REPORT for NVIDIA Fundermentals of Deep learning and computer vision Certification course

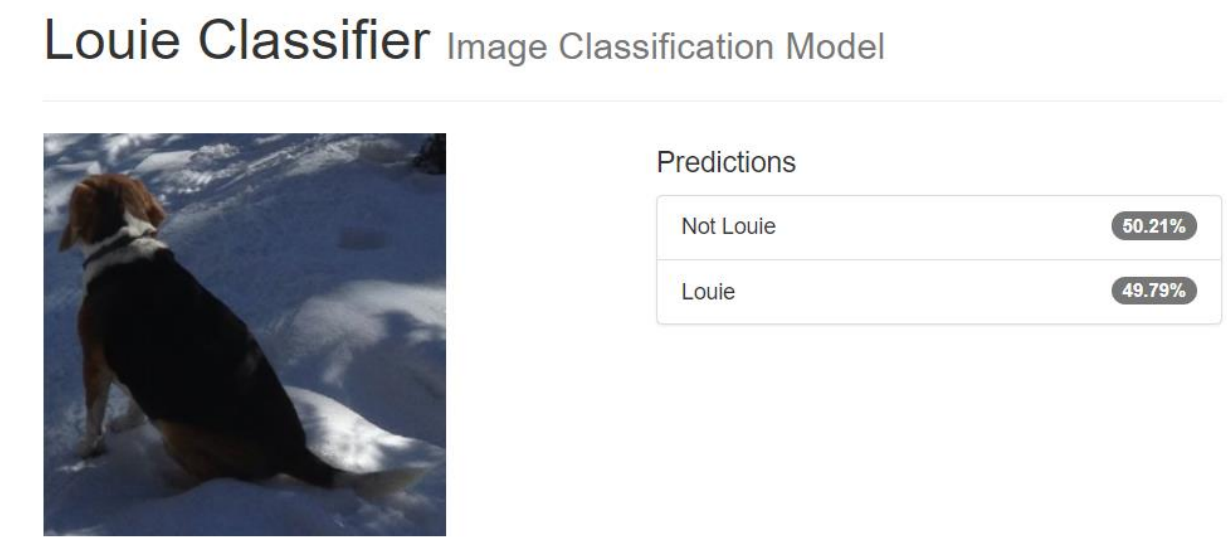
By- Jai Soni

TASK -1

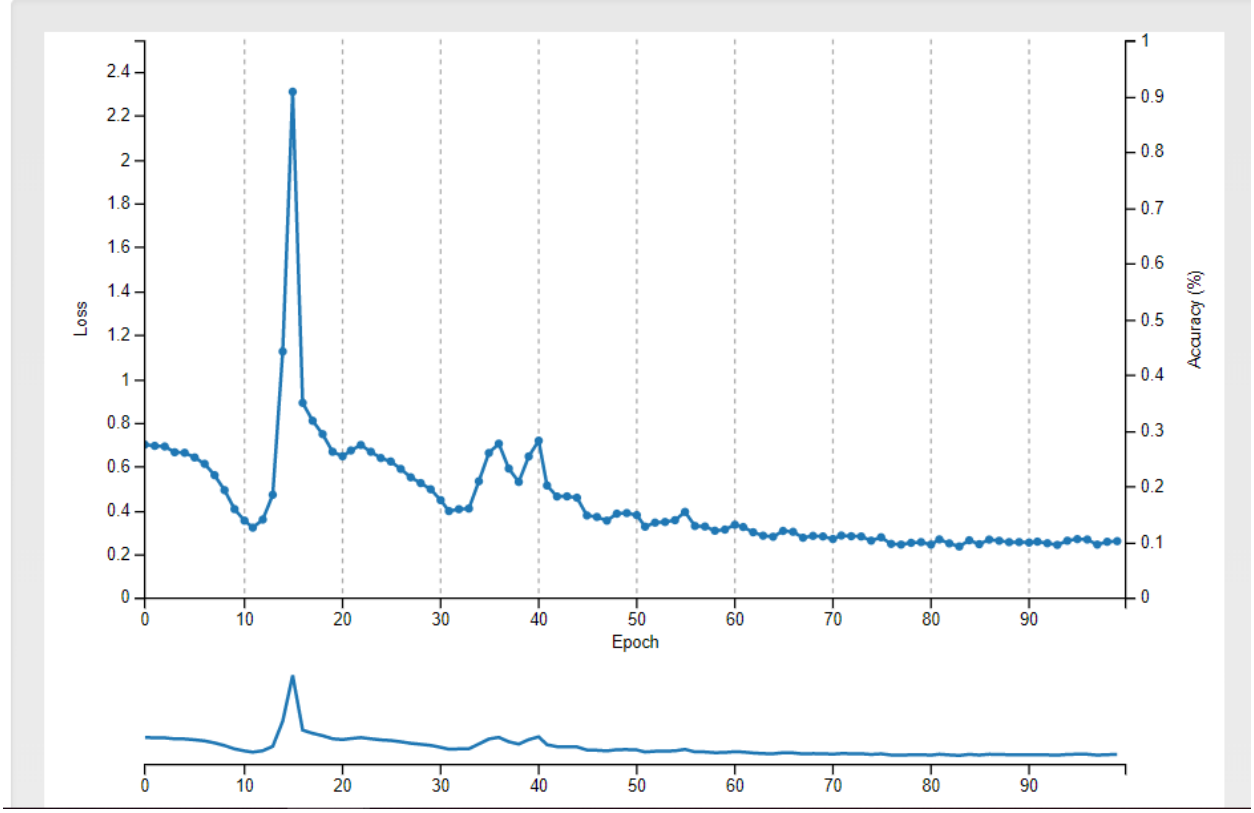
Model 1

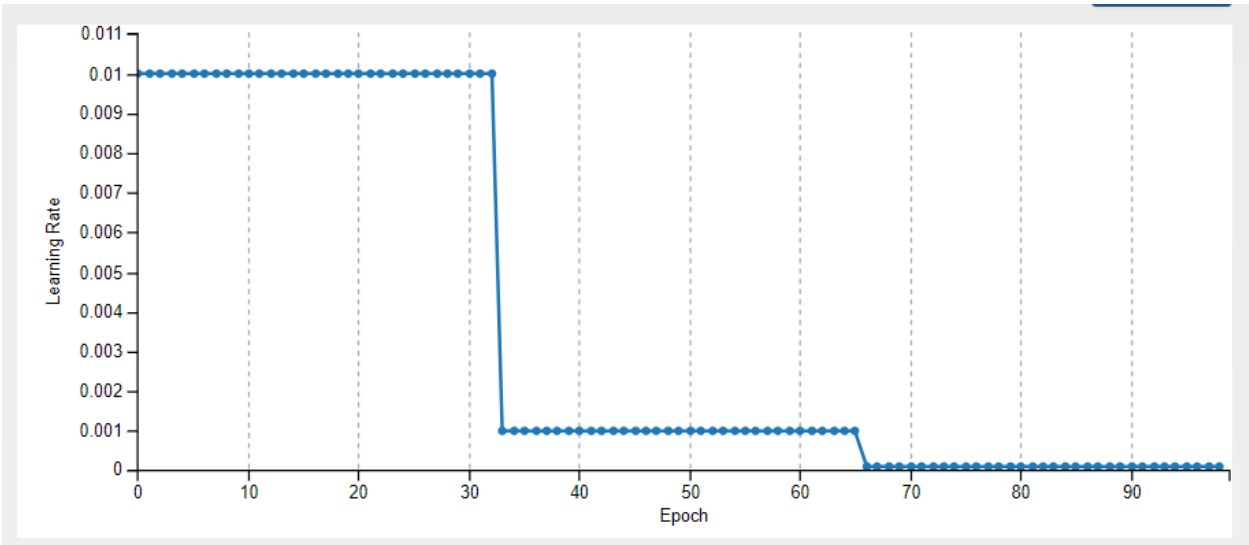
Training a 2 epoch AlexNet model with 16 images and identifying the test is Louie or not

Result 50:50



Training a 100 epoch model with same 16 images and identifying the test image is louie or not


Loss
Accuracy (%)
Epoch
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
0.1
0.3
0.5
0.7
0.9
10
20
30
40
50
60
70
80
90


Learning Rate
Epoch
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
0.011
10
20
30
40
50
60
70
80
90

Louie Classifier after 100 epochs Image Classification Model

Predictions

Louie	100.0%
Not Louie	0.0%

Louie Classifier after 100 epochs Image Classification Model

Predictions

Not Louie	98.69%
Louie	1.31%

Result 100% louie, as the model is overfitted and memorizes the pictures it has seen before, instead of learning important features of louie from the data. When a different image of louie is provided, This model is unable to identify whether its louie or not

