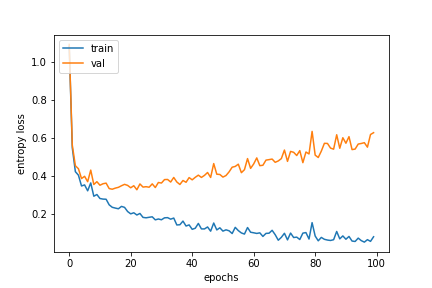
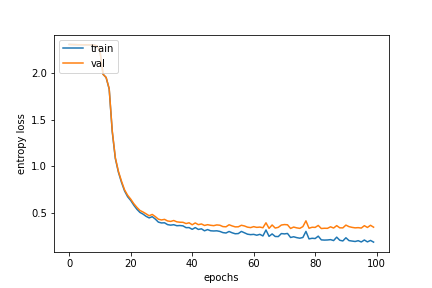
5.b

Relu:

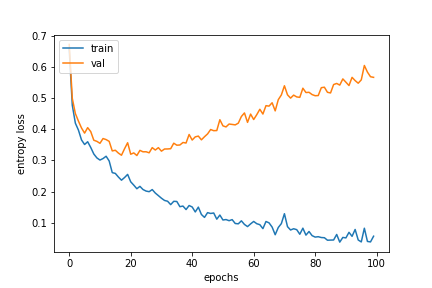
**Loss vs epoch**

Sigmoid:

**Loss vs epoch**

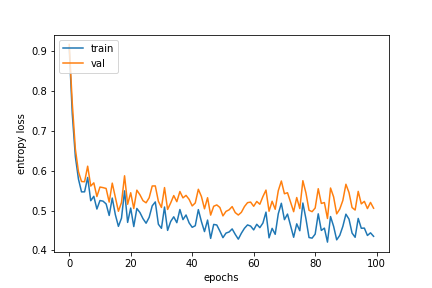
tanh:

**Loss vs epoch**



linear:

**Loss vs epoch**



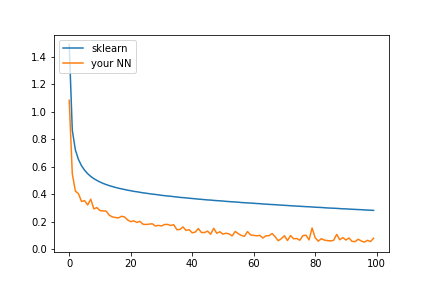
5.c

|  |  |  |
| --- | --- | --- |
| Accuracy | Iteration | Model |
| 0.888 | 50 | Relu |
| 0.886 | 100 | Relu |
| 0.83 | 50 | Linear |
| 0.832 | 100 | Linear |
| 0.871 | 50 | Sigmoid |
| 0.886 | 100 | Sigmoid |
| 0.891 | 50 | Tanh |
| 0.887 | 100 | Tanh |

6.

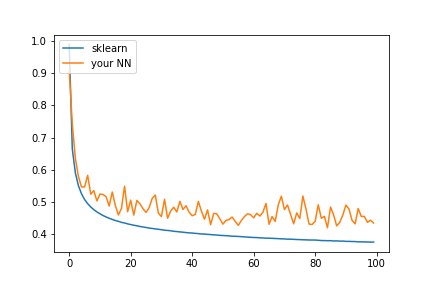
RELu

**Loss vs epoch**



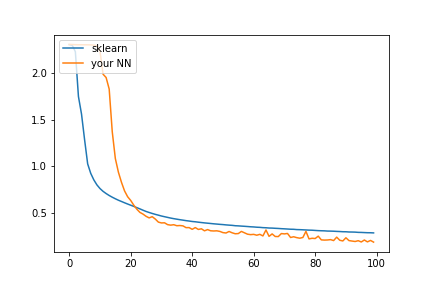
Linear

**Loss vs epoch**



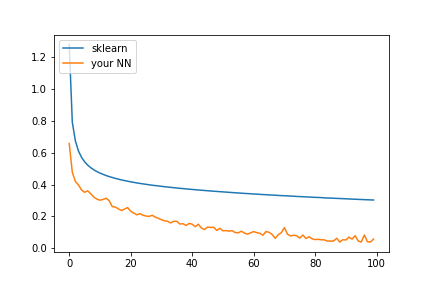
Sigmoid

**Loss vs epoch**



Tanh

**Loss vs epoch**



Learning rate of sigmoid was 0.02( as it’s a slow learner). Rest had 0.001.

|  |  |  |
| --- | --- | --- |
| Accuracy | Iteration | Scikit’s Model |
| 0.873 | 50 | Relu |
| 0.888 | 100 | Relu |
| 0.861 | 50 | Linear |
| 0.865 | 100 | Linear |
| 0.857 | 50 | Sigmoid |
| 0.889 | 100 | Sigmoid |
| 0.872 | 50 | Tanh |
| 0.891 | 100 | Tanh |

Tanh with 100 epochs is the best model in both sci kit implementation and my implementation in terms of accuracy.

