Q3: Even deeper! Resnet18 for PASCAL classification (15 pts)

Hopefully we all got much better accuracy with the deeper model! Since 2012, much deeper architectures have been proposed. ResNet (https://arxiv.org/abs/1512.03385) is one of the popular ones. In this task, we attempt to further improve the performance with the “very deep” ResNet-18 architecture.

3.1 Build ResNet-18 (1 pts)

Write a network modules for the Resnet-18 architecture (refer to the original paper). You can use `torchvision.models` for this section, so it should be very easy!

In [ ]:
```
import torch
import torch.nn as nn
import torch.nn.functional as F
from torchvision import models
import matplotlib.pyplot as plt
%matplotlib inline

import trainer
from utils import ARGS
from simple_cnn import SimpleCNN
from voc_dataset import VOCDataset
```

3.2 Add Tensorboard Summaries (6 pts)

You should've already written tensorboard summary generation code into `trainer.py` from q1. However, you probably just added the most basic summary features. Please implement the more advanced summaries listed here:

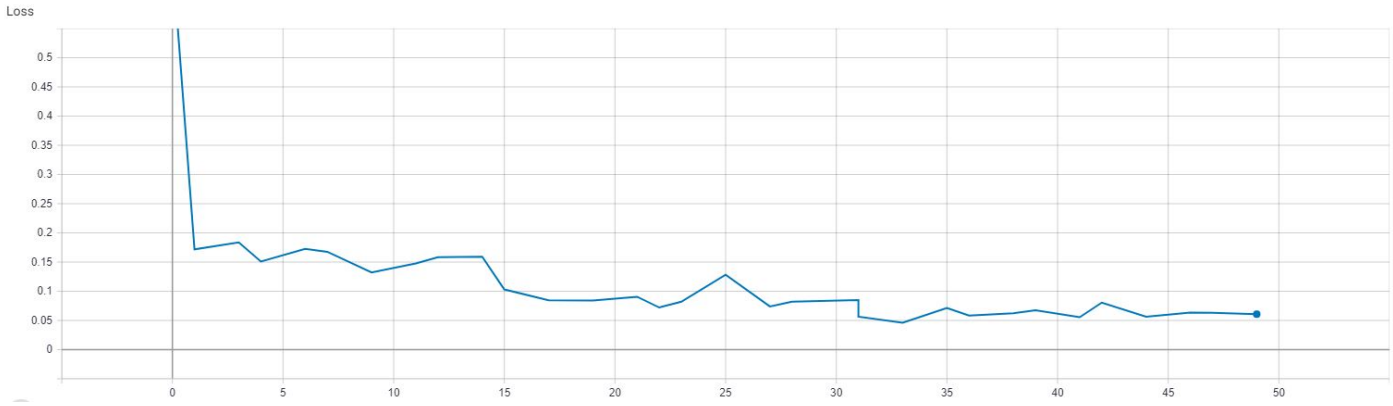
- training loss (should be done)
- testing MAP curves (should be done)
- learning rate
- histogram of gradients

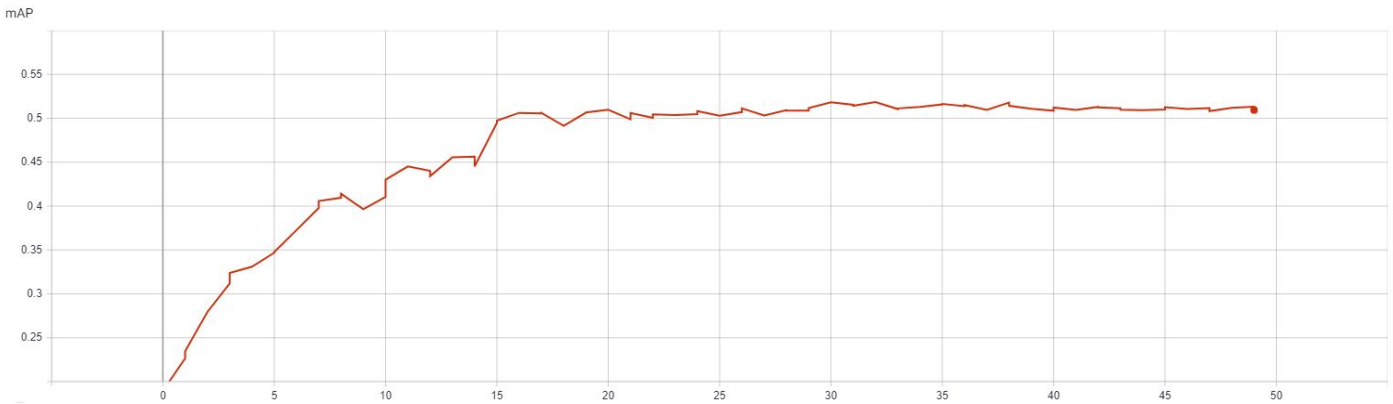
3.3 Train and Test (8 pts)

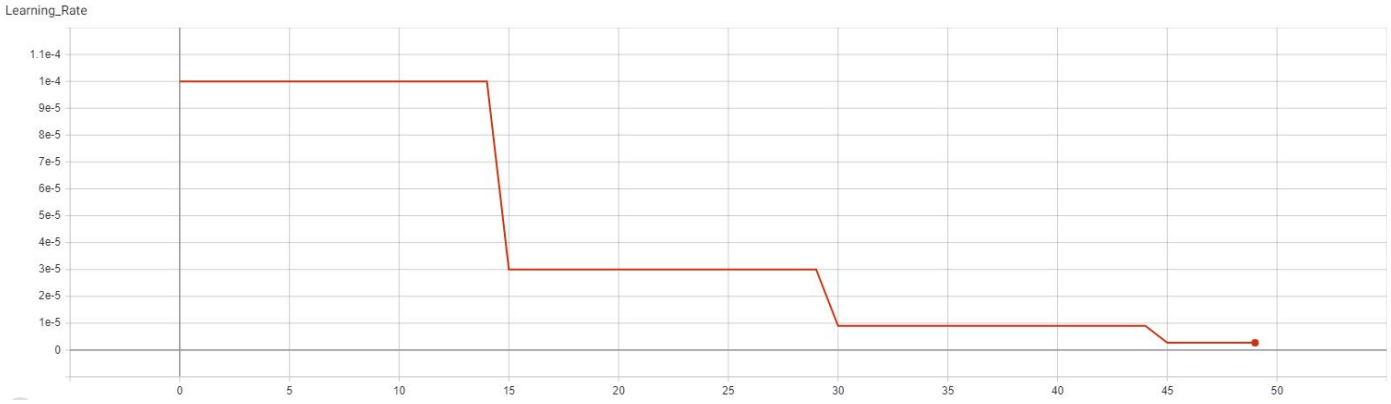
Use the same hyperparameter settings from Task 2, and train the model for 50 epochs. Report tensorboard screenshots for *all* of the summaries listed above (for image summaries show screenshots at $n \geq 3$ iterations)

