 566, 72, 42, 0, 9]
[13, 8, 22, 43, 78, 593, 17, 5, 21]
[20, 1, 58, 47, 27, 7, 635, 0, 5]
[2, 6, 4, 1, 1, 2, 0, 665, 119]
[9, 57, 5, 6, 10, 10, 6, 132, 1365]
```

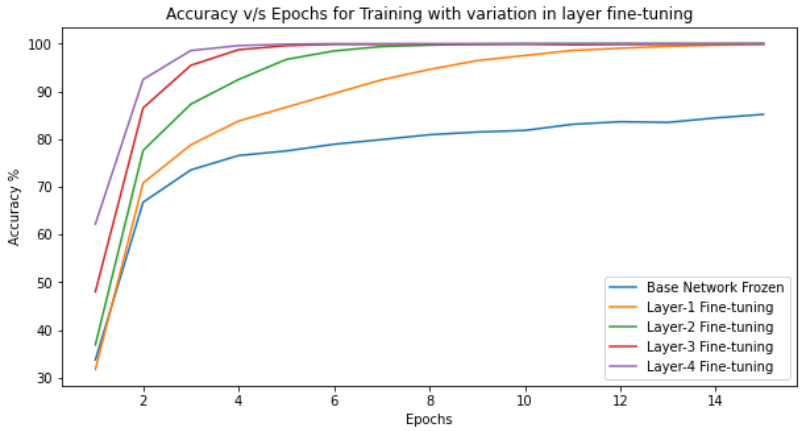
```
Class-Wise Accuracy

For Class-0 :- 82.5%

For Class-1 :- 88.125%

For Class-2 :- 69.75%

For Class-3 :- 78.375%

For Class-4 :- 70.75%

For Class-5 :- 74.125%

For Class-6 :- 79.375%

For Class-7 :- 83.125%

For Class-8 :- 85.3125%

Mean Class-Wise Accuracy :-79.048611111111111%

Overall Accuracy :-79.675%
```

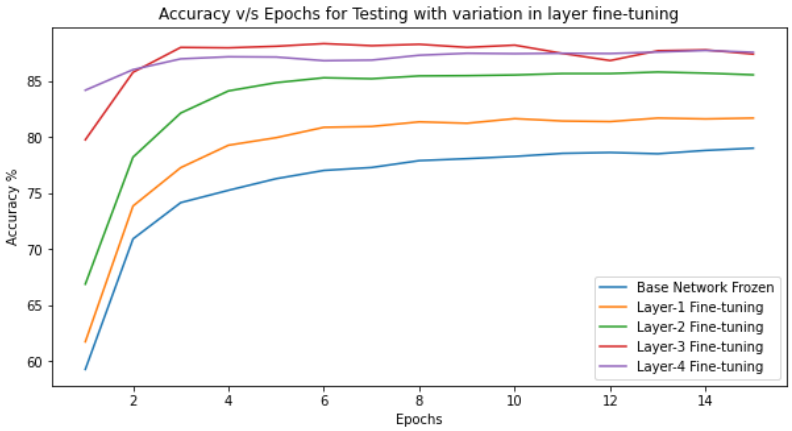
From the above, it can be said that adding FC layers at the end of the pre-trained frozen ResNet-50 Model, outperformed ResNet-50+SVM Combinations by a huge margin. In fact, Confusion Matrix was trend-shifted as well.

2. Default Settings v/s Single-Layer Trainable Configurations










```
Confusion Matrix

[657, 0, 49, 21, 13, 10, 30, 3, 17]
[1, 711, 1, 0, 0, 3, 0, 10, 74]
[17, 1, 589, 65, 64, 16, 41, 3, 4]
[21, 2, 46, 654, 17, 39, 15, 1, 5]
[16, 0, 67, 21, 582, 72, 34, 2, 6]
[15, 8, 12, 33, 49, 641, 18, 3, 21]
[16, 1, 47, 38, 26, 19, 647, 4, 2]
[0, 7, 2, 1, 0, 0, 0, 697, 93]
[8, 55, 5, 2, 2, 7, 4, 126, 1391]

Class-Wise Accuracy

For Class-0 :- 82.125%

For Class-1 :- 88.875%

For Class-2 :- 73.625%

For Class-3 :- 81.75%

For Class-4 :- 72.75%

For Class-5 :- 80.125%

For Class-6 :- 80.875%

For Class-7 :- 87.125%

For Class-8 :- 86.9375%

Mean Class-Wise Accuracy :-81.57638888888889%

Overall Accuracy :-82.1125%
```

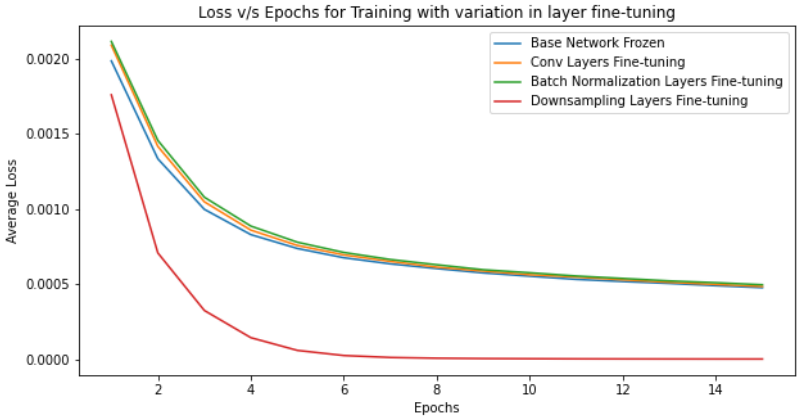
```
Confusion Matrix

[674, 0, 42, 14, 15, 9, 18, 0, 28]
[0, 717, 1, 1, 0, 1, 0, 6, 74]
[17, 0, 627, 51, 58, 8, 32, 1, 6]
[16, 1, 35, 691, 17, 21, 11, 0, 8]
[10, 0, 43, 30, 621, 52, 36, 2, 6]
[4, 3, 10, 20, 51, 683, 11, 3, 15]
[12, 0, 45, 22, 20, 5, 692, 1, 3]
[2, 3, 0, 1, 0, 0, 0, 745, 49]
[6, 45, 2, 3, 1, 3, 1, 89, 1450]

Class-Wise Accuracy

For Class-0 :- 84.25%

For Class-1 :- 89.625%

For Class-2 :- 78.375%

For Class-3 :- 86.375%

For Class-4 :- 77.625%

For Class-5 :- 85.375%

For Class-6 :- 86.5%

For Class-7 :- 93.125%

For Class-8 :- 90.625%

Mean Class-Wise Accuracy :-85.76388888888889%

Overall Accuracy :-86.25%
```

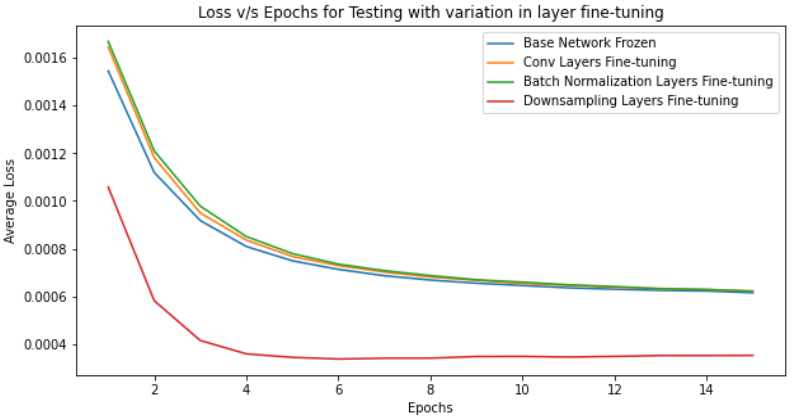
```
Confusion Matrix

[697, 0, 36, 9, 15, 4, 33, 3, 3]
[1, 739, 1, 1, 1, 2, 2, 1, 52]
[14, 0, 629, 30, 80, 3, 41, 1, 2]
[11, 1, 35, 700, 20, 28, 5, 0, 0]
[9, 0, 46, 22, 629, 41, 48, 2, 3]
[10, 2, 7, 20, 55, 692, 7, 1, 6]
[9, 0, 28, 26, 21, 7, 707, 2, 0]
[0, 2, 3, 0, 0, 0, 0, 743, 52]
[14, 42, 2, 4, 6, 7, 1, 45, 1479]

