Content

1 Introduction: FPGA+ALGORITHM

1.1 Background

1.1.1 Aim: ① Compactness ② Speed ③ Accuracy

Explanation: I. Compression (reduce MODEL SIZE or RUNTIME—low bit numbers from 8 bits to 1bit). II. Acceleration (tricks on software and hardware)

1.1.2 Requirement: Python + Pytorch or Tensorflow + Cuda + VHDL or Verilog (FPGA Simulator)

1.2 Experiment

1.2.1 Classic low bit model: Binary-Net, Tenary-Net

Link: https://pan.baidu.com/s/14DJFtvLTwmRS3PTHsU3xEw Verification Code: 9so4

Reference Link: https:/github.com/BertMoons/QuantizedNeuralNetworks-Keras-Tensorflow
https://github.com/NervanaSystems/distiller
https://github.com/dongyp13/Stochastic-Quantization

1.2.2 Hardware realization: https://arxiv.org/abs/1702.03044

1.2.3 Result: ① Accuracy Performance-compared with CPU

② Power Waster-compared with GPU

Explanation: The main performance bottleneck for CPU is that convolution requires many multiplications and addition operations. The large number of weight parameters involved in the calculation will bring lots of requests of memory access. However, bit operation in FPGA will accelerate those complex multiplications.

1.3 Other points

1.3.1 Convolution optimization

Memory exchanges time
Multiplication optimization
GPU optimization

1.3.2 Structural pruning

Sparse connection（×reduce network runtime）
Tensor decomposition (SVD、tucker、block)
channel pruning
low bit network

1.4 Reference:

用 TensorFlow 压缩神经网络
tensorflow 模型量化
https://openreview.net/forum?id=By5ugjyCb
https://openreview.net/forum?id=Skh4jRcKQ
https://arxiv.org/abs/1702.03044