REMEMBER TO SAVE A MODEL AT THE END OF TRAINING

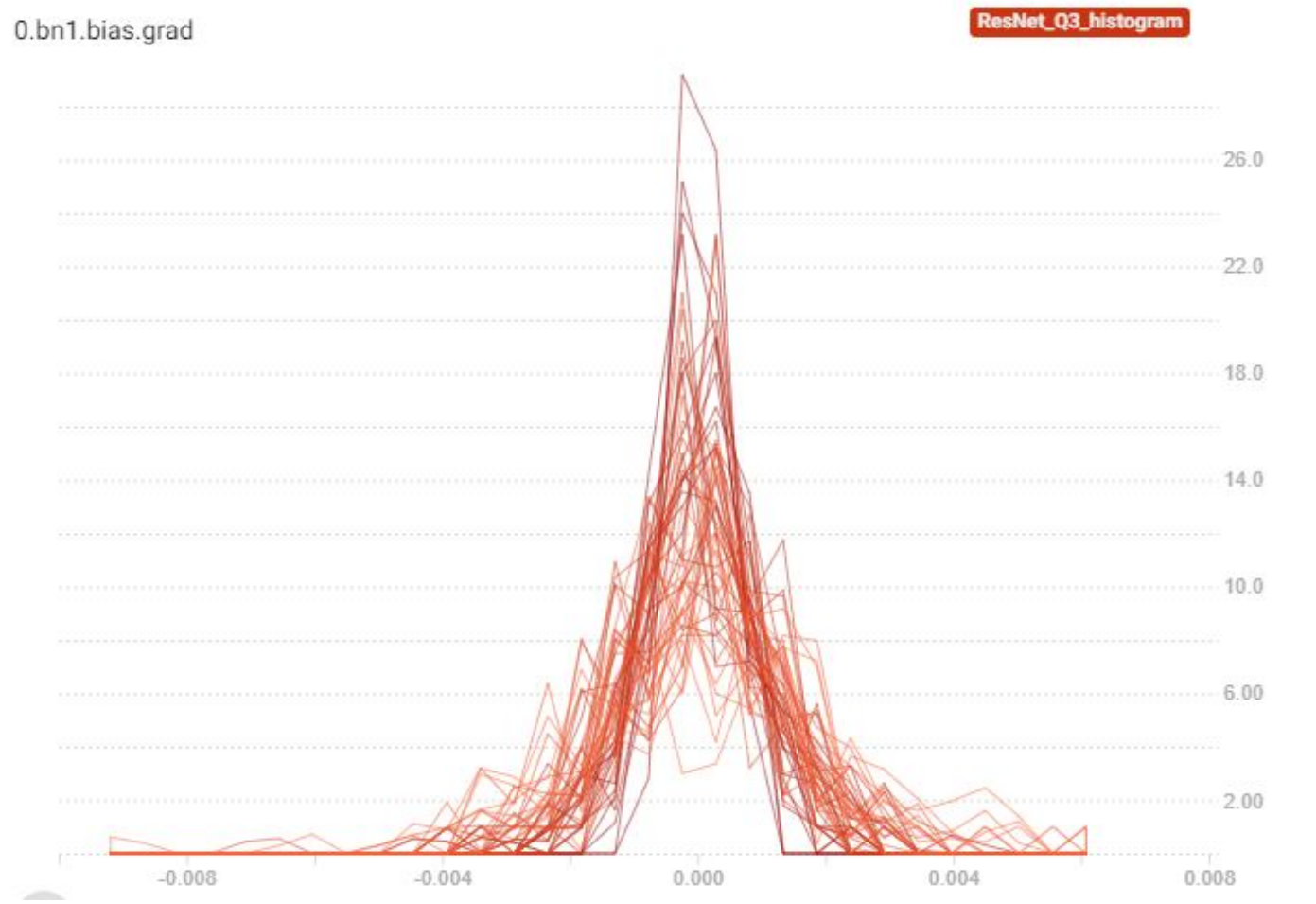
The figure shown below display the training loss, training mAP, learning rate, and histogram of gradients as displayed through TensorBoard.





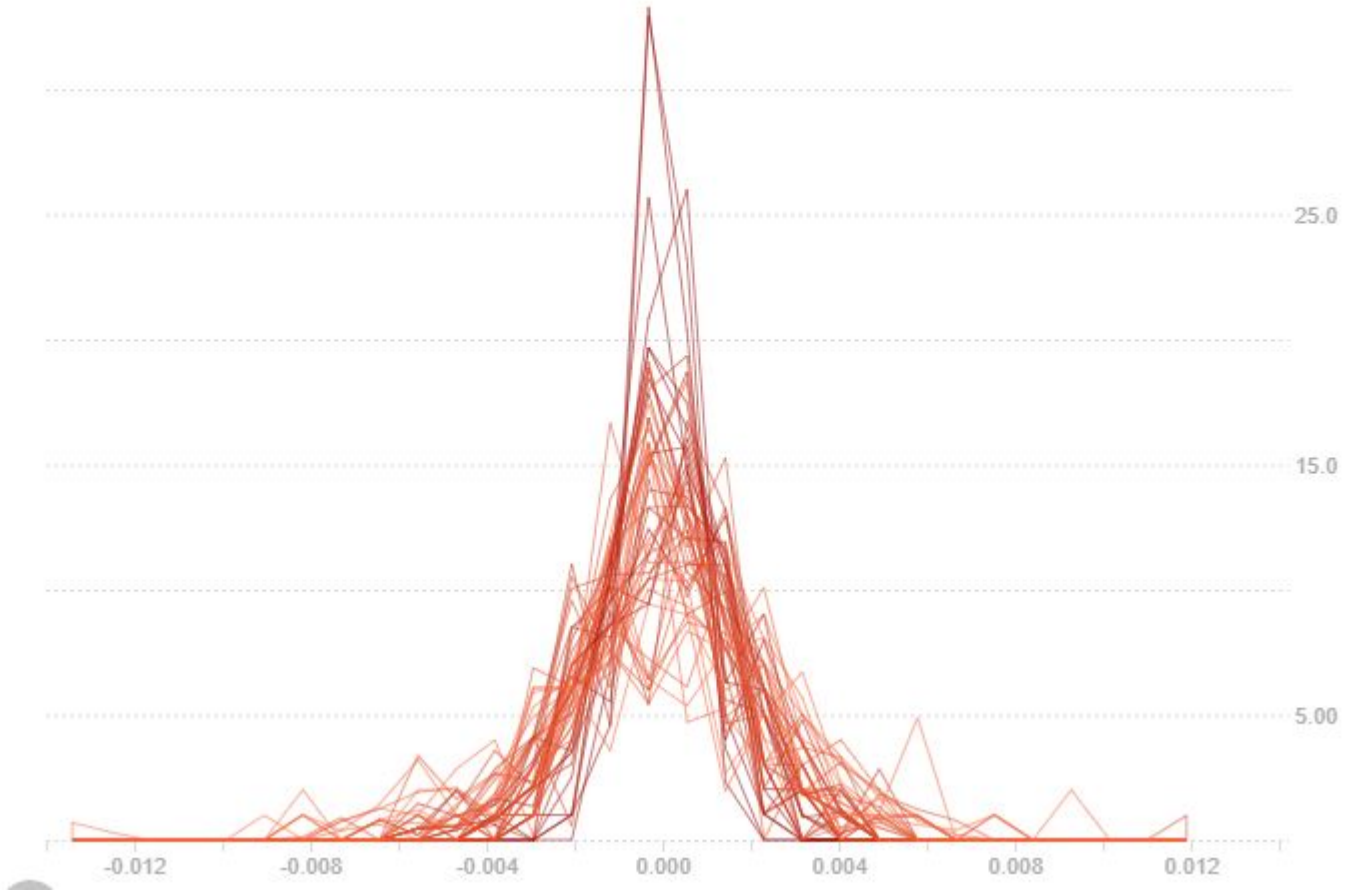


The following 4 images below showcase histogram of gradients for the first two layers:



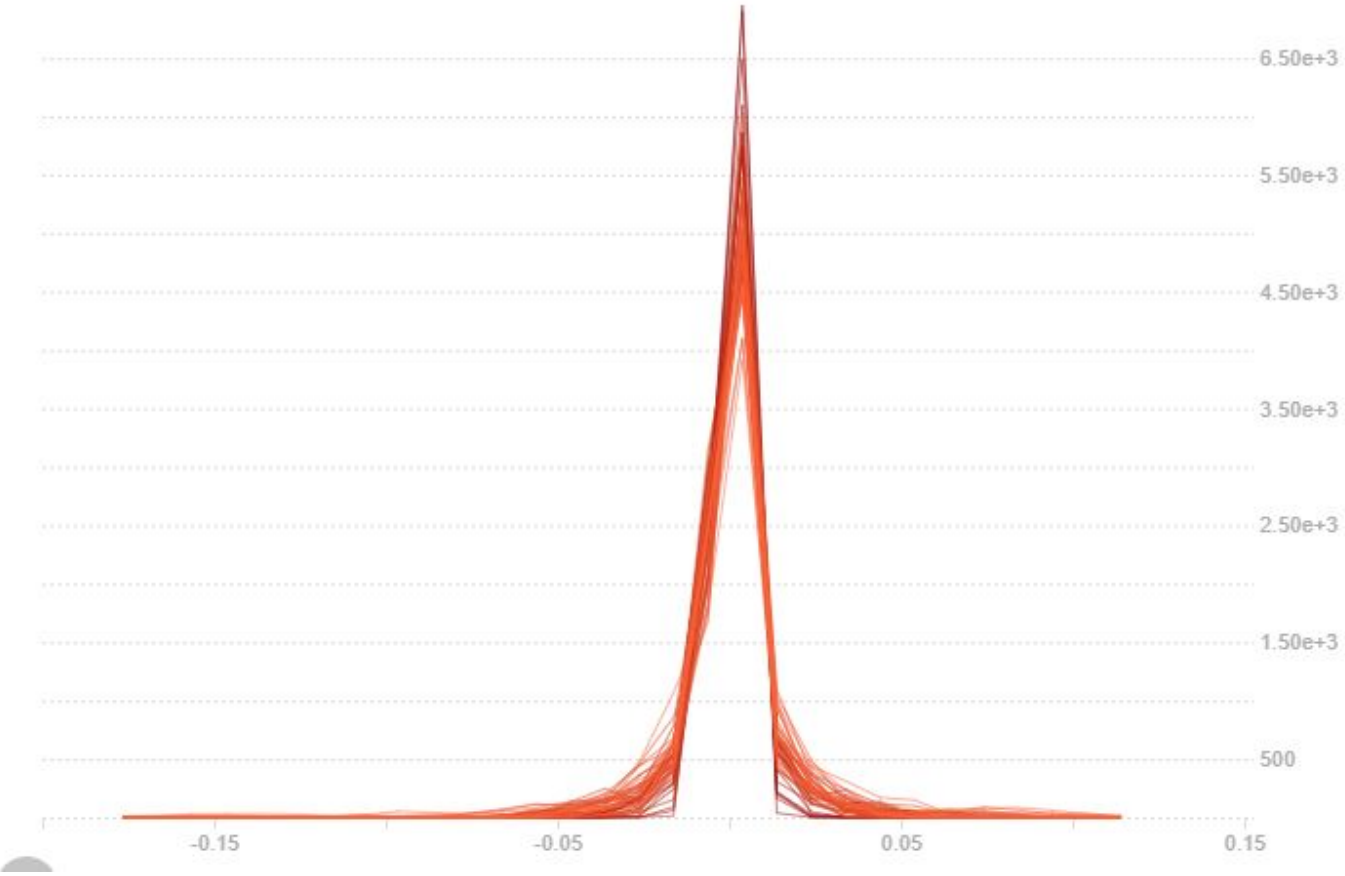
0.bn1.weight.grad

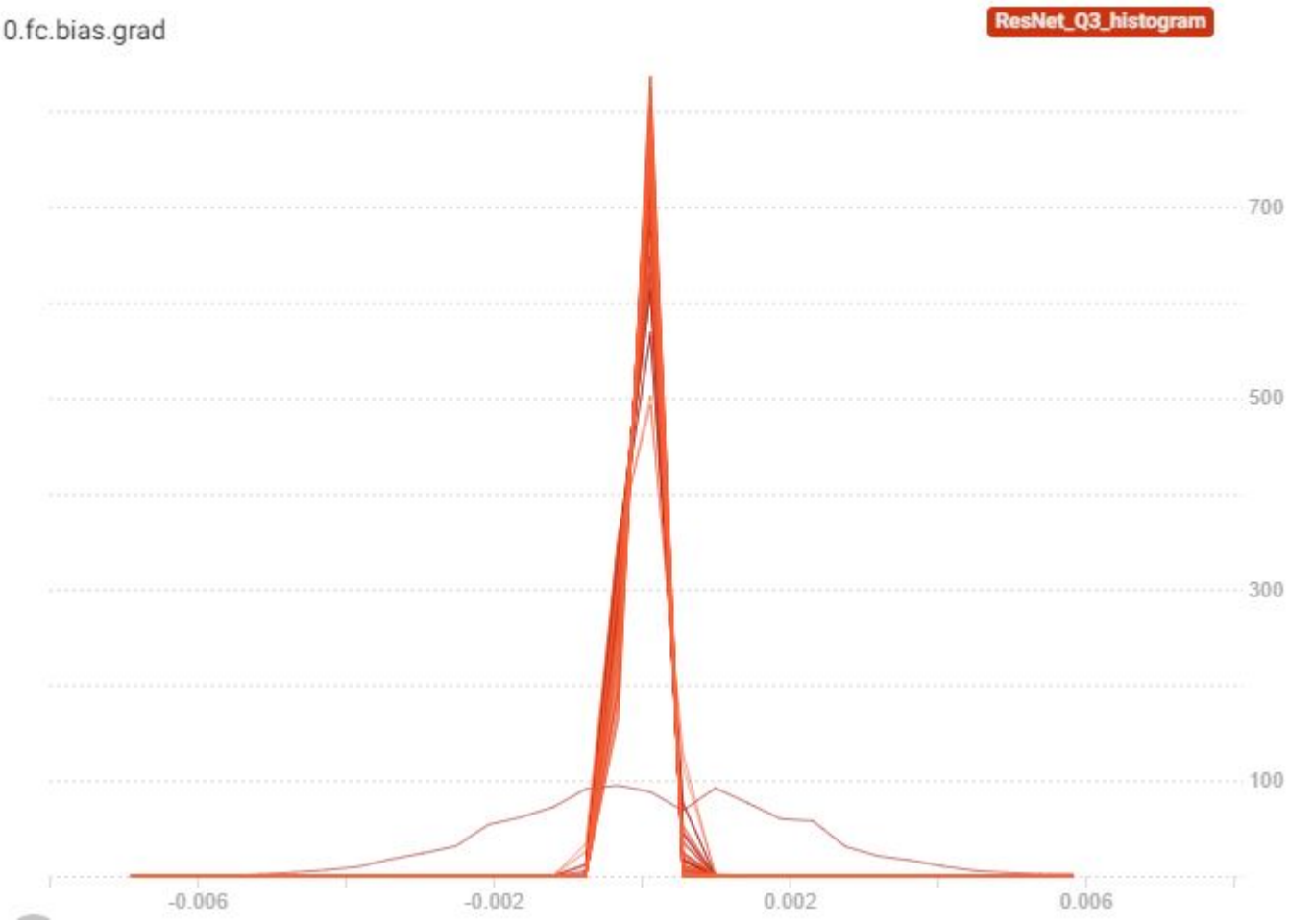
ResNet_Q3_histogram



0.conv1.weight.grad

ResNet_Q3_histogram

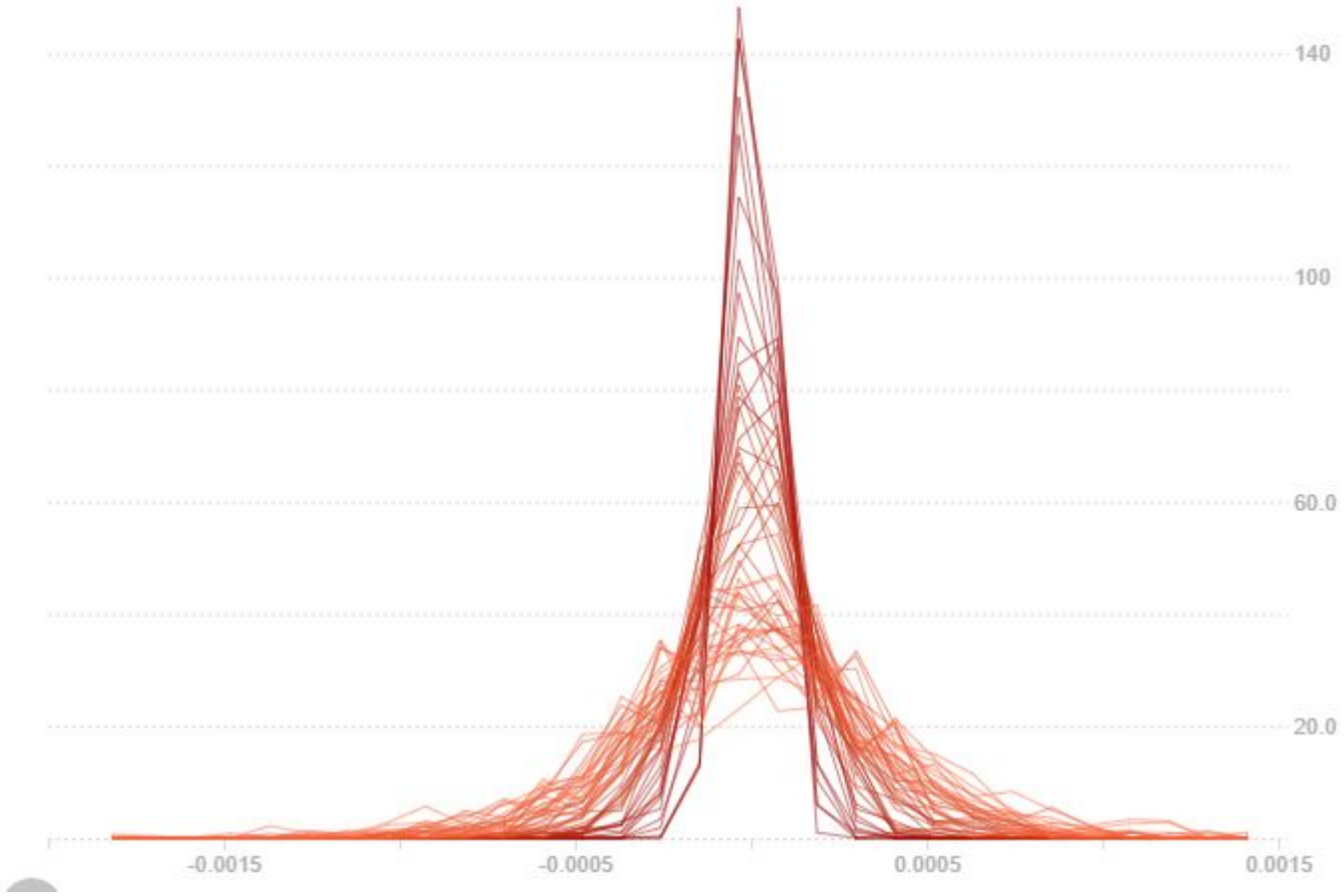




The following 4 images below showcase histogram of gradients for the middle two layers:

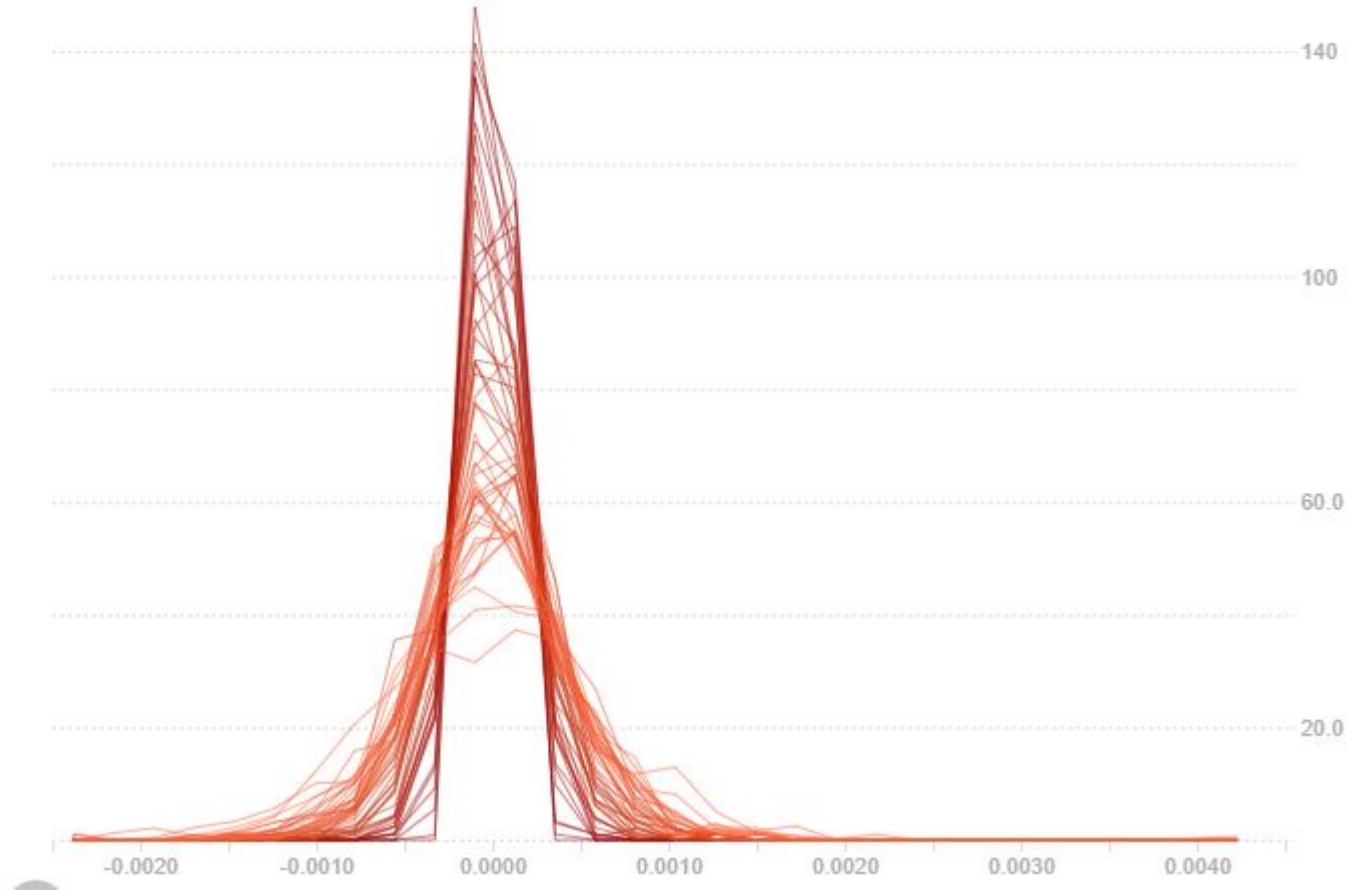
0.layer3.0.bn2.bias.grad

ResNet_Q3_histogram



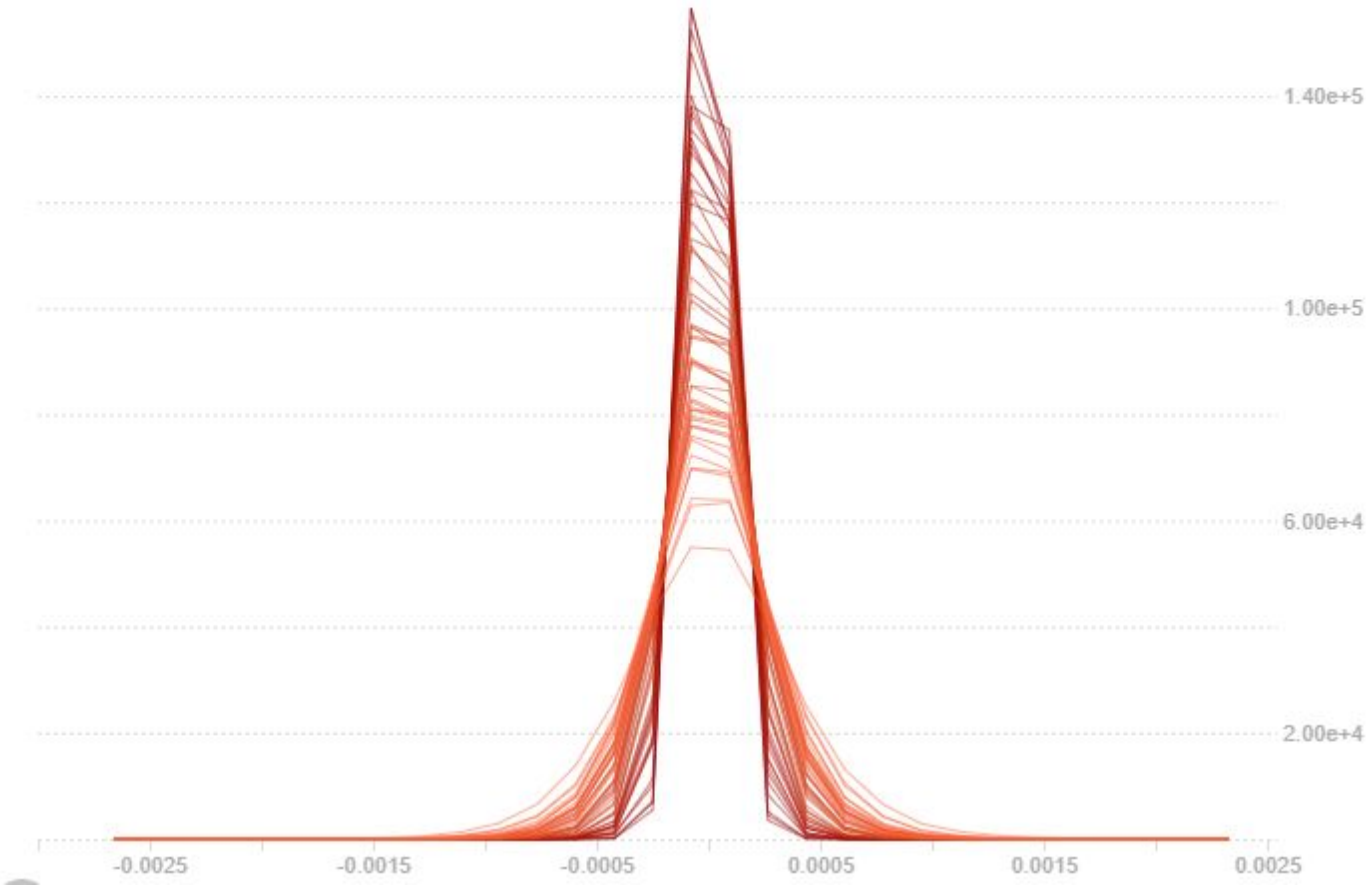
0.layer3.0.bn2.weight.grad