Q2

Preprocessing:

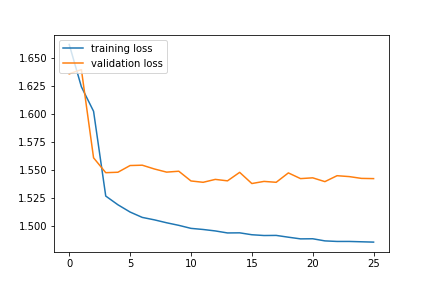
applychanges = biglight.transforms.Compose([biglight.transforms.Resize(I applied these 4 transformations on dataset.

#  Resize resizes the image to 224X224

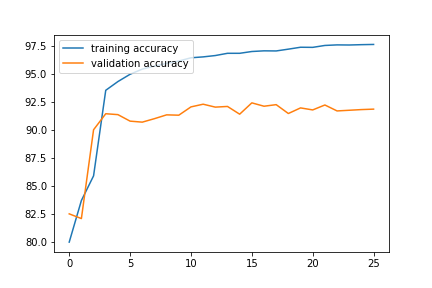
#   converts the image to greyscale

#   converts the image/np array to tensor

#   normalizes the images using mean and std. of imagenet dataset. 3 means because of 3 colors: R,G,B

**Loss vs epochs graph** 

**Accuracy vs epochs graph**



test set accuracy:0.920500002304713

[[562. 0. 13. 18. 1. 0. 61. 0. 3. 0.]

[ 0. 601. 0. 8. 2. 0. 0. 0. 0. 0.]

[ 7. 1. 549. 4. 14. 0. 35. 0. 2. 0.]

[ 7. 3. 2. 530. 11. 0. 9. 0. 0. 0.]

[ 3. 1. 20. 29. 517. 0. 37. 0. 0. 0.]

[ 0. 0. 0. 1. 0. 591. 0. 13. 0. 8.]

[ 54. 0. 27. 27. 12. 0. 477. 0. 3. 0.]

[ 0. 0. 0. 0. 0. 2. 0. 518. 0. 17.]

[ 0. 0. 0. 2. 0. 1. 5. 0. 590. 0.]

[ 0. 0. 0. 0. 0. 3. 0. 11. 0. 588.]]

for class 0

accuracy:0.8541033434650456

confusion matrix

[[ 562. 96.]

[ 71. 5271.]]

for class 1

accuracy:0.983633387888707

confusion matrix

[[6.010e+02 1.000e+01]

[5.000e+00 5.384e+03]]

for class 2

accuracy:0.8970588235294118

confusion matrix

[[ 549. 63.]

[ 62. 5326.]]

for class 3

accuracy:0.9430604982206405

confusion matrix

[[ 530. 32.]

[ 89. 5349.]]

for class 4

accuracy:0.8517298187808896

confusion matrix

[[ 517. 90.]

[ 40. 5353.]]

for class 5

accuracy:0.964110929853181

confusion matrix

[[ 591. 22.]

[ 6. 5381.]]

for class 6

accuracy:0.795

confusion matrix

[[ 477. 123.]

[ 147. 5253.]]

for class 7

accuracy:0.9646182495344506

confusion matrix

[[ 518. 19.]

[ 24. 5439.]]

for class 8

accuracy:0.9866220735785953

confusion matrix

[[ 590. 8.]

[ 8. 5394.]]

for class 9

accuracy:0.9767441860465116

confusion matrix

[[ 588. 14.]

[ 25. 5373.]]

Class 6 has least accuracy. My model is able to predict class 8 with best accuracy. Many times my model predicts Class 6 even though actual class is 0.Also, many times when actual class is 6, my model predicts it as 0.

Q3

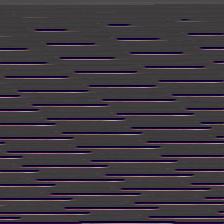
a)

initial pic: 

after 5 iteration:



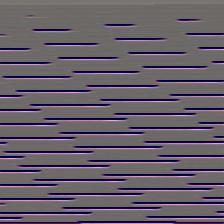
after 10 iteration:



after 15 iteration:



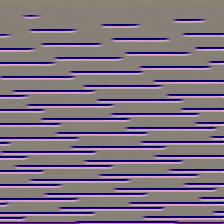
after 20 iteration:



after 25 iteration:



After 30 iteration:

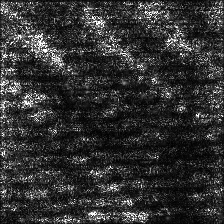


Observation: images are becoming more bright.

b)



These are grey gradients of the same cat image.



These are colored gradients.

c)Pics of a random image for each convolution layer of my model is in Q3c\_generatedimages folder.