Class-Wise Accuracy

For Class-0 :- 87.125%

For Class-1 :- 92.375%

For Class-2 :- 78.625%

For Class-3 :- 87.5%

For Class-4 :- 78.625%

For Class-5 :- 86.5%

For Class-6 :- 88.375%

For Class-7 :- 92.875%

For Class-8 :- 92.4375%

Mean Class-Wise Accuracy :-87.15972222222223%

Overall Accuracy :-87.6875%
```

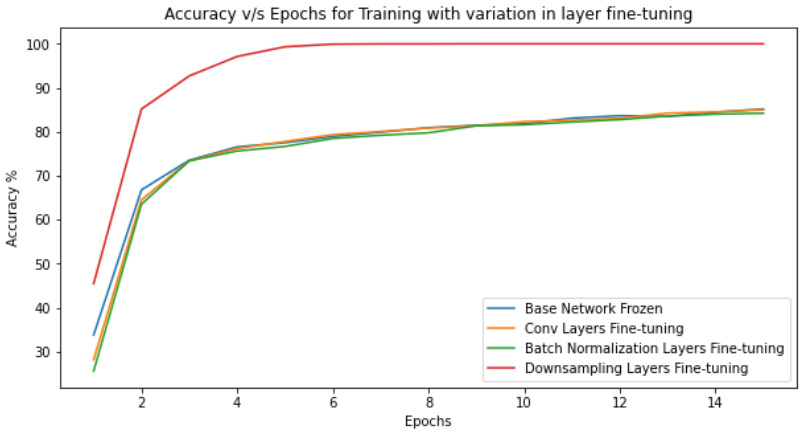
```
Confusion Matrix

[726, 0, 33, 8, 9, 1, 13, 1, 9]
[0, 748, 5, 2, 4, 1, 0, 1, 39]
[23, 0, 621, 44, 73, 7, 32, 0, 0]
[11, 1, 39, 697, 18, 17, 16, 0, 1]
[14, 2, 54, 30, 597, 66, 33, 2, 2]
[5, 2, 10, 26, 45, 693, 14, 0, 5]
[17, 1, 32, 15, 14, 7, 714, 0, 0]
[0, 2, 3, 1, 0, 0, 0, 747, 47]
[4, 35, 9, 3, 3, 11, 6, 49, 1480]

Class-Wise Accuracy

For Class-0 :- 90.75%

For Class-1 :- 93.5%

For Class-2 :- 77.625%

For Class-3 :- 87.125%

For Class-4 :- 74.625%

For Class-5 :- 86.625%

For Class-6 :- 89.25%

For Class-7 :- 93.375%

For Class-8 :- 92.5%

Mean Class-Wise Accuracy :-87.26388888888889%

Overall Accuracy :-87.7875%
```

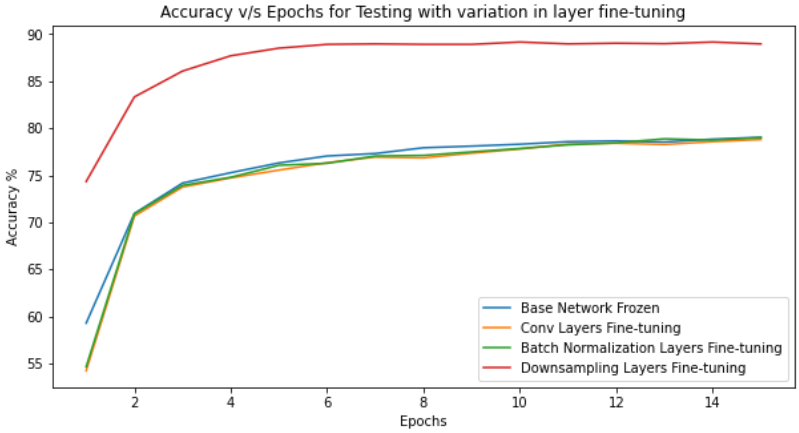
From the above, it can be said that fine-tuning one layer at a time in the pre-trained ResNet-50 Model with respect to STL-10, converges faster as well as Accuracy shoots up at quite initial epochs (Maybe in 2-3 epochs).

3. Default Settings v/s Varied Trainable Layer Configurations










```
Confusion Matrix

[660, 2, 28, 22, 18, 5, 47, 2, 16]
[1, 698, 2, 1, 1, 3, 1, 11, 82]
[22, 1, 557, 87, 64, 18, 41, 5, 5]
[19, 1, 50, 628, 25, 53, 16, 1, 7]
[13, 1, 69, 34, 546, 87, 39, 1, 10]
[13, 7, 19, 42, 57, 623, 17, 3, 19]
[22, 0, 55, 56, 23, 11, 626, 0, 7]
[5, 5, 3, 1, 0, 3, 0, 668, 115]
[11, 54, 6, 7, 7, 12, 6, 146, 1351]

Class-Wise Accuracy

For Class-0 :- 82.5%

For Class-1 :- 87.25%

For Class-2 :- 69.625%

For Class-3 :- 78.5%

For Class-4 :- 68.25%

For Class-5 :- 77.875%

For Class-6 :- 78.25%

For Class-7 :- 83.5%

For Class-8 :- 84.4375%

Mean Class-Wise Accuracy :-78.90972222222223%

Overall Accuracy :-79.4625%
```

```
Confusion Matrix

[676, 2, 23, 24, 18, 7, 37, 3, 10]
[1, 705, 2, 1, 0, 5, 0, 11, 75]
[25, 0, 552, 83, 70, 18, 41, 5, 6]
[25, 1, 52, 623, 22, 53, 17, 2, 5]
[15, 1, 69, 24, 565, 82, 33, 2, 9]
[15, 8, 25, 33, 63, 621, 10, 4, 21]
[27, 1, 55, 53, 27, 11, 618, 3, 5]
[4, 5, 3, 1, 0, 2, 0, 691, 94]
[10, 62, 6, 7, 9, 13, 4, 170, 1319]

Class-Wise Accuracy

For Class-0 :- 84.5%

For Class-1 :- 88.125%

For Class-2 :- 69.0%

For Class-3 :- 77.875%

For Class-4 :- 70.625%

For Class-5 :- 77.625%

For Class-6 :- 77.25%

For Class-7 :- 86.375%

For Class-8 :- 82.4375%

Mean Class-Wise Accuracy :-79.3125%

Overall Accuracy :-79.625%
```

```
Confusion Matrix

[737, 0, 23, 5, 9, 2, 18, 1, 5]
[0, 759, 1, 2, 0, 3, 0, 2, 33]
[15, 0, 648, 42, 61, 6, 23, 3, 2]
[13, 1, 34, 712, 10, 23, 6, 0, 1]
[13, 0, 52, 16, 638, 55, 23, 2, 1]
[4, 1, 12, 19, 47, 708, 6, 0, 3]
[14, 0, 28, 22, 18, 8, 709, 1, 0]
[1, 1, 0, 0, 0, 0, 0, 752, 46]
[5, 45, 1, 1, 1, 6, 2, 62, 1477]