ResNet_Q3_histogram



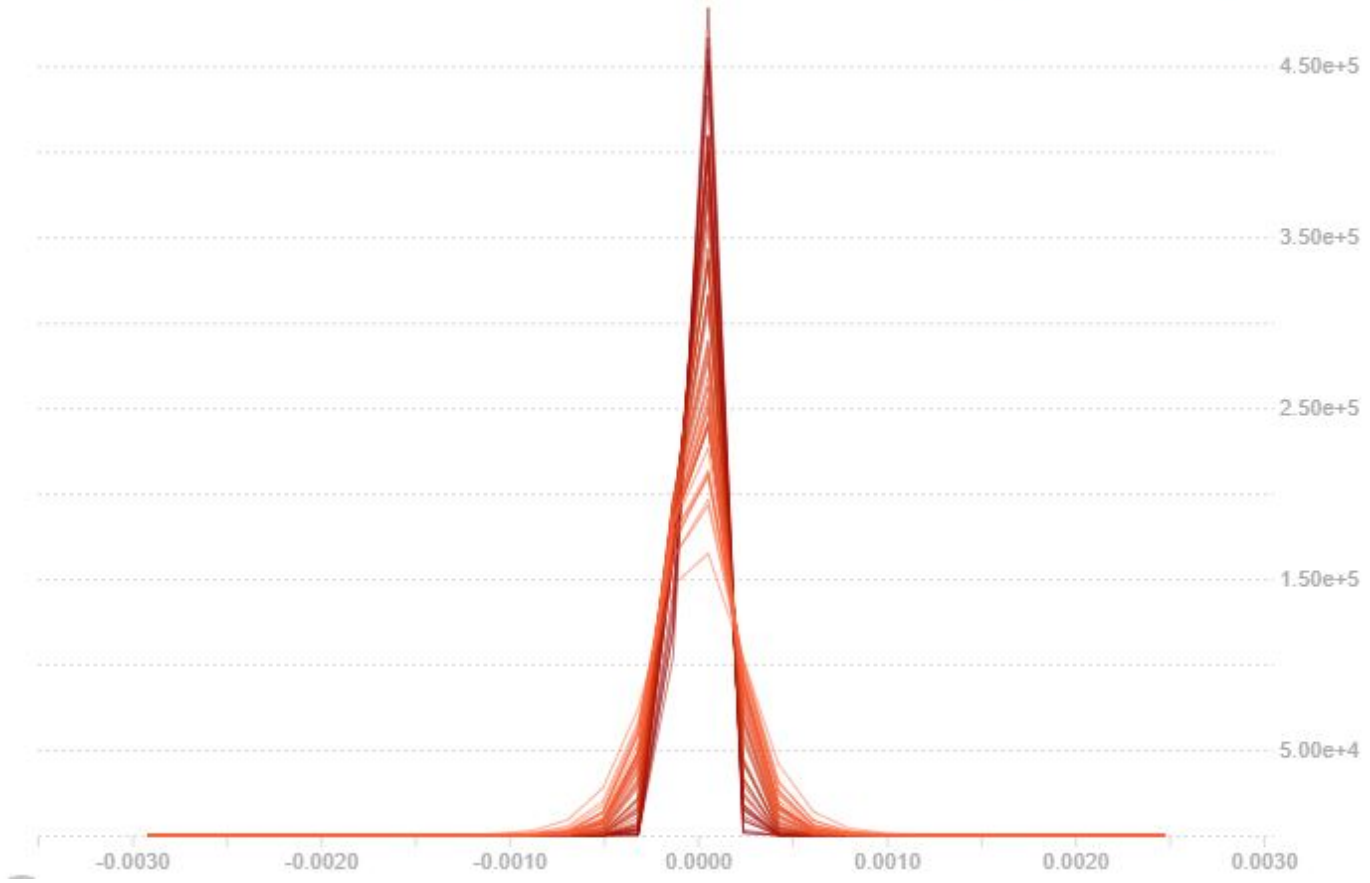
0.layer3.0.conv1.weight.grad

ResNet_Q3_histogram

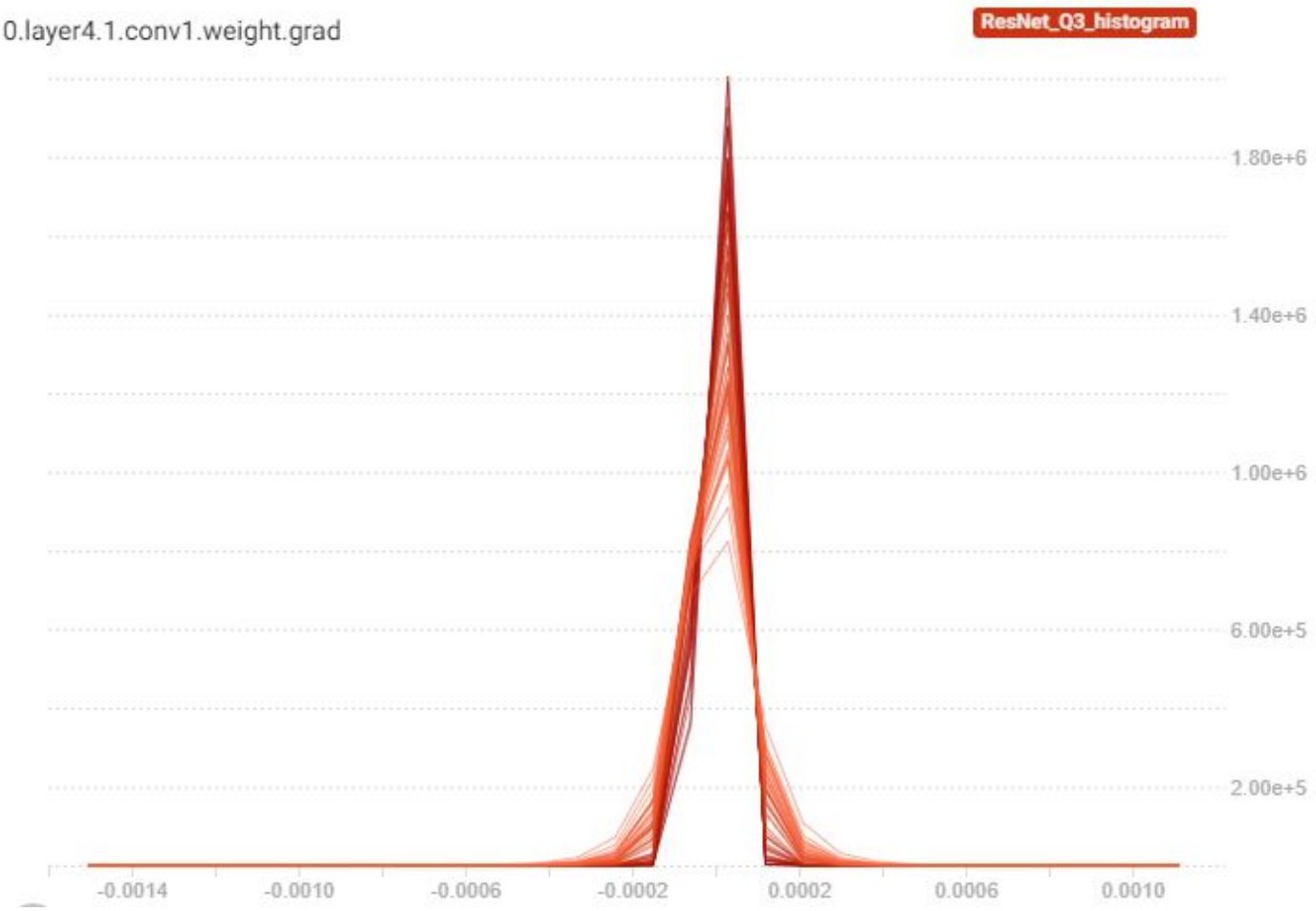


0.layer3.0.conv2.weight.grad