2 Report 1: Binary network

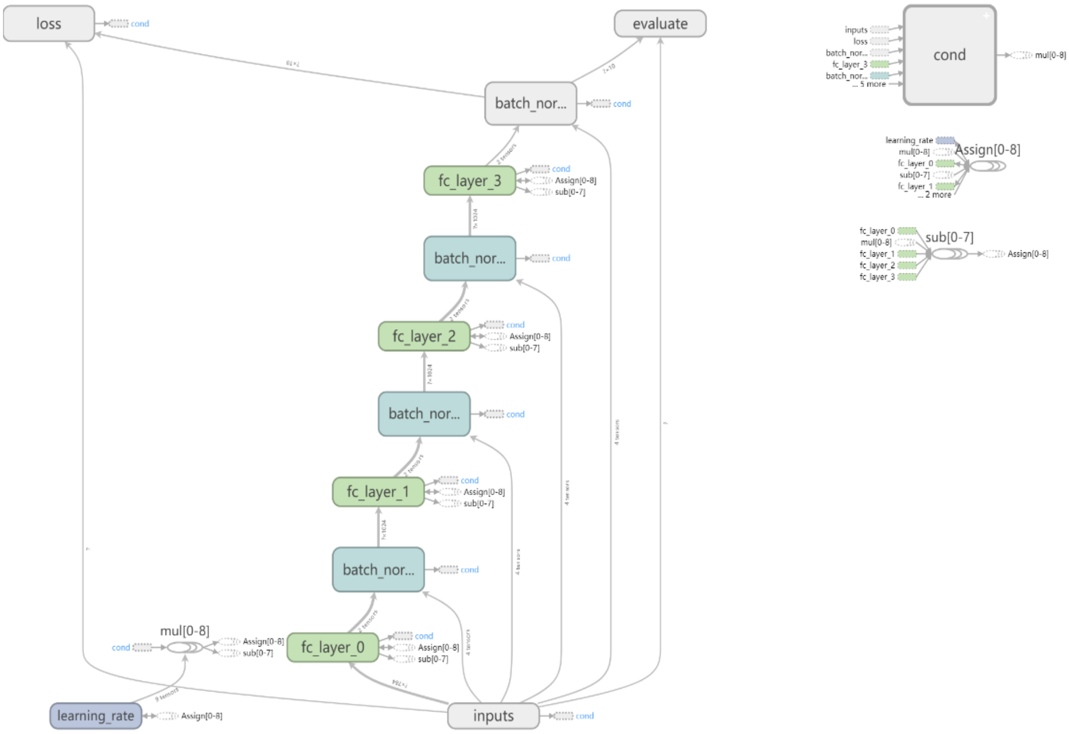


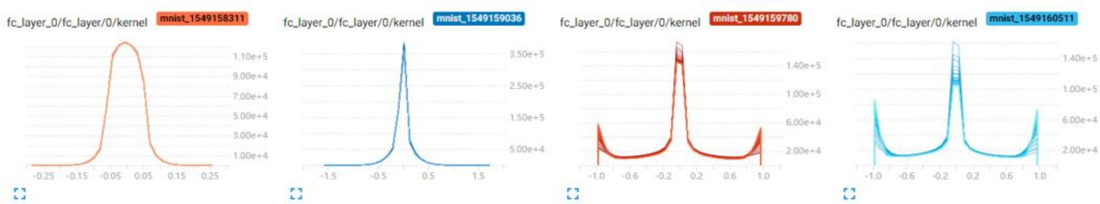
Figure.1. Full connected network for the Mnist classifications

2.1 Experiment 1: classification of the Mnist dataset

Dataset: Mnist   batch size: 200   resolution: 28×28   total number of weights: 2,910,208

Dropout	Binary	Stochastic	Learning rate	Layer_0	Layer_1	Layer_2	Layer_4	Accuracy
×	×	×	10	0%	0.1%	13.7%	19.9%	98.6%
√	×	×	100	0%	0%	16.2%	24.8%	98.8%
×	√	×	10000	0%	0%	14.3%	19.1%	98.6%
×	√	√	10000	0%	5.9%	53.7%	53.7%	98.4%