Class-Wise Accuracy

For Class-0 :- 92.125%

For Class-1 :- 94.875%

For Class-2 :- 81.0%

For Class-3 :- 89.0%

For Class-4 :- 79.75%

For Class-5 :- 88.5%

For Class-6 :- 88.625%

For Class-7 :- 94.0%

For Class-8 :- 92.3125%

Mean Class-Wise Accuracy :-88.90972222222223%

Overall Accuracy :-89.25%
```

From the above, it can be said that fine-tuning custom type of layers at a time in the pre-trained ResNet-50 Model with respect to STL-10, didn't impact that much, except for the case of Downsampling Trainable Layers, where the difference of 10% was clearly visible amongst other variants.

Question-2

Download the Tiny ImageNet dataset from here and pick the first 50 classes from 200 of them. Finetune ResNet 18 [20 Marks for ResNet code and data I/O] with the following.

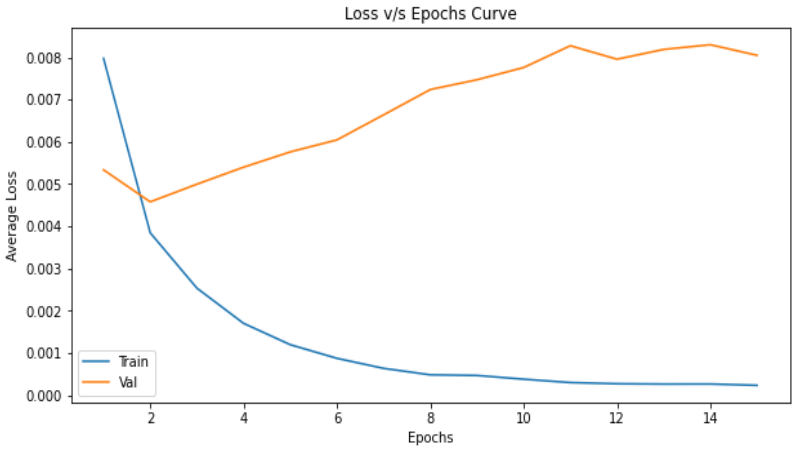
a. Cross-Entropy as the final classification loss function [10 marks]
b. Center loss as the final classification loss function [20 marks]
c. Triplet Loss as the final classification loss function [20 marks]

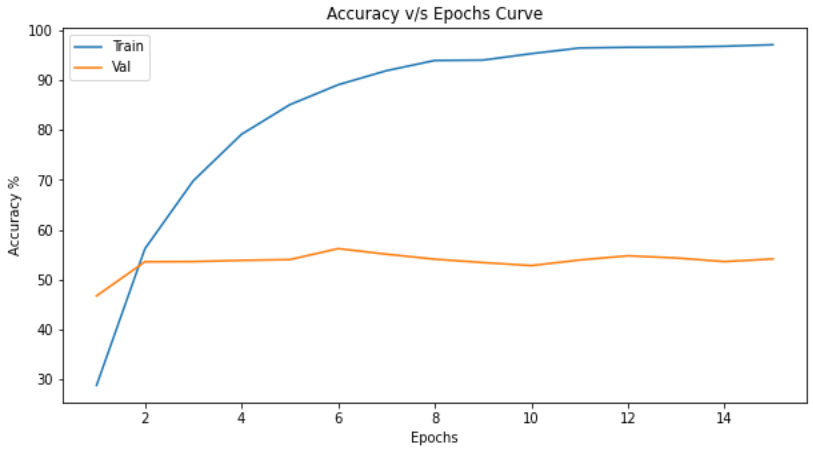
Choose any evaluation metrics (at least 3) and compare the models in a, b and c, comment on which one is better and why? [5+5+5+15 = 30 marks]

Soln-2

Most of the Comparisons and Evaluation analyses here are done in the form of Loss-v/s-Epochs and Accuracy-v/s-Epochs curves.

(a) Cross-Entropy Loss as final-classification-loss function



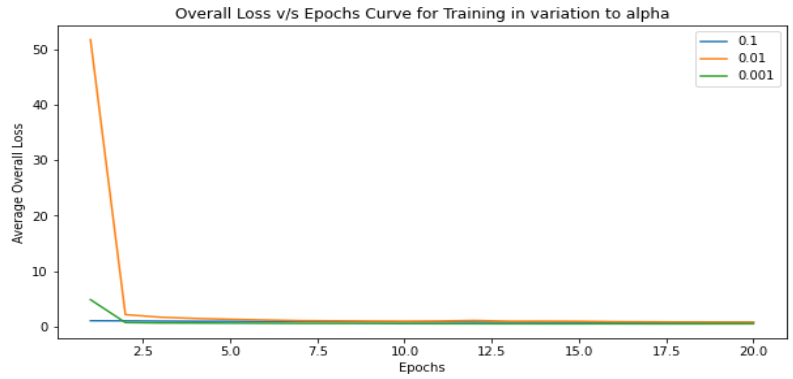


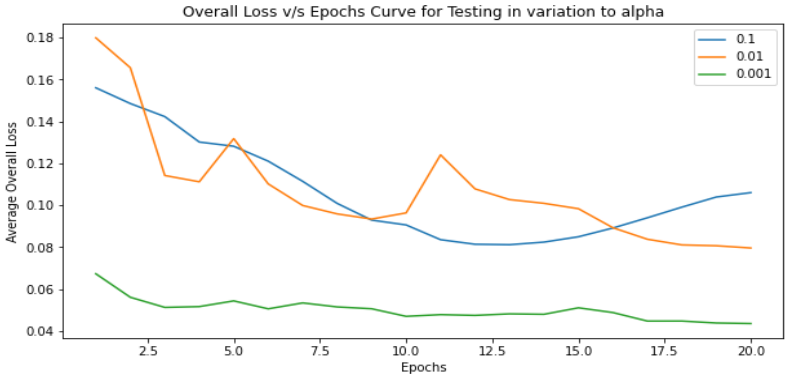
From here, it can be seen that ResNet-18 with Cross-Entropy Loss heavily overfits over the tiny-imagenet dataset.