ResNet_Q3_histogram

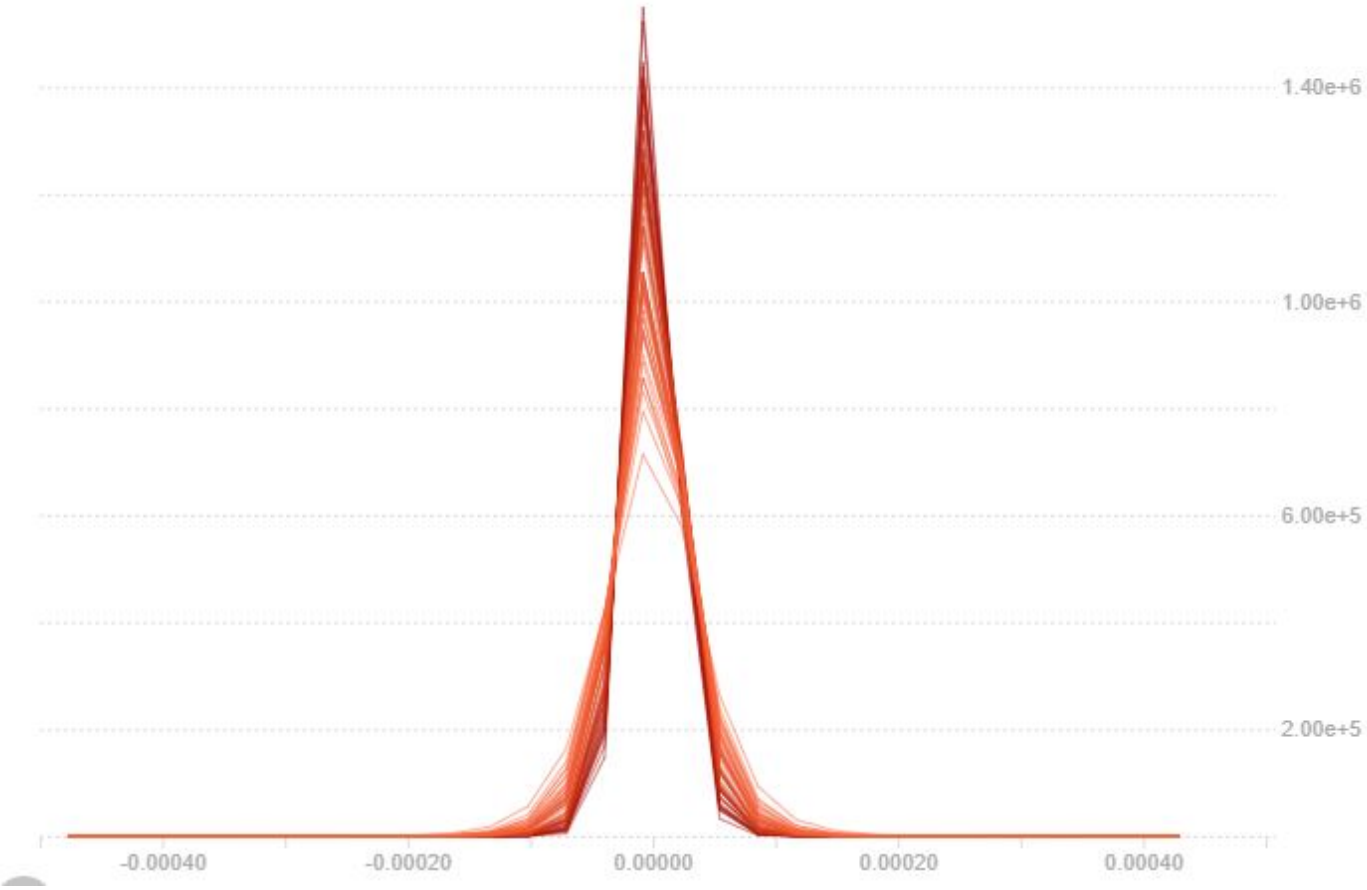


The following 4 images below showcase histogram of gradients for the last two layers:



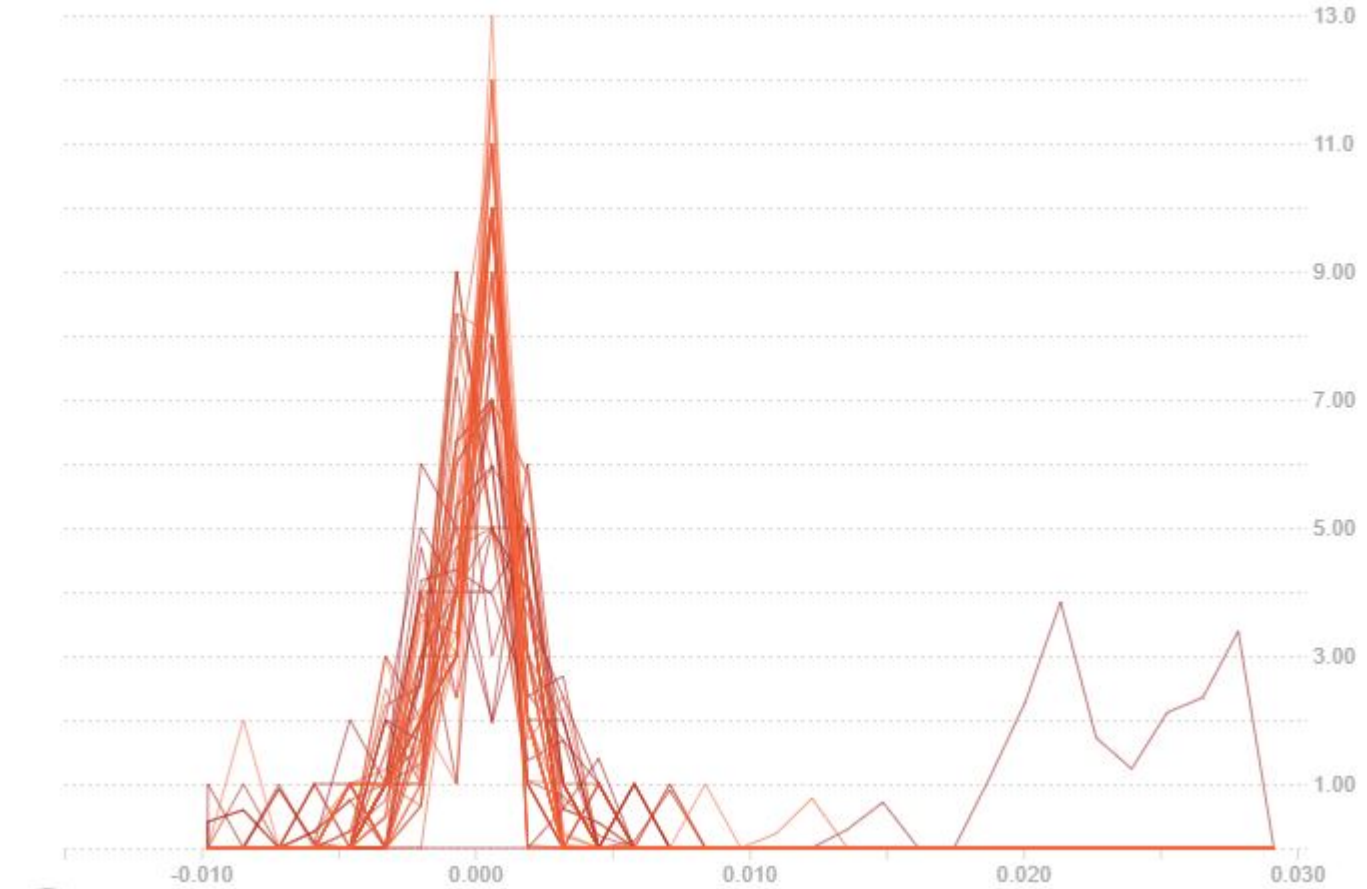
0.layer4.1.conv2.weight.grad

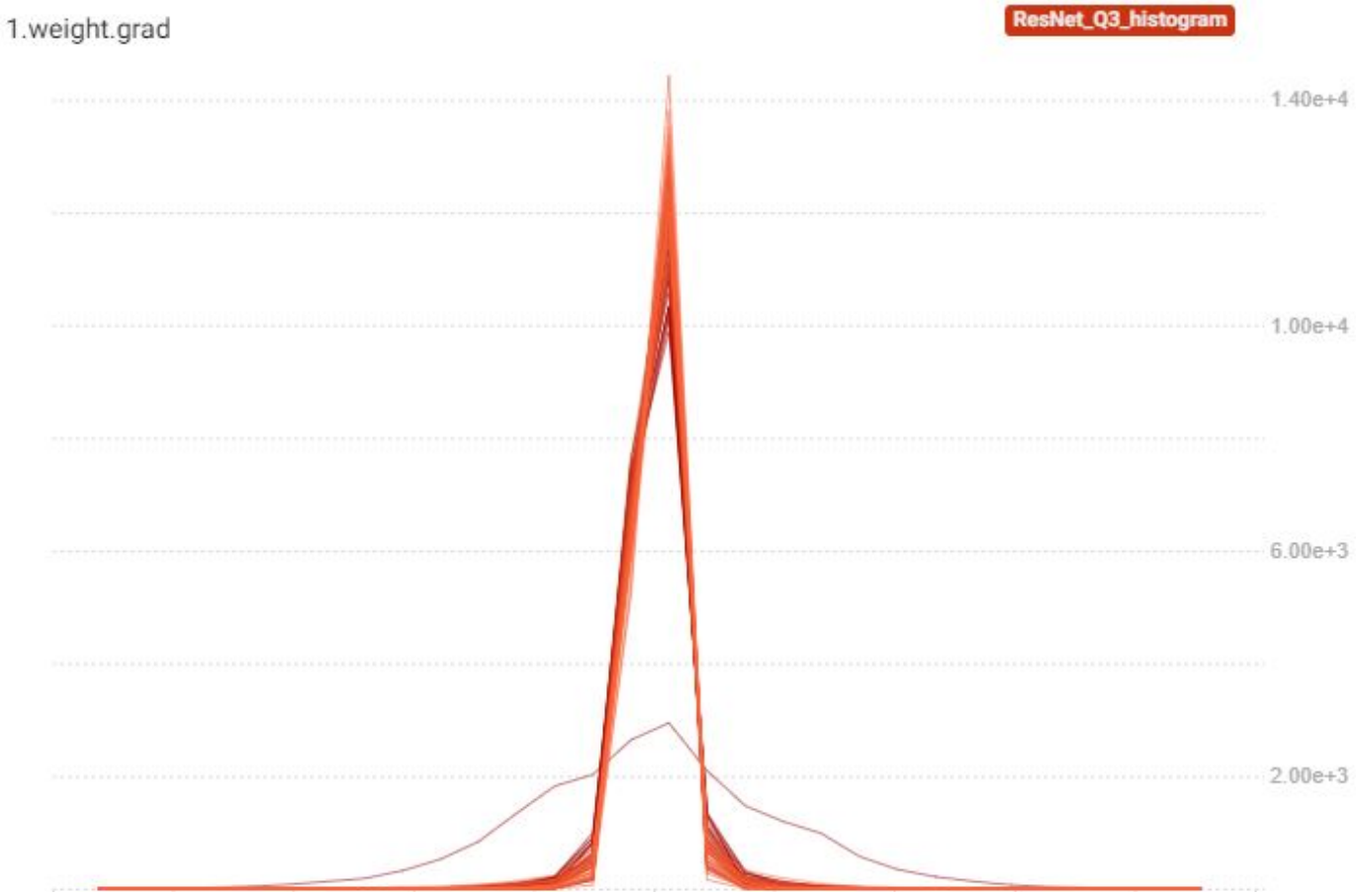
ResNet_Q3_histogram



1.bias.grad

ResNet_Q3_histogram





In [ ]:
```
args = ARGS(batch_size = 32, epochs=50, lr = 0.0001)
args.gamma = 0.3
modelres = models.resnet18(pretrained=False)
model= nn.Sequential(modelres,nn.Linear(1000,20,bias=True))
optimizer = torch.optim.Adam(model.parameters(), lr = args.lr)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=15, gamma=arg
s.gamma)
if __name__ == '__main__':
    test_ap, test_map = trainer.train(args, model, optimizer, scheduler)
    print('test map:', test_map)
```

Appendex A: Training Epoch Information

The following log shown below displays the batch iteration, loss, and mAP calculation at various points during the training process:

Train Epoch: 0 [0 (0%)] Loss: 0.720259 | mAP: 0.073370
Train Epoch: 0 [100 (64%)] Loss: 0.215044 | mAP: 0.189455
Train Epoch: 1 [200 (27%)] Loss: 0.216712 | mAP: 0.226419
Train Epoch: 1 [300 (91%)] Loss: 0.208612 | mAP: 0.234548
Train Epoch: 2 [400 (55%)] Loss: 0.181140 | mAP: 0.279389
Train Epoch: 3 [500 (18%)] Loss: 0.202289 | mAP: 0.311690
Train Epoch: 3 [600 (82%)] Loss: 0.197823 | mAP: 0.323791
Train Epoch: 4 [700 (46%)] Loss: 0.184834 | mAP: 0.330959
Train Epoch: 5 [800 (10%)] Loss: 0.185951 | mAP: 0.346840
Train Epoch: 5 [900 (73%)] Loss: 0.165875 | mAP: 0.347633
Train Epoch: 6 [1000 (37%)] Loss: 0.165044 | mAP: 0.372633
Train Epoch: 7 [1100 (1%)] Loss: 0.156291 | mAP: 0.397830
Train Epoch: 7 [1200 (64%)] Loss: 0.149558 | mAP: 0.405792
Train Epoch: 8 [1300 (28%)] Loss: 0.155647 | mAP: 0.409463
Train Epoch: 8 [1400 (92%)] Loss: 0.146398 | mAP: 0.414038
Train Epoch: 9 [1500 (55%)] Loss: 0.153696 | mAP: 0.396369
Train Epoch: 10 [1600 (19%)] Loss: 0.137831 | mAP: 0.410398
Train Epoch: 10 [1700 (83%)] Loss: 0.167851 | mAP: 0.430070
Train Epoch: 11 [1800 (46%)] Loss: 0.130406 | mAP: 0.445091
Train Epoch: 12 [1900 (10%)] Loss: 0.152793 | mAP: 0.440219
Train Epoch: 12 [2000 (74%)] Loss: 0.140506 | mAP: 0.434110
Train Epoch: 13 [2100 (38%)] Loss: 0.140575 | mAP: 0.455477
Train Epoch: 14 [2200 (1%)] Loss: 0.118644 | mAP: 0.456327
Train Epoch: 14 [2300 (65%)] Loss: 0.138294 | mAP: 0.445303
Train Epoch: 15 [2400 (29%)] Loss: 0.103462 | mAP: 0.494774
Train Epoch: 15 [2500 (92%)] Loss: 0.114315 | mAP: 0.497309
Train Epoch: 16 [2600 (56%)] Loss: 0.105307 | mAP: 0.506209
Train Epoch: 17 [2700 (20%)] Loss: 0.126236 | mAP: 0.505680
Train Epoch: 17 [2800 (83%)] Loss: 0.111384 | mAP: 0.506179
Train Epoch: 18 [2900 (47%)] Loss: 0.089492 | mAP: 0.491313
Train Epoch: 19 [3000 (11%)] Loss: 0.129798 | mAP: 0.506378
Train Epoch: 19 [3100 (75%)] Loss: 0.124896 | mAP: 0.506707
Train Epoch: 20 [3200 (38%)] Loss: 0.097039 | mAP: 0.509701
Train Epoch: 21 [3300 (2%)] Loss: 0.069134 | mAP: 0.498853
Train Epoch: 21 [3400 (66%)] Loss: 0.109458 | mAP: 0.506017
Train Epoch: 22 [3500 (29%)] Loss: 0.070703 | mAP: 0.500423
Train Epoch: 22 [3600 (93%)] Loss: 0.074366 | mAP: 0.504471
Train Epoch: 23 [3700 (57%)] Loss: 0.092572 | mAP: 0.503754
Train Epoch: 24 [3800 (20%)] Loss: 0.100229 | mAP: 0.504639
Train Epoch: 24 [3900 (84%)] Loss: 0.089912 | mAP: 0.508232
Train Epoch: 25 [4000 (48%)] Loss: 0.083635 | mAP: 0.502918
Train Epoch: 26 [4100 (11%)] Loss: 0.083434 | mAP: 0.507066
Train Epoch: 26 [4200 (75%)] Loss: 0.070950 | mAP: 0.511265

Train Epoch: 27 [4300 (39%)] Loss: 0.073017 | mAP: 0.503205
Train Epoch: 28 [4400 (3%)] Loss: 0.074057 | mAP: 0.509178
Train Epoch: 28 [4500 (66%)] Loss: 0.089060 | mAP: 0.508615
Train Epoch: 29 [4600 (30%)] Loss: 0.081091 | mAP: 0.508820
Train Epoch: 29 [4700 (94%)] Loss: 0.082942 | mAP: 0.511548
Train Epoch: 30 [4800 (57%)] Loss: 0.063460 | mAP: 0.518176
Train Epoch: 31 [4900 (21%)] Loss: 0.073838 | mAP: 0.515324
Train Epoch: 31 [5000 (85%)] Loss: 0.062788 | mAP: 0.514318
Train Epoch: 32 [5100 (48%)] Loss: 0.067382 | mAP: 0.518381
Train Epoch: 33 [5200 (12%)] Loss: 0.063792 | mAP: 0.510551
Train Epoch: 33 [5300 (76%)] Loss: 0.058528 | mAP: 0.511280
Train Epoch: 34 [5400 (39%)] Loss: 0.080845 | mAP: 0.512993
Train Epoch: 35 [5500 (3%)] Loss: 0.071587 | mAP: 0.515748
Train Epoch: 35 [5600 (67%)] Loss: 0.066521 | mAP: 0.516310
Train Epoch: 36 [5700 (31%)] Loss: 0.038676 | mAP: 0.513884
Train Epoch: 36 [5800 (94%)] Loss: 0.109776 | mAP: 0.514998
Train Epoch: 37 [5900 (58%)] Loss: 0.056390 | mAP: 0.509673
Train Epoch: 38 [6000 (22%)] Loss: 0.057494 | mAP: 0.517916
Train Epoch: 38 [6100 (85%)] Loss: 0.048027 | mAP: 0.514223
Train Epoch: 39 [6200 (49%)] Loss: 0.062421 | mAP: 0.510778
Train Epoch: 40 [6300 (13%)] Loss: 0.064487 | mAP: 0.508616
Train Epoch: 40 [6400 (76%)] Loss: 0.060632 | mAP: 0.512278
Train Epoch: 41 [6500 (40%)] Loss: 0.076664 | mAP: 0.509637
Train Epoch: 42 [6600 (4%)] Loss: 0.053598 | mAP: 0.513044
Train Epoch: 42 [6700 (68%)] Loss: 0.054771 | mAP: 0.512391
Train Epoch: 43 [6800 (31%)] Loss: 0.044373 | mAP: 0.511334
Train Epoch: 43 [6900 (95%)] Loss: 0.071721 | mAP: 0.509659
Train Epoch: 44 [7000 (59%)] Loss: 0.068732 | mAP: 0.509379
Train Epoch: 45 [7100 (22%)] Loss: 0.108448 | mAP: 0.510053
Train Epoch: 45 [7200 (86%)] Loss: 0.069663 | mAP: 0.512628
Train Epoch: 46 [7300 (50%)] Loss: 0.054835 | mAP: 0.510637
Train Epoch: 47 [7400 (13%)] Loss: 0.069490 | mAP: 0.511583
Train Epoch: 47 [7500 (77%)] Loss: 0.059993 | mAP: 0.508102
Train Epoch: 48 [7600 (41%)] Loss: 0.058032 | mAP: 0.512003
Train Epoch: 49 [7700 (4%)] Loss: 0.056961 | mAP: 0.513160
Train Epoch: 49 [7800 (68%)] Loss: 0.047862 | mAP: 0.509374
test map: 0.5124332392148173