Table.1. the percentages of weights(kernels) equal to 1 and -1 in each layer



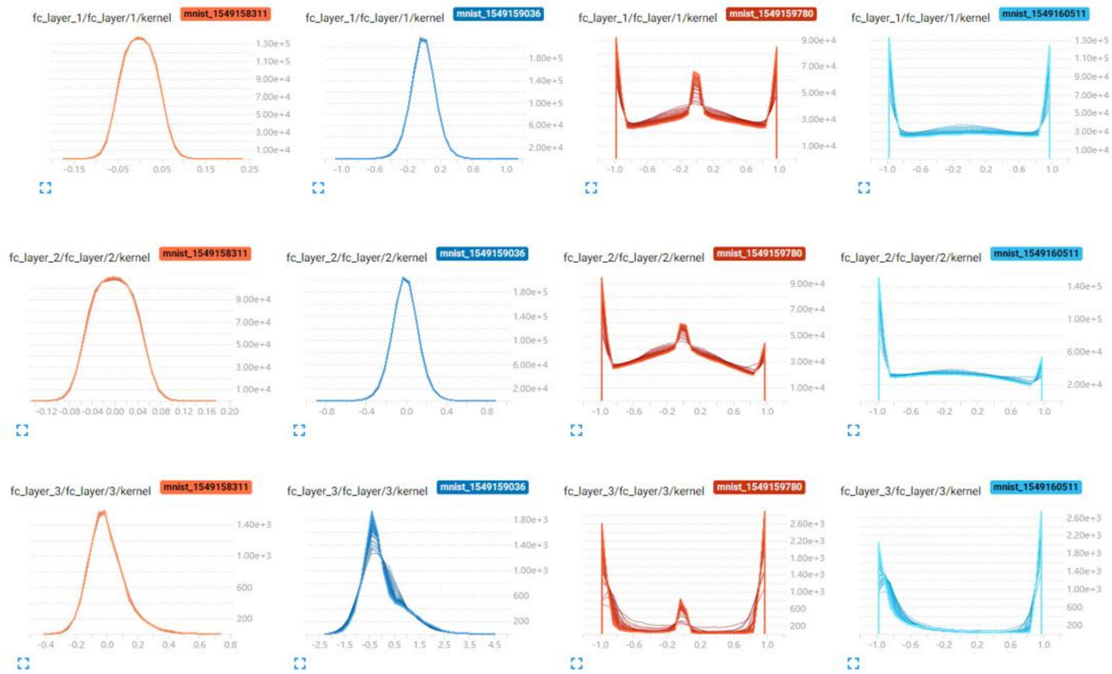


Figure.2. Weight distributions in four network structures: baseline, dropout, binary and binary+stochastic

Figure.3. Bias: all are near zero.