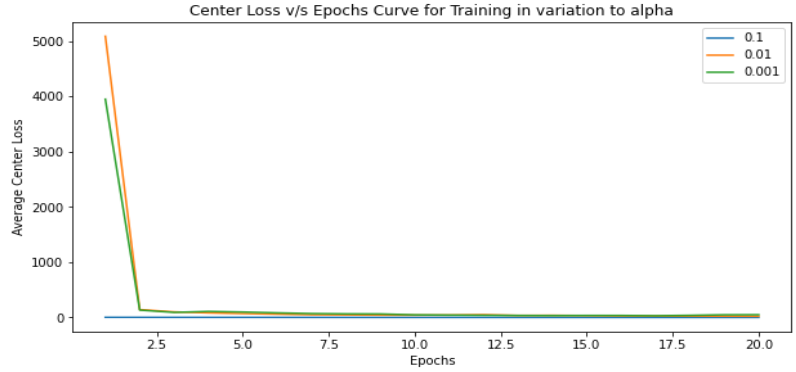
Since Training Loss converge, Training Accuracy peaked nearly to 100%, but Testing Loss kept on increasing with an increase in the number of epochs and Testing Accuracy stagnated to nearly 50%.

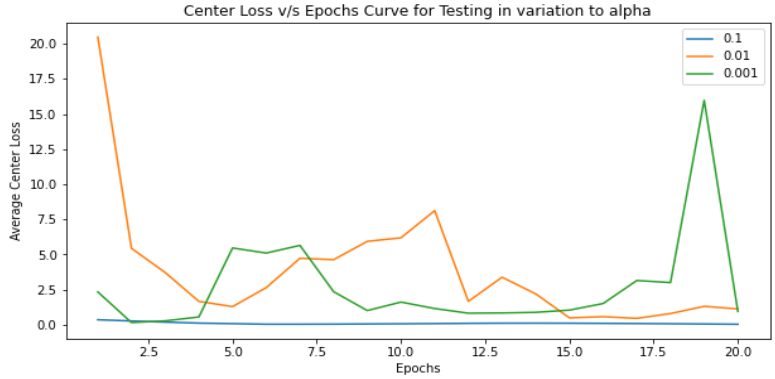
(b) Center-Loss as final-classification-loss function (Complemented by Cross-entropy loss for label-wise classification)

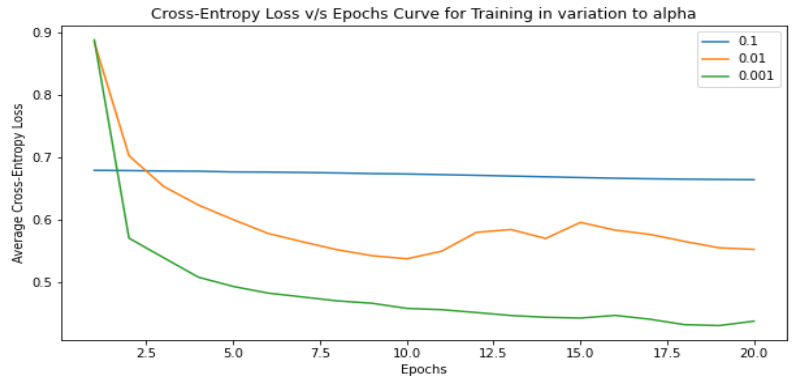
- **Variation in Alpha**

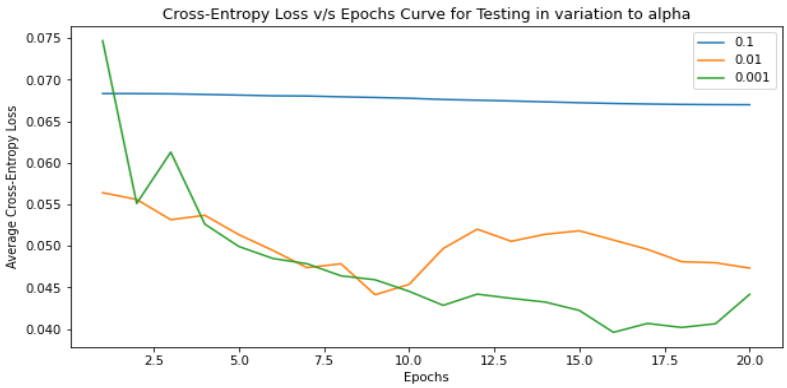


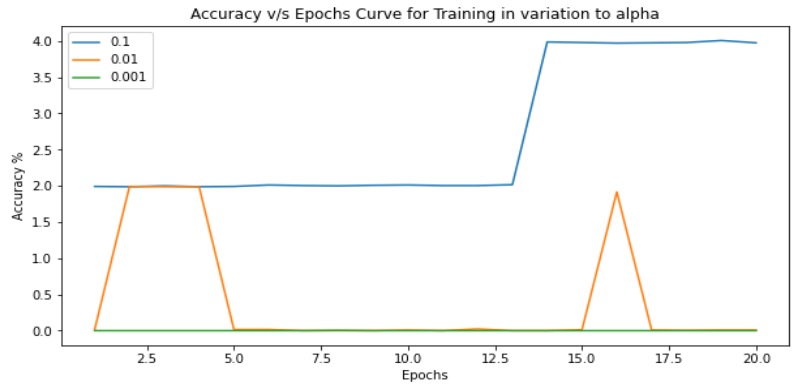


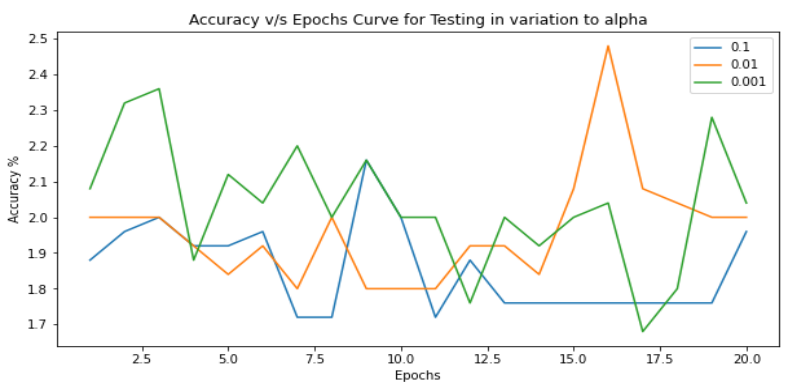




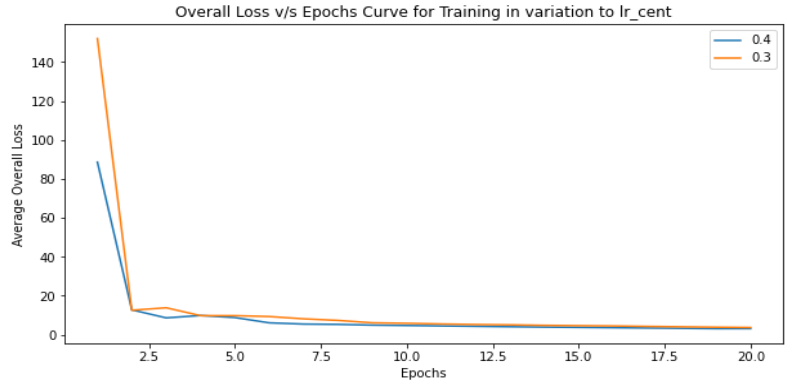


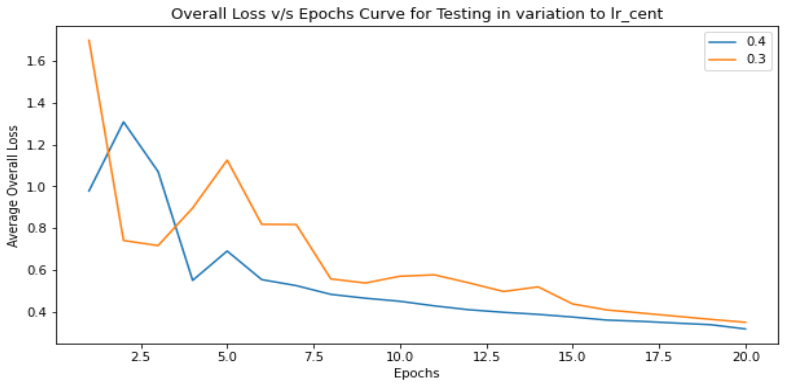


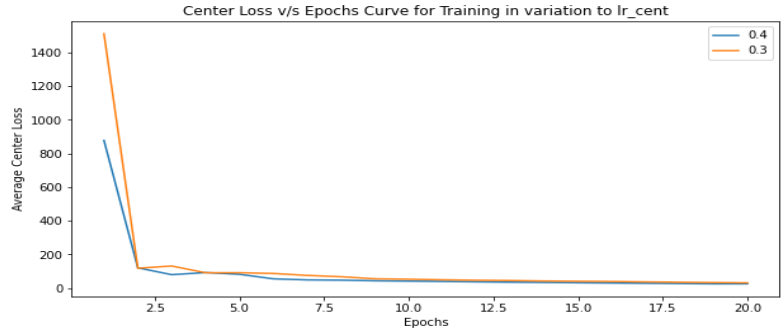


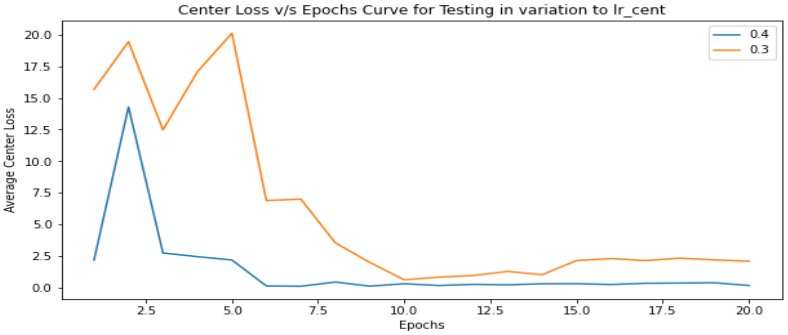


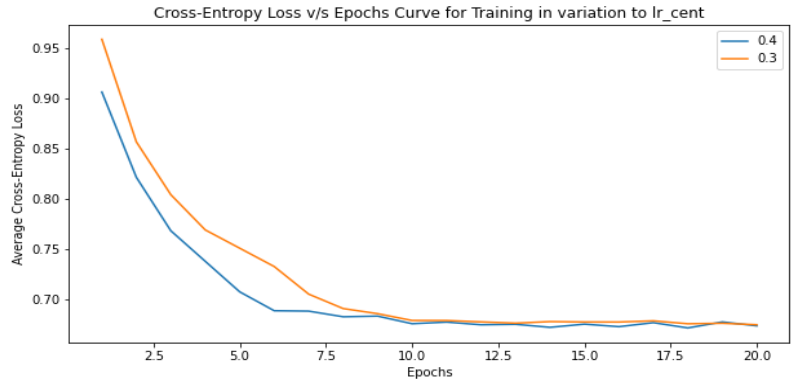
- **Variation in Center-Loss's Learning Rate**

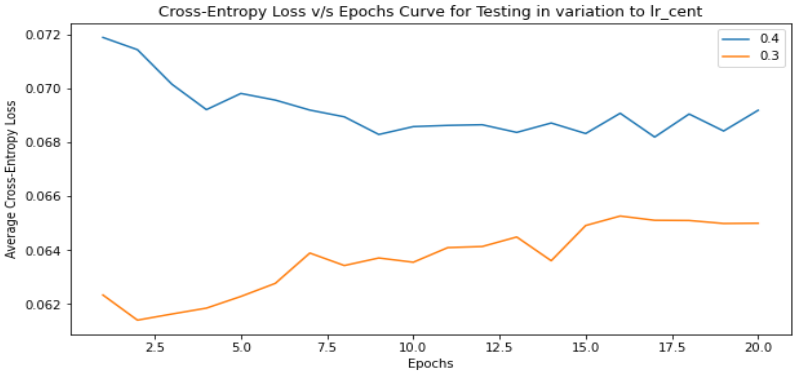


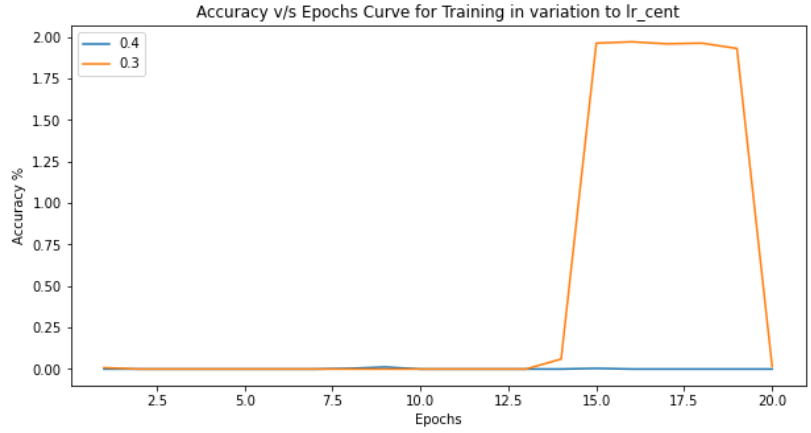


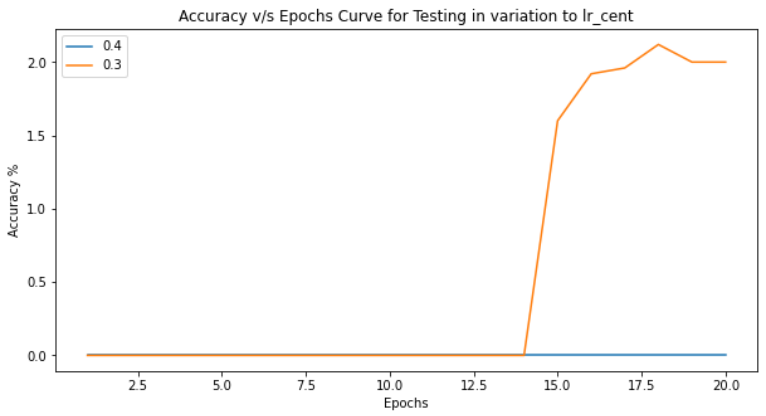










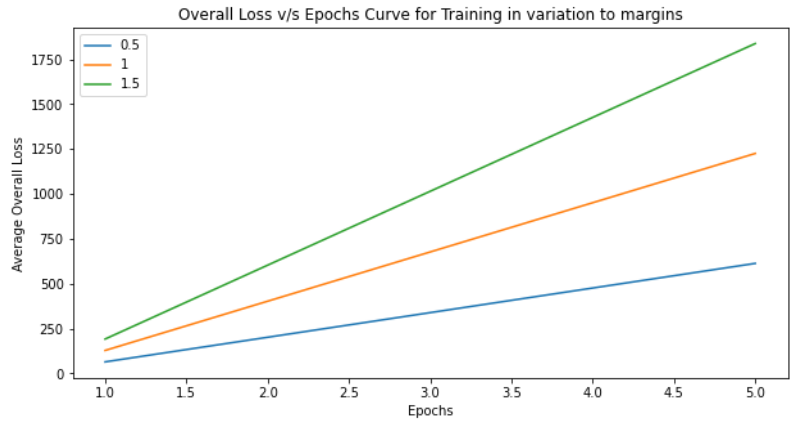


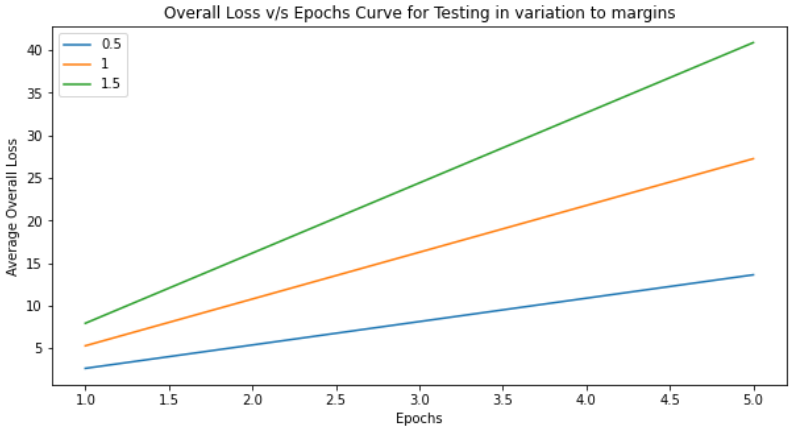
From the above graphs and the conceptual assumption of training a DL Model with thousands of epochs for the case of Center-Loss (For better clarity of Cluster Centers and Skewed Classes), with a decrease in alpha, losses converge quickly, but there's no clear trend visible for Accuracy Curvature. Also, for an increase in learning rate for center loss, losses converge quickly, but accuracy curvature dips down.

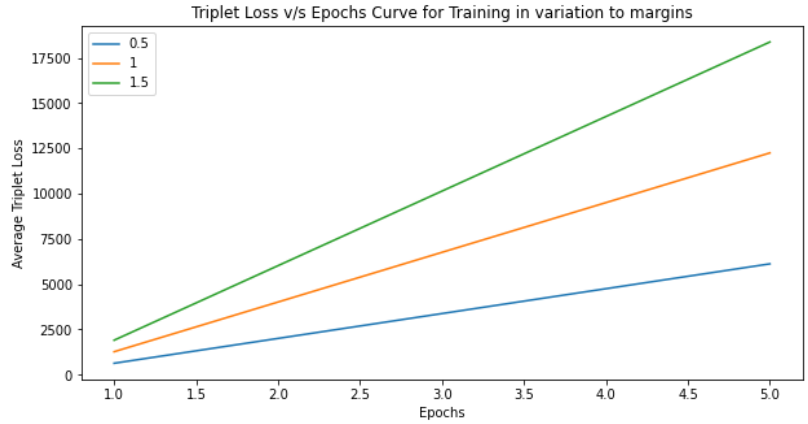