TASK -1

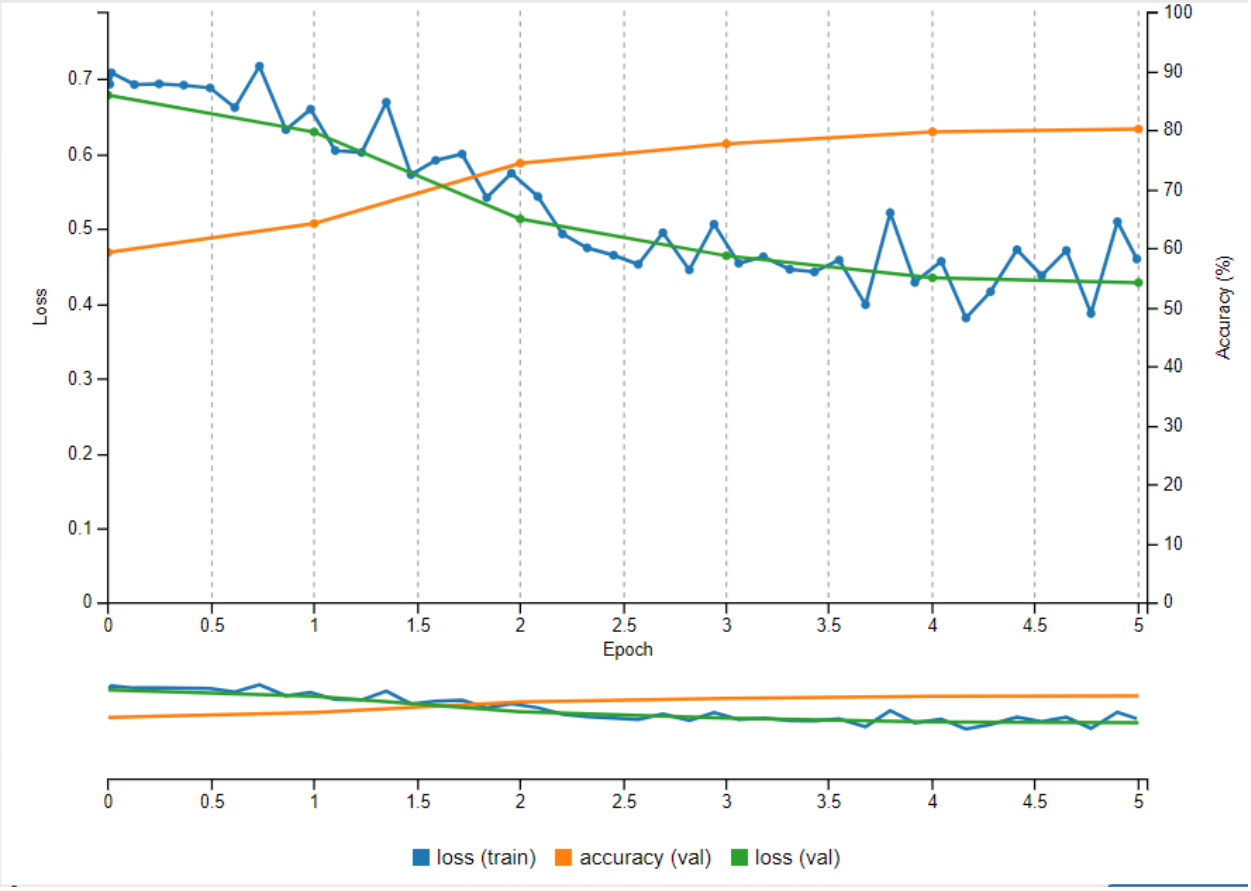
Model 3:

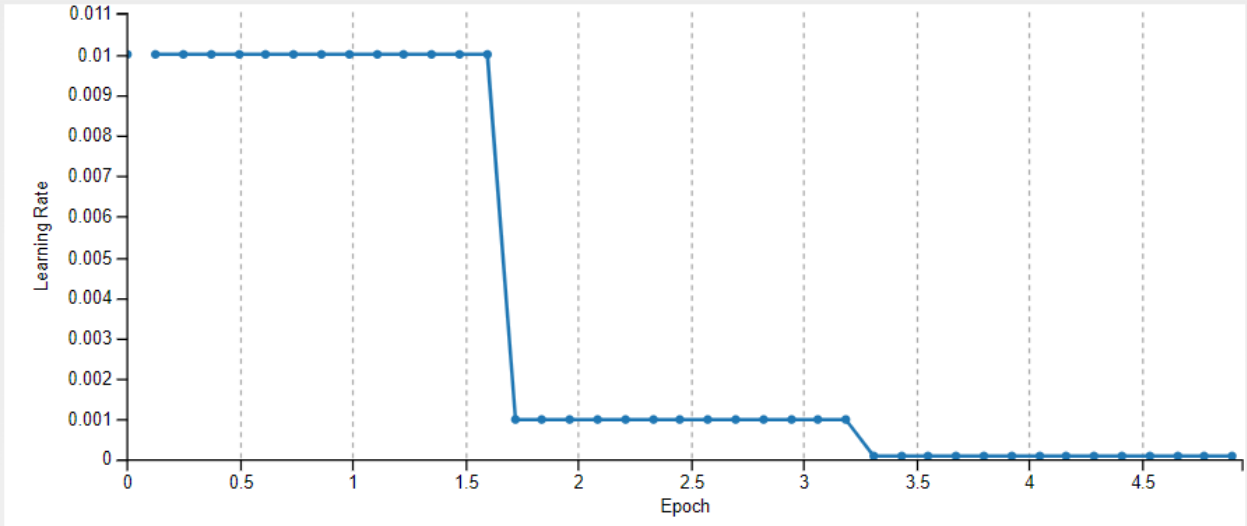
Image size 256 X 256,

Cats Vs Dogs, Kaggle data set

75 % Training, 25 % validation

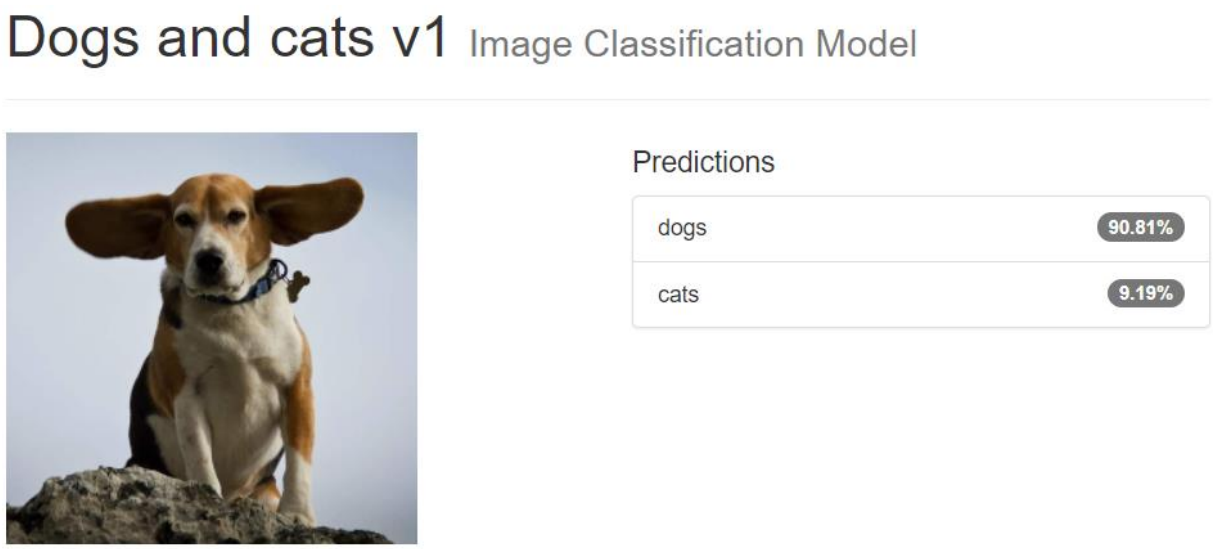
Alex Net, 5 Epochs





Learning Rate? :

Results: When presenting with a unlabled image of Louie(Dog), the model is able to correctly classify it as dog.



Accuracy ~ 80%

Loss (Val)~ 0.5

Loss(Train)~ Fluctuating

TASK -3

Deployment of the model

Steps:

1. Load the model architecture save in .prototxt file and model weights in .caffemodel file

2.using caffé to build a model with the provided architecture and weights

“

```
# Initialize the Caffe model using the model trained in DIGITS
net = caffe.Classifier(ARCHITECTURE, WEIGHTS,
                       channel_swap =(2, 1, 0), #Color images have three channels, Red, Green, and Blue.
                       raw_scale=255) #Each pixel value is a number between 0 and 255
                       #Each "channel" of our images are 256 x 256
```

”

3. Preprocess expected input - To create an Expected Input, we need to preprocess the input in the same way as we did it for training the model.

4. Use your model to predict a test image

“

make prediction

prediction = net.predict([ready_image])

print prediction

”

