Deep Compression

November 24, 2017

Procedure

- VGG19 was used in CIFAR10.
- Regularization techniques employed during training were L2, Batch normalization, Dropout, Data Augmentation.
- Bias terms (w_0) were employed in all layers, including the convolutional layers.
- Batch normalization was used after each convolutional layer.
- Model Description: VGG19_BN_drop_10
  Image size: 32x32x3

[64, 64, 'M',	16x16x64
128, 128, 'M',	8x8x128
256, 256, 256, 256, 'M',	4x4x256
512, 512, 512, 512, 'M',	2x2x512
512, 512, 512, 512, 'M',	1x1x512
4906, 4906, 10]	

Pruning

- VGG19 using CIFAR10 generates an overall of 38M (38,969,930) parameters to be trained.
- Pruning employs the standard deviation as a quality parameter.
- 1 iteration is composed by a "pruning" and "retraining" stage.
- The retraining has the following configuration:
  iterations: 25
  number of epochs: 25
  initial learning rate: 0.05,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,
  0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.001,0.001
  learning rate schedule: 3,12,18

Weights Distribution

- go to Distribution Path
- go to Distribution Path for weight clustering
/home/medina/DeepLearning/notebooks/DeepCompression/Distributions

Experiments