2.2 Experiment 2: classification of the Cifar-10 dataset

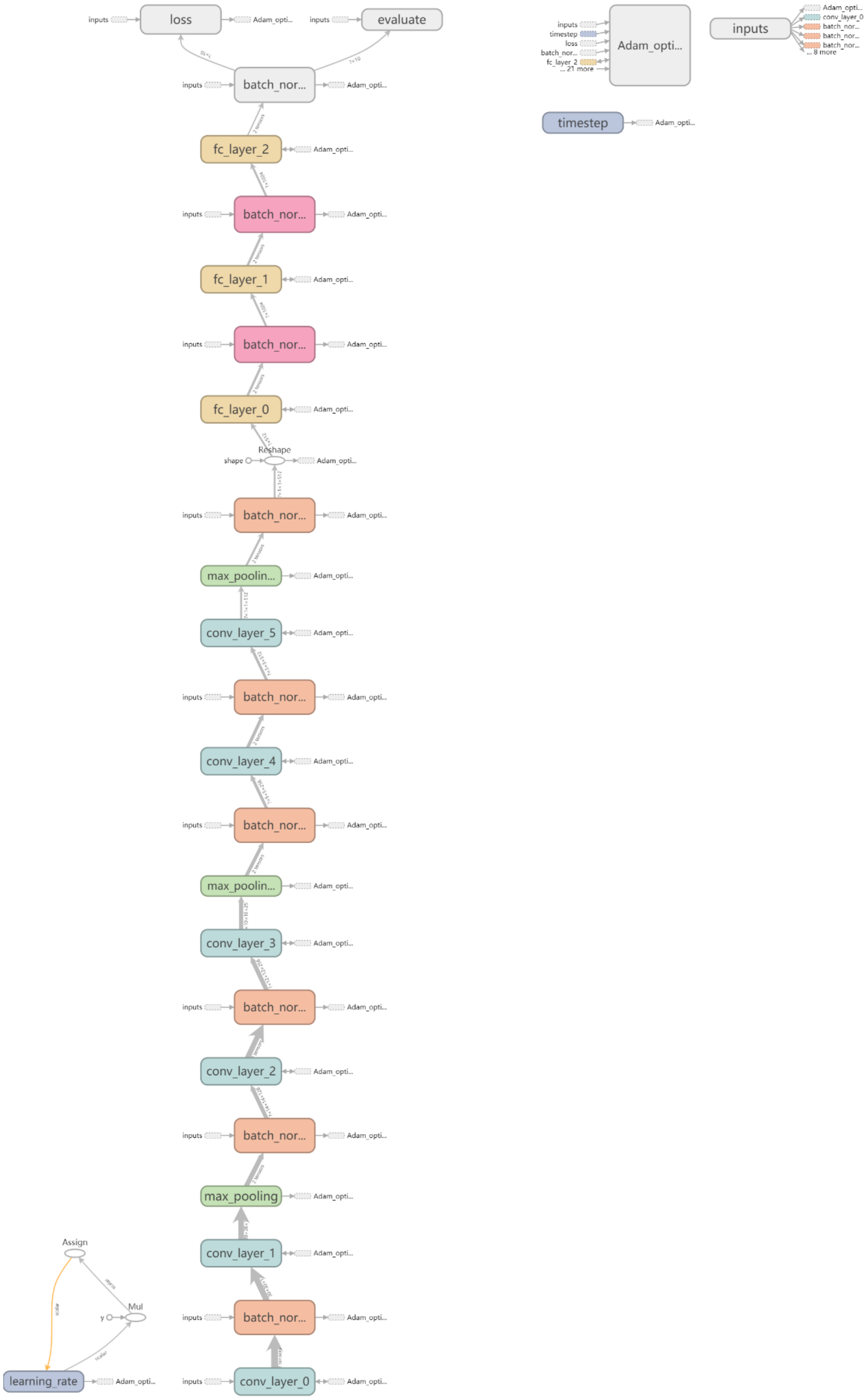
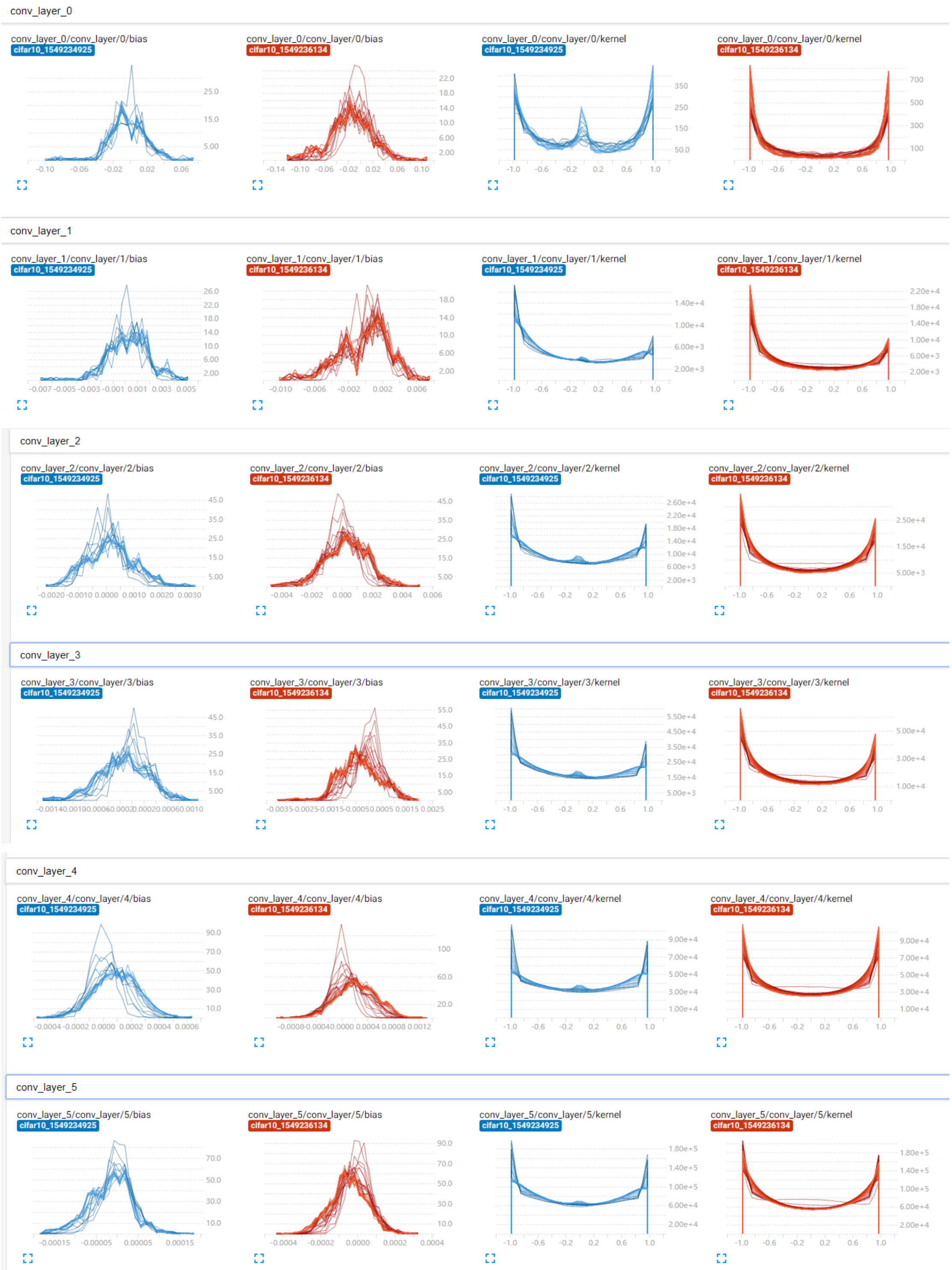