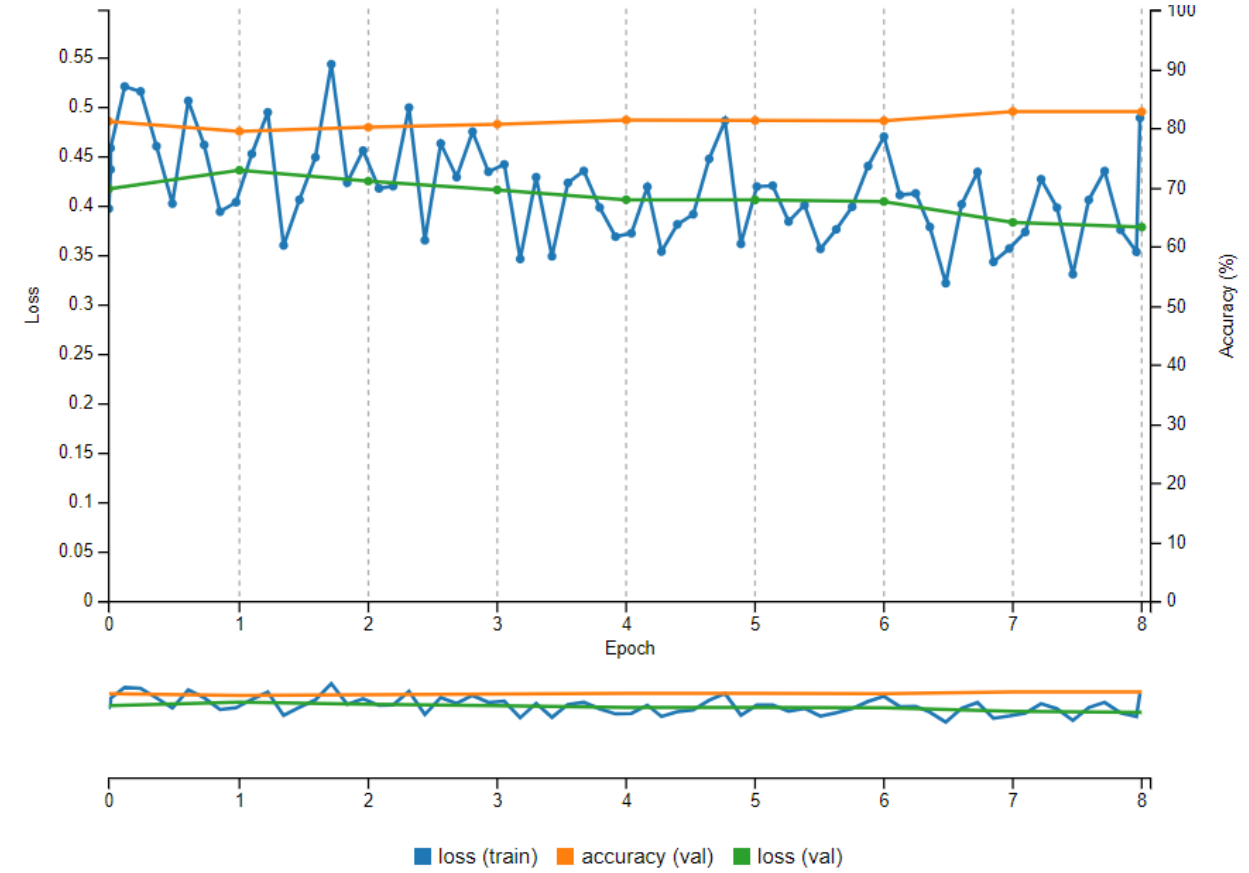
5. Use this process to take an image as input and predict the output using the model, .py file or a .sh file can be used for the same

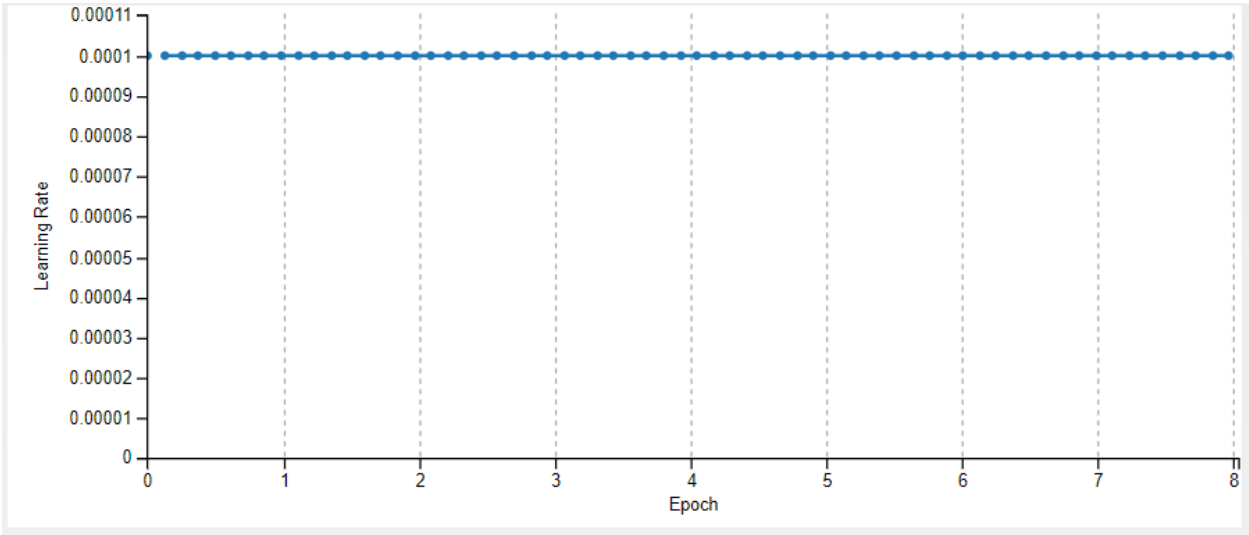
TASK -4

The learning rate decreases throughout the training session because the network is getting closer to its ideal solution. At first, the network being trained knows *nothing* about what it may see.

Starting from a pretrained model dogs vs cats made earlier, that provided around 80 % accuracy, we increase the accuracy to 8 , so that the total epochs is equivalent to 13.

Instead of having a step down kind of learning rate we fix the learning rate to 0.0001.





We can further improve the model performance by tweaking the Hyperparameters, learning rate , model architecture and curating the data as required.