It's strange to see so much low Accuracy Metrics (2-5% in 15 epochs), despite Loss dipping down and converging according to the graphs. [Maybe an implementation mistake or something wrong in loss function's class]

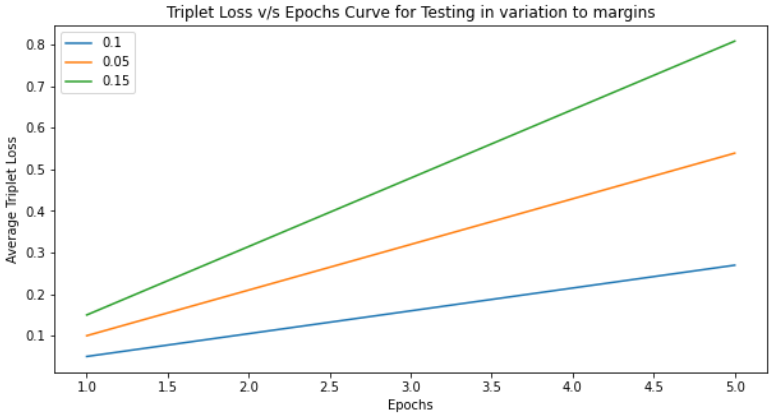
(c) Triplet Loss as final-classification-loss function (Complemented by Cross-entropy loss for label-wise classification)

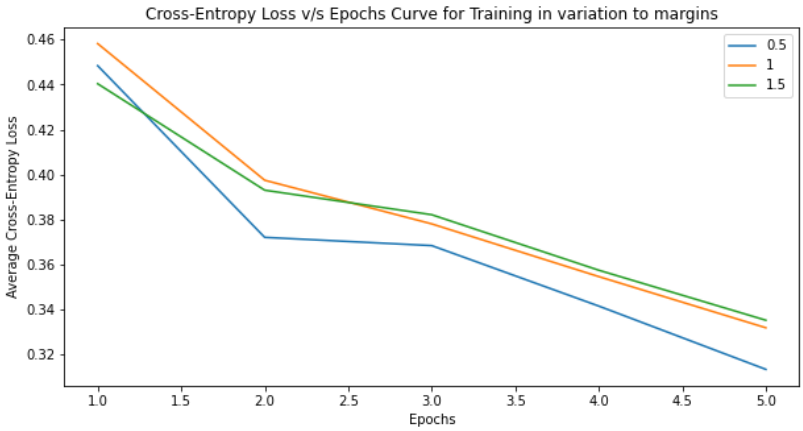
- **Variation in Margin**

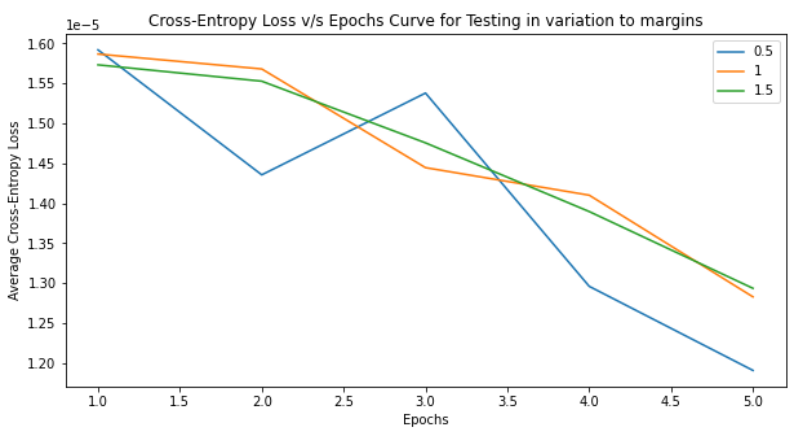


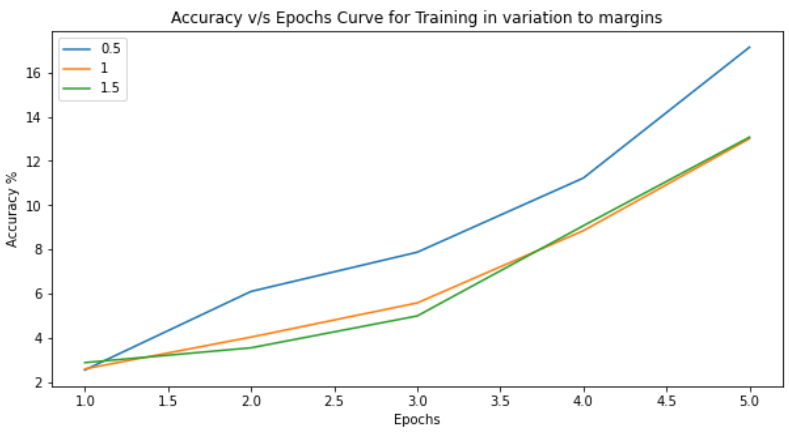


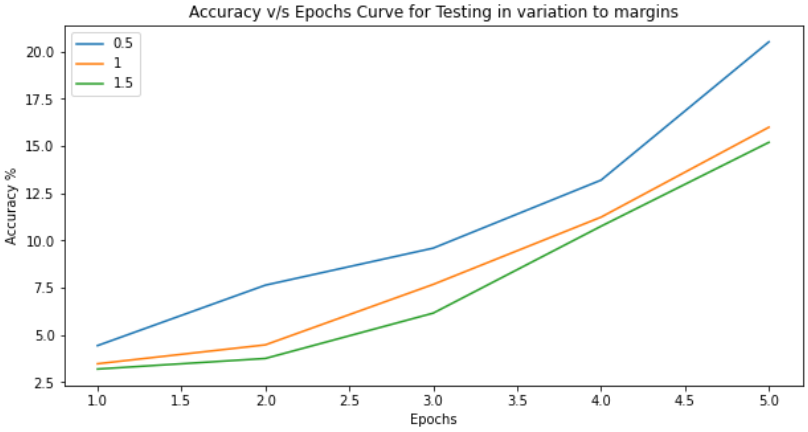












As a conclusive note, it can be said that Cross-Entropy Loss can work for smaller Epochs very well, but if Longer Epochs with more computational resources would be given, Center and Triplet Loss can easily outperform Cross-Entropy Loss.

Question-3

Implement a three-layer CNN network, for classification tasks for the Dogs vs. Cats dataset. [You can use the necessary libraries/modules]

a. Compare the accuracy on the test dataset (split into train and test [70:30]) for the following optimization techniques: [5+5+5+5 = 20 Marks]
   i. Vanilla SGD
   ii. Mini Batch SGD
   iii. Mini Batch with momentum
   iv. Mini Batch with Adam

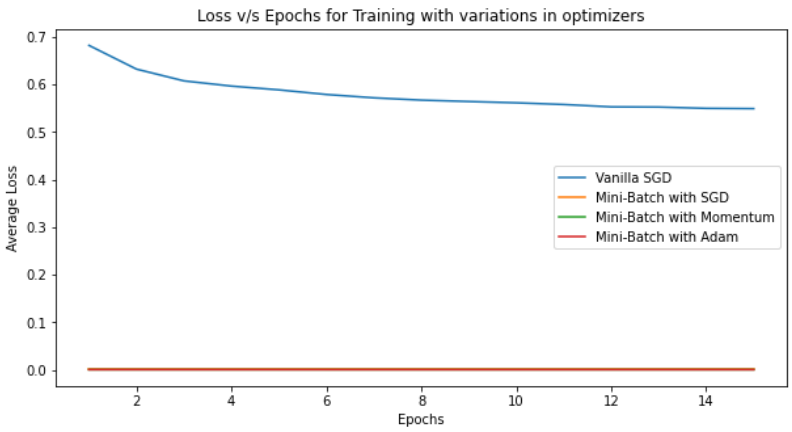
b. What are your preferred mini-batch sizes? Explain your choice with proper gradient update plots. [3 marks]
c. What are the advantages of shuffling and partitioning the mini-batches? [3 marks]

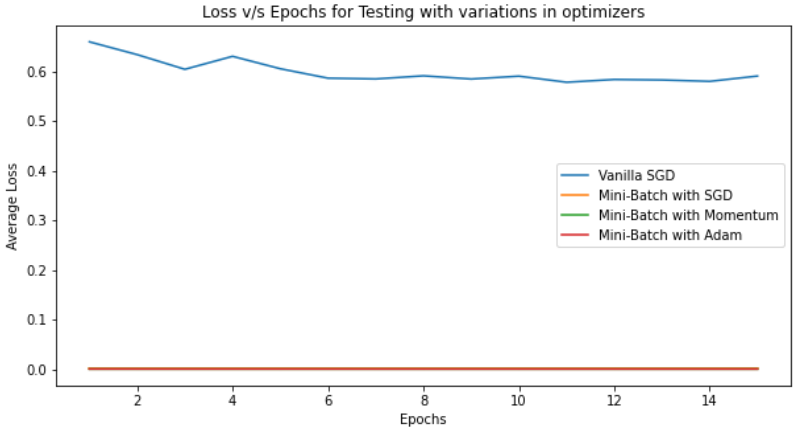
d. Explain the choice of beta (β) and how changing it changes the update. Explain the gradient update using plots. [3 marks]
e. Are there any advantages of using Adam over the other optimization methods? [3 marks]