Figure.4. Convolutional neural network for the Cifar10 classifications

Dropout	Binary	Stochastic	Learning rate	Layer_0	Layer_1	Layer_2	Layer_4	Accuracy
√	×	×	0.001					83.28%
×	√	×	0.1					83.70%
×	√	√	0.1					84.83%

Table.2. the percentages of weights(kernels) equal to 1 and -1 in each layer

In fact, the binary operation is the changes of Adam optimizer



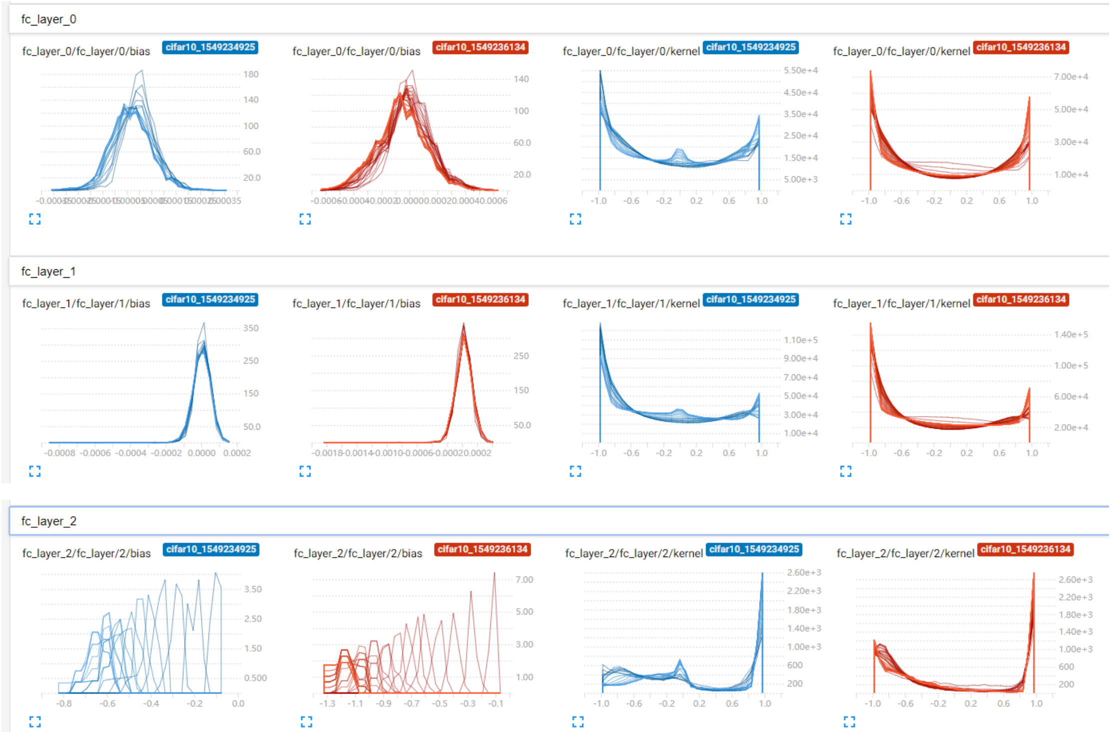


Figure.5. Weight and bias distributions in three network structures: baseline, binary and binary+stochastic