TASK -5

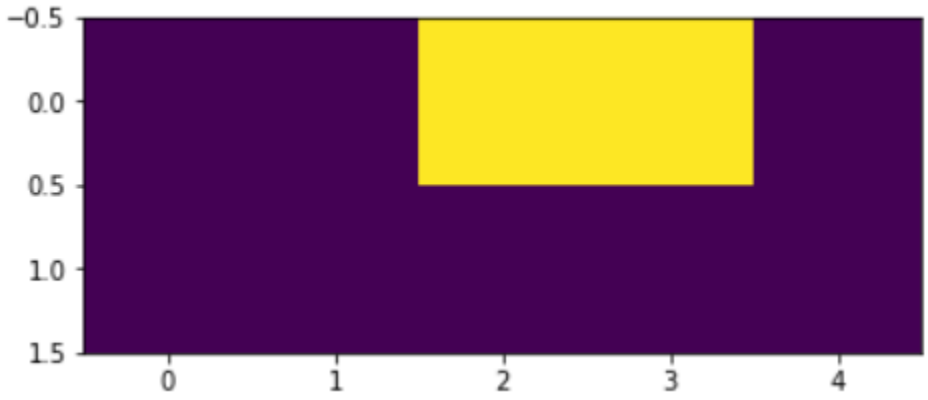
OBJECT DETECTION

Sliding window approach:

Sliding window approach involves deploying an image classifier trained to classify 256X256 portions of an image as either "dog" or "cat" to an application that moves across an image one 256X256 section at a time

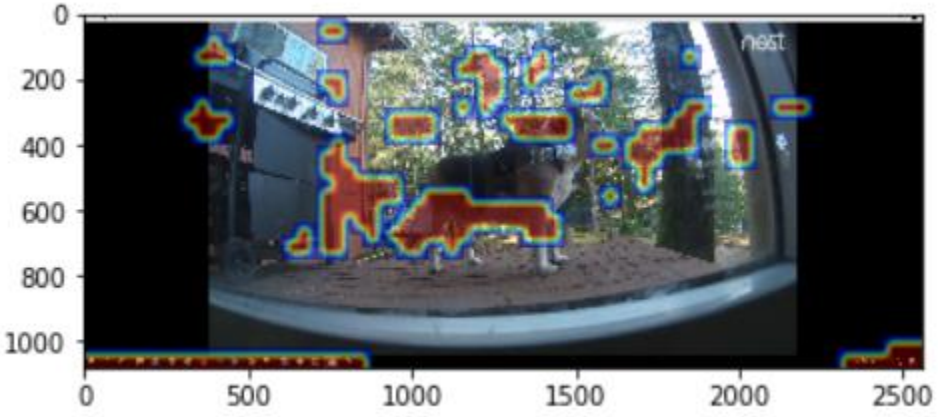
Input file:

Output File:



Sliding window block is cutting our input image into 256X256 squares, running each of them through our dog classifier, and creating a new image that is purple where there is not a dog and yellow where there is one, as determined by our classifier