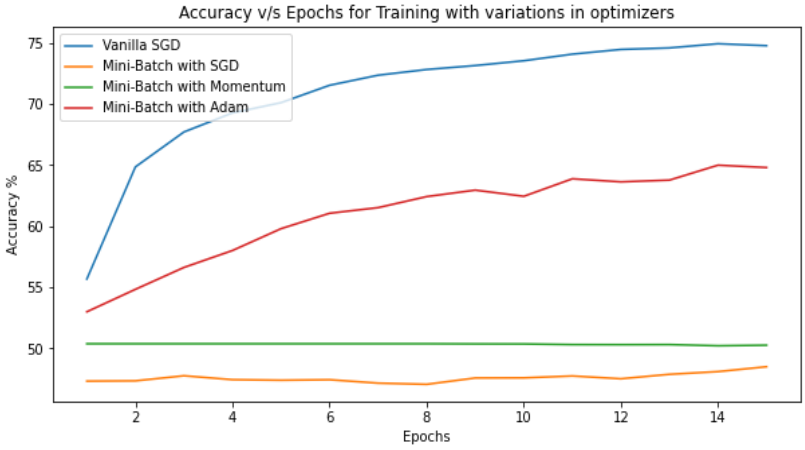
Soln-3

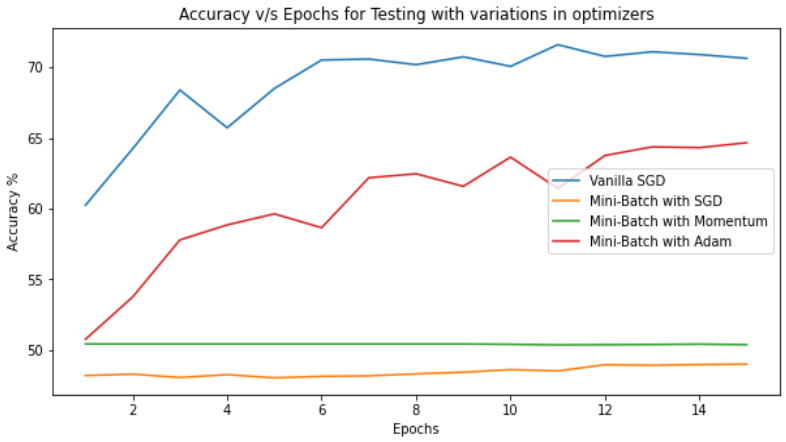
For the (a) subpart, various optimization methods were used while taking 70:30 splits of Cat v/s Dog Dataset and in other subparts as well, Loss v/s Epochs and Accuracy v/s Epochs graphs were plotted alongside necessary legends and scales taken as well.

(a) Variations in Optimization Techniques on the performance of CNN Architecture







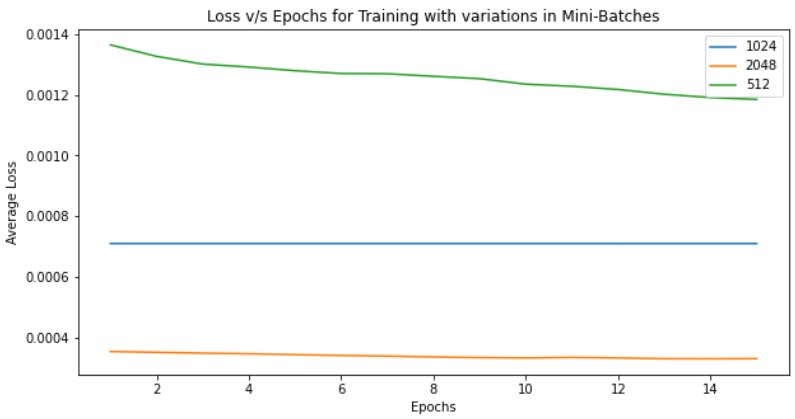


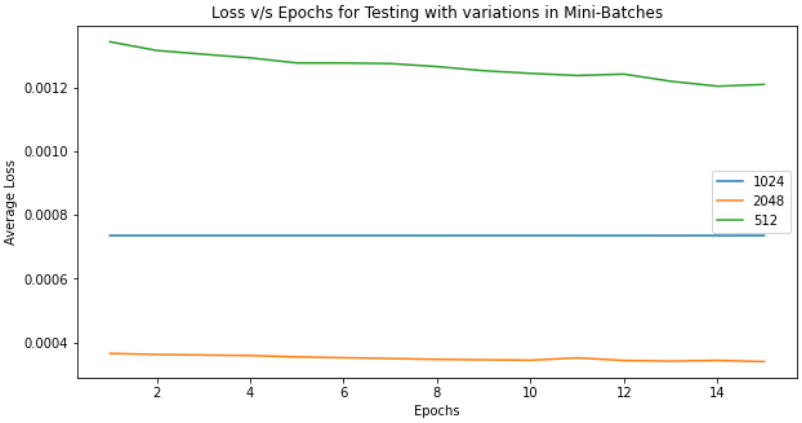
Although the time taken by Vanilla SGD (1 datapoint at a time) was the most amongst the other optimization methods, it outperformed other optimizers in terms of Accuracy (nearly 75% on training and 70% on testing). Here, the ability

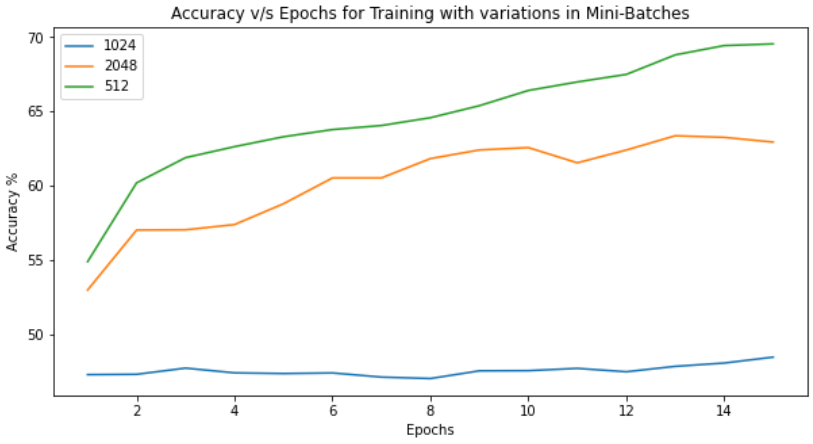
of an optimizer can't be judged only by the nature of Loss curvature, but the way an Accuracy curvature follows clearly indicates otherwise.

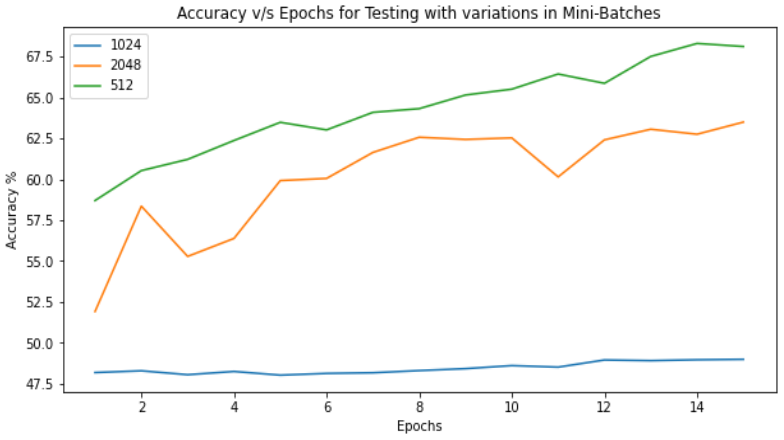
The reason behind this trend is because Vanilla SGD focuses on a single datapoint at a time (whether an outlier or a normal class-wise skewed point), which enables the model to work in a more generalizable manner when compared to other approaches, as they take the support of mini-batches, which sometimes lead to biases amongst groups of outliers, skewed points, and saddled plateaus.

(b) Variations in Mini-batches (512, 1024, 2048) on the performance of CNN Architecture.







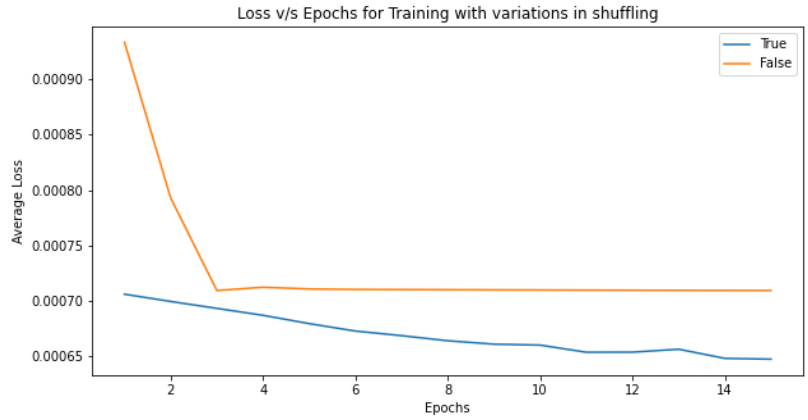


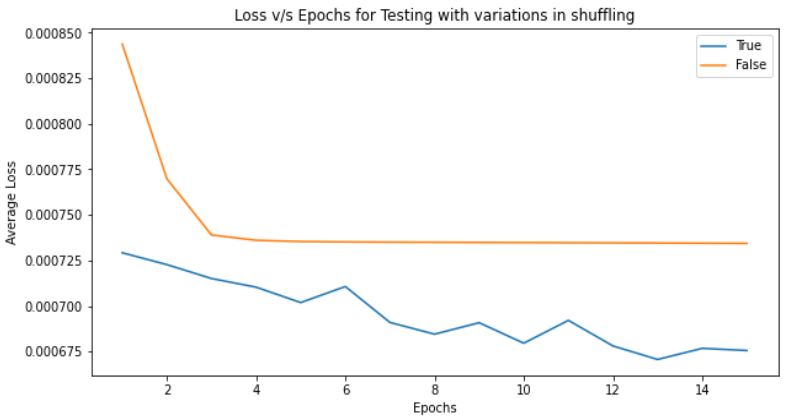