3 Report 2: Tenary Weight Network

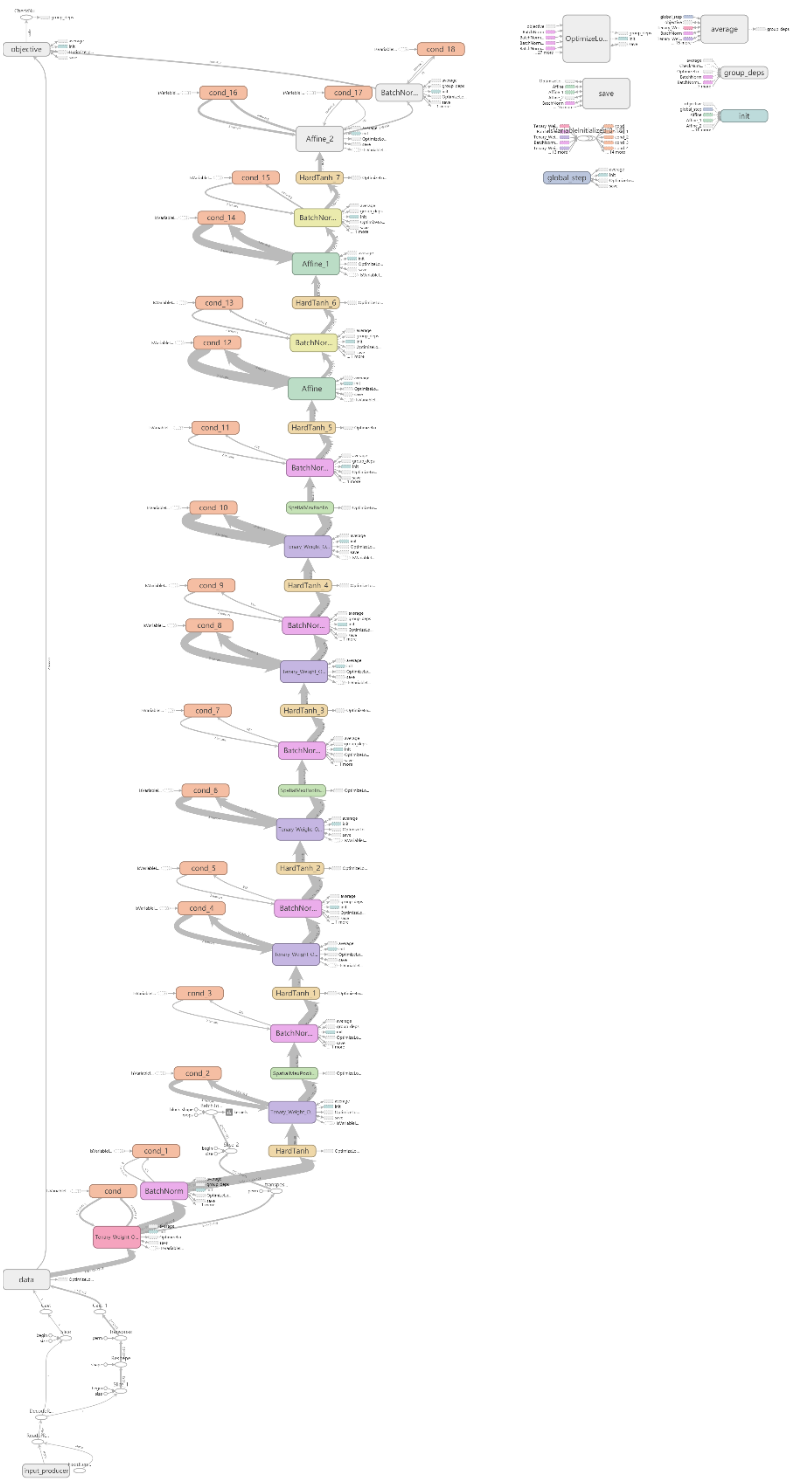


Figure.6. Tenary Weight Network for the Cifar10 classifications

Explination: All the weights of the network are limited to 1, 0, -1, and only 2-bit is needed to store the weights information. The Euclidean distance between TWNs and full-precision networks is guaranteed to be the smallest. In order to achieve this efficiently, a threshold-based function is used to approximate it.