Rebuilding from an existing neural network

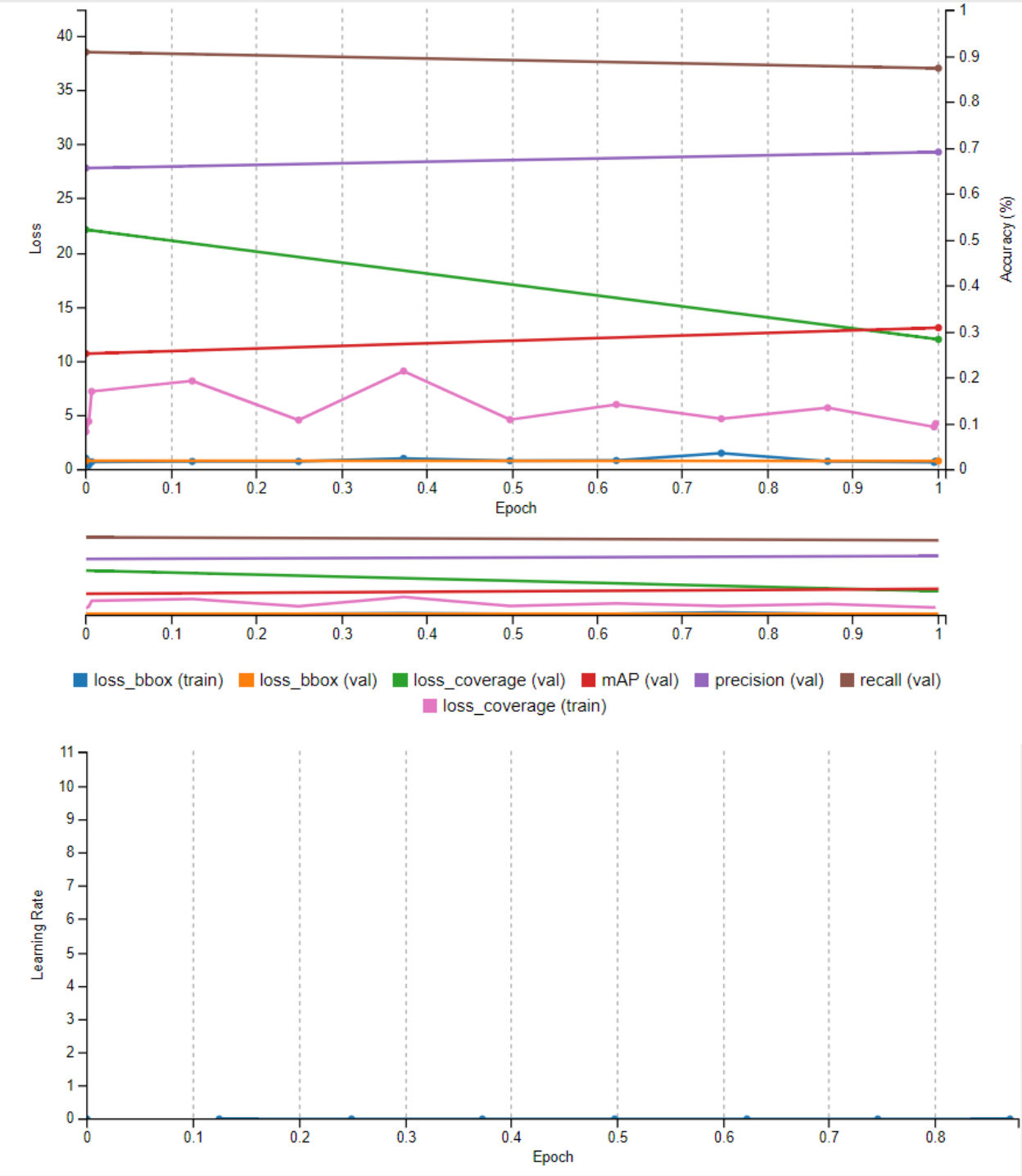


DetectNet

Bounding Box

The MeanAverage Precision (mAP) is a combined measure of how well the network can detect the dogs and how accurate its bounding box (localization) estimates were for the validation dataset.

Training a FCN with pretrained model weights for 1 epoch and a small learning rate of 1e-100


Loss
Accuracy (%)
Epoch
loss_bbox (train)
loss_bbox (val)
loss_coverage (val)
mAP (val)
precision (val)
recall (val)
loss_coverage (train)
Learning Rate
Epoch

Test:

Input:

Output:

Inference visualization

bbox-list [[158. 466. 296. 639. 2.99609375] [0. 0. 0. 0. 0.]]

Only places that has dog has number >0

Key Takeaways from the exercise:

There are a few key takeaways here:

1. The right network and data for the job at hand are vastly superior to hacking your own solution.
2. The right network (and sometimes even pretrained model) might be available in DIGITS or on the Model Zoo of your framework.

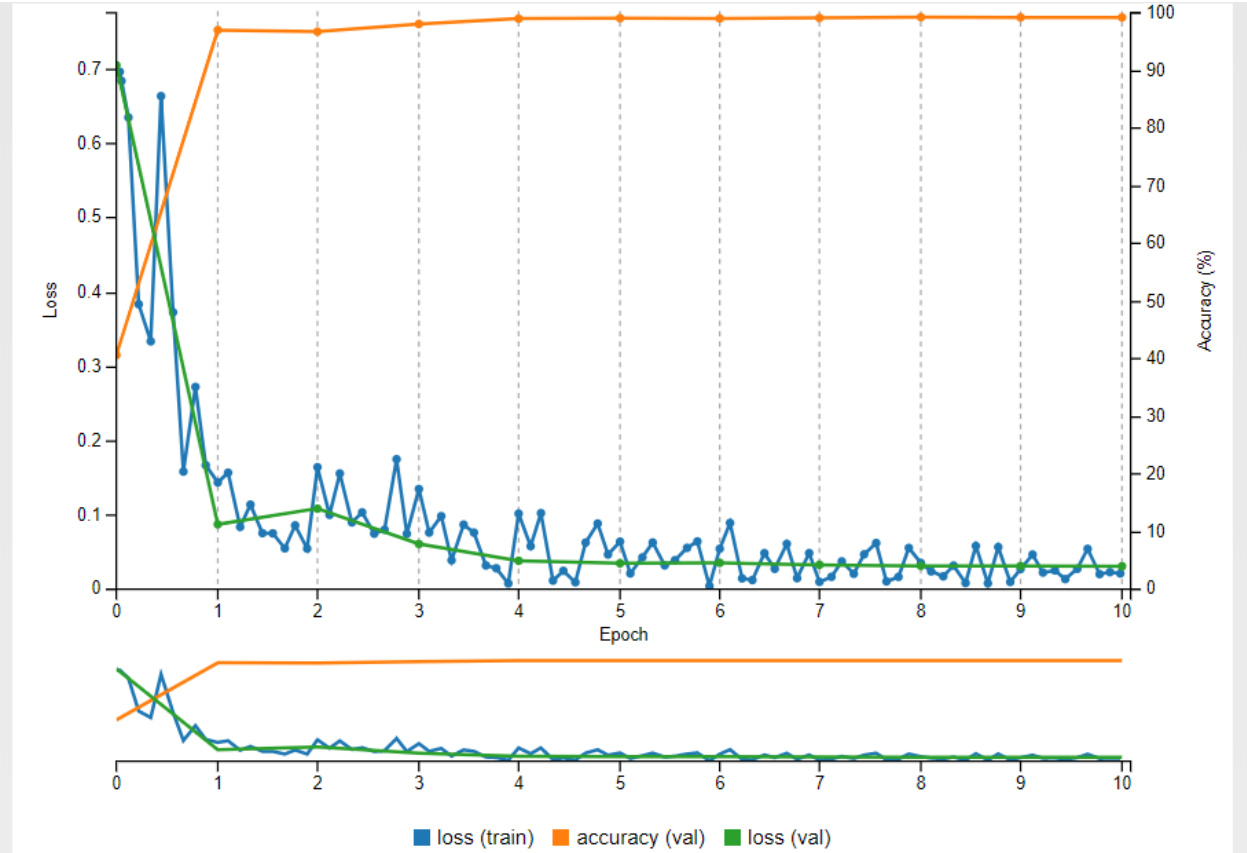
So one problem solving framework you can use is:

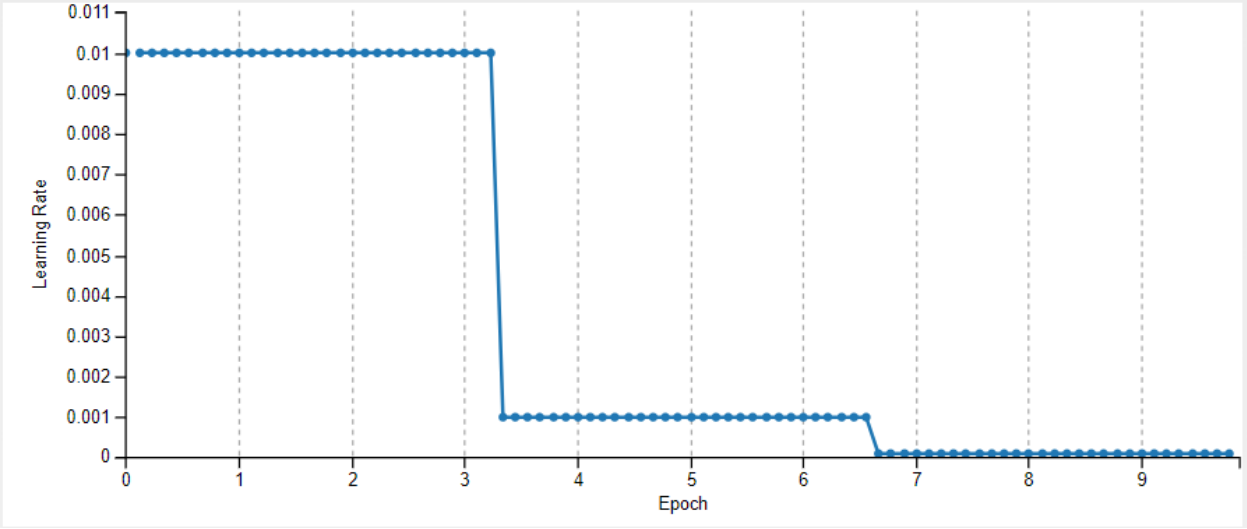
- determine if someone else has already solved your problem, use their model
- if they have not, determine if someone else has already solved a problem *like* yours, use their network and your data
- if they have not, determine if you can identify the shortcomings of someone else's solution in solving your problem and work to design a new network
- if you can not, use an existing solution other problem solving techniques (eg. python) to solve your problem
- either way, continue to experiment and take labs to increase your skill set!

Assessment:

Face vs No Face:

Trained a model with 10 epochs and 0.001 learning rate to get model accuracy >95%


Loss
Accuracy (%)
Epoch
loss (train)
accuracy (val)
loss (val)


Learning Rate
Epoch

face vs no face Image Classification Model

Predictions

not face	93.98%
face	6.02%

NVIDIA DEEP LEARNING INSTITUTE
CERTIFICATE OF COMPETENCY

This certificate is awarded to
JAI SONI

for demonstrating competence in the completion of
FUNDAMENTALS OF DEEP LEARNING FOR COMPUTER VISION

Will Ramey
Senior Director, Developer Programs, NVIDIA

2019
Year issued

NVIDIA DEEP LEARNING INSTITUTE

Year issued2019