As clearly seen from the graphs, with the reduction in Batch-Size, Loss Increases, but Accuracy Increases as well.

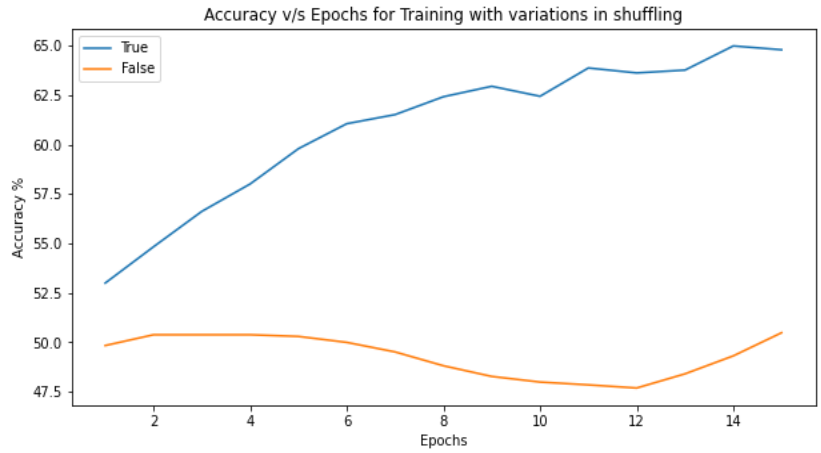
The reason behind this trend can be, because smaller batch-size will ensure that the model understands each and every data point to greater depth and enrich the feature maps in a more specific manner, thereby leading to generalizable resulting metrics.

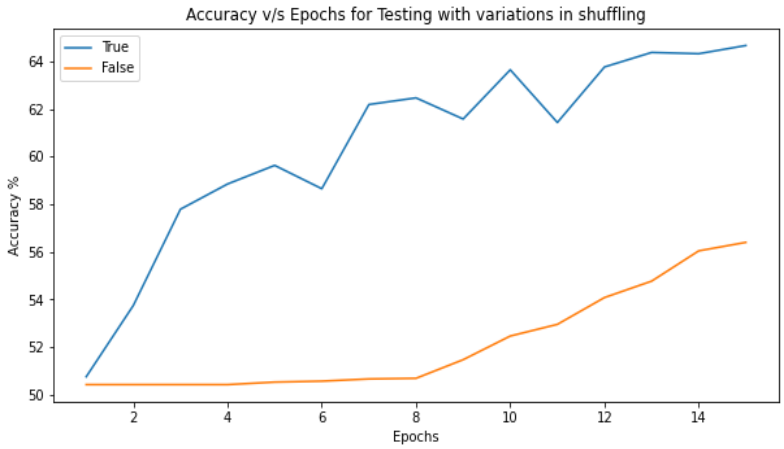
On the other hand, a bigger batch size will ensure less convergence and inference-time but thereby lead to a tradeoff in loss and accuracy metrics.

(c) Shuffling v/s Non-Shuffling on the performance of CNN Architecture





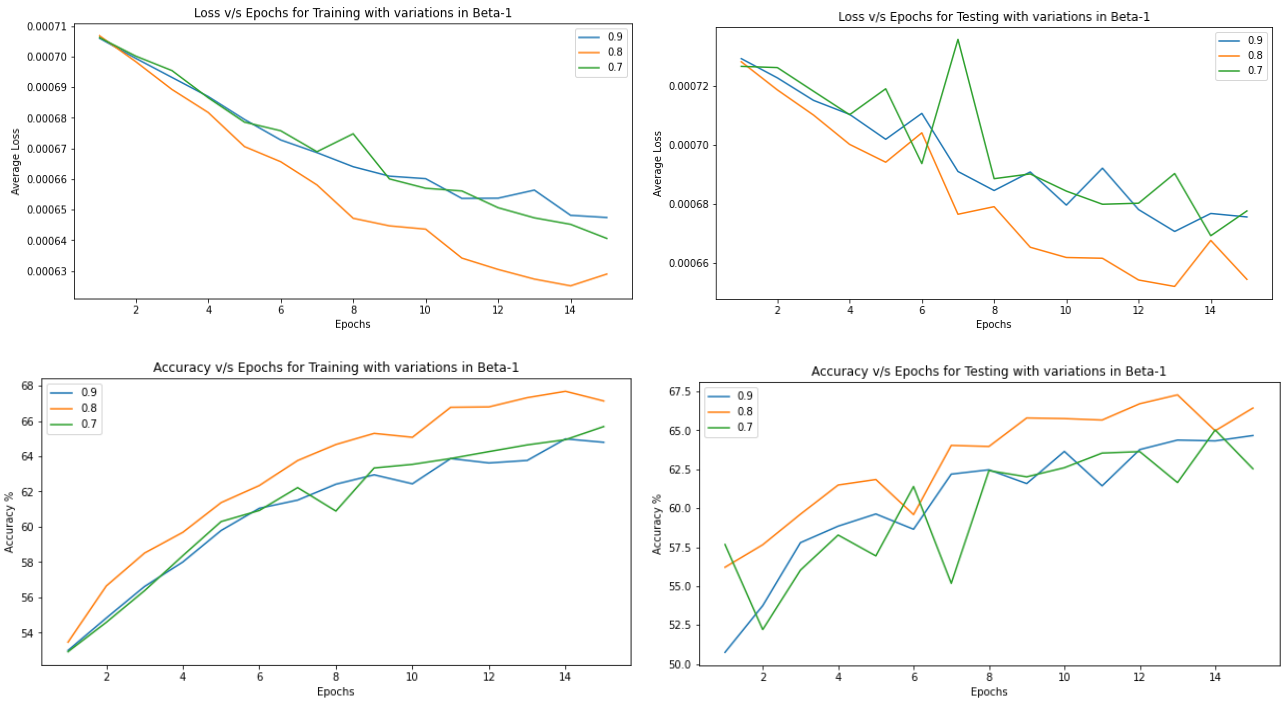




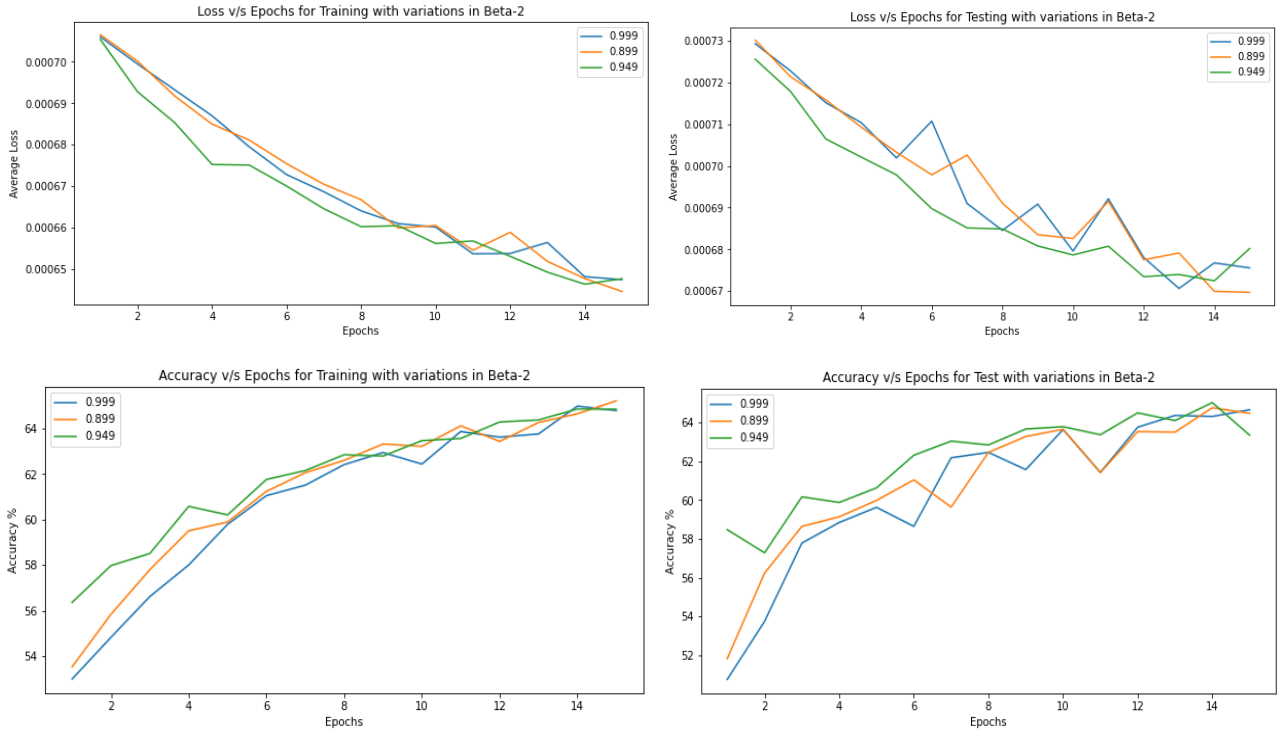
As clearly seen from the graph, shuffling the data points will lead to smoother convergence in loss curvature, as well as a drastic increase in Accuracy, when compared to the traditionally non-shuffling approach.

The reason behind this can be because outliers and skewed points together can be understood and analyzed by the model simultaneously when data points are made to shuffle. On the other hand, a non-shuffling approach will not only introduce biases amongst batches it would also influence the inability to notice the drastic changes in gradients of model parameters, thereby leading to stagnant learning.

(d-1) Variations in Beta-1 (Adam as optimizer) on the performance of CNN Architecture



(d-2) Variations in Beta-2 (Adam as optimizer) on the performance of CNN Architecture



Although there haven't been drastic changes observed while varying Beta-2, varying Beta-1 led to drastic jumps in Accuracy Curvature, as the reduction in Beta-1 led to increase in accuracy.