In terms of performance, TWNs are more descriptive than binary precision, and can compress 16-32 times compared with full-precision networks, and the cost of multiplication will be reduced.

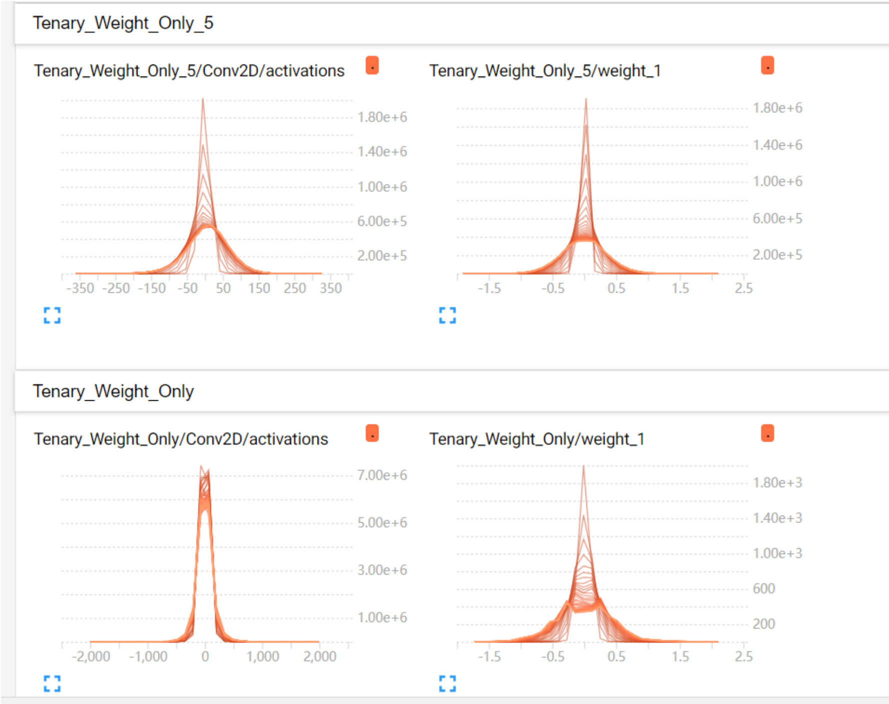
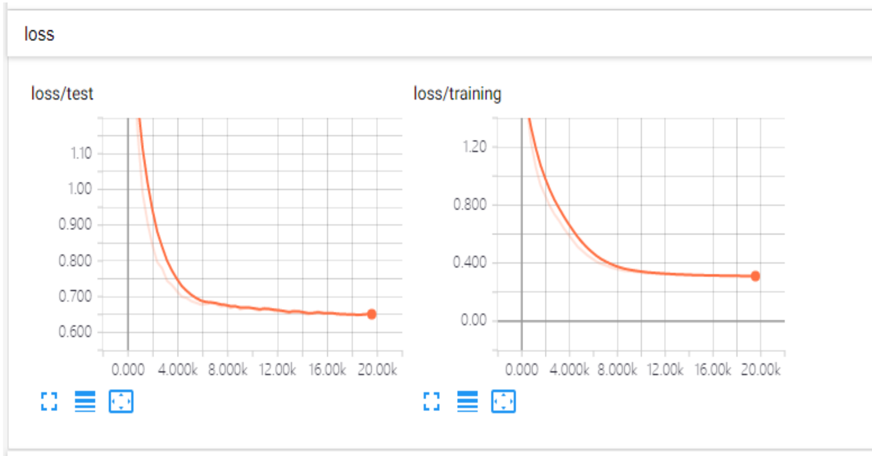


Figure.7. the boundary constraint of the parameters: upper bound 0.2, lower bound -0.2



Finished epoch 50

Test Accuracy: 0.855

Test Loss: 0.653

Training Accuracy: 1.000

Training Loss: 0.310