For the scenario of Beta-2, it can be said in a specific manner that reduction in Beta-2 leads to smoother convergence in Loss Curvature, but not that many changes in Accuracy Curvature.

(e) **Using Adam and its variants as an optimizer, when compared to other types of optimizers, can / cannot be better, because it totally depends on the use case and the distribution of data points.**

Also, if the Loss Curvature in multi-dimensional feature space constitutes several saddle points and flattened plateaus, for those cases it cannot perform better. But for Loss Curvature with multiple minima (global as well as local), Adam can work great.

References

Question-1

https://jimmy-shen.medium.com/pytorch-freeze-part-of-the-layers-4554105e03a6#:~:text=In%20PyTorch%20we%20can%20freeze,to%20apply%20a%20pretrained%20model.

https://gist.github.com/L0SG/2f6d81e4ad119c4f798ab81fa8d62d3f

https://blog.ineuron.ai/AUC-ROC-score-and-curve-in-multiclass-classification-problems-2ja4jOHb2X

https://github.com/pytorch/vision/issues/48

StackOverflow and Pytorch Documentations

Question-2

https://github.com/KaiyangZhou/pytorch-center-loss

https://medium.com/the-owl/extracting-features-from-an-intermediate-layer-of-a-pretrained-model-in-pytorch-c00589bda32b

StackOverflow Documentations

Question-3

https://github.com/amitrajitbose/cat-v-dog-classifier-pytorch/blob/master/train.py

https://towardsdatascience.com/understanding-pytorch-with-an-example-a-step-by-step-tutorial-81fc5f8c4e8e

StackOverflow and Geeks-For-Geeks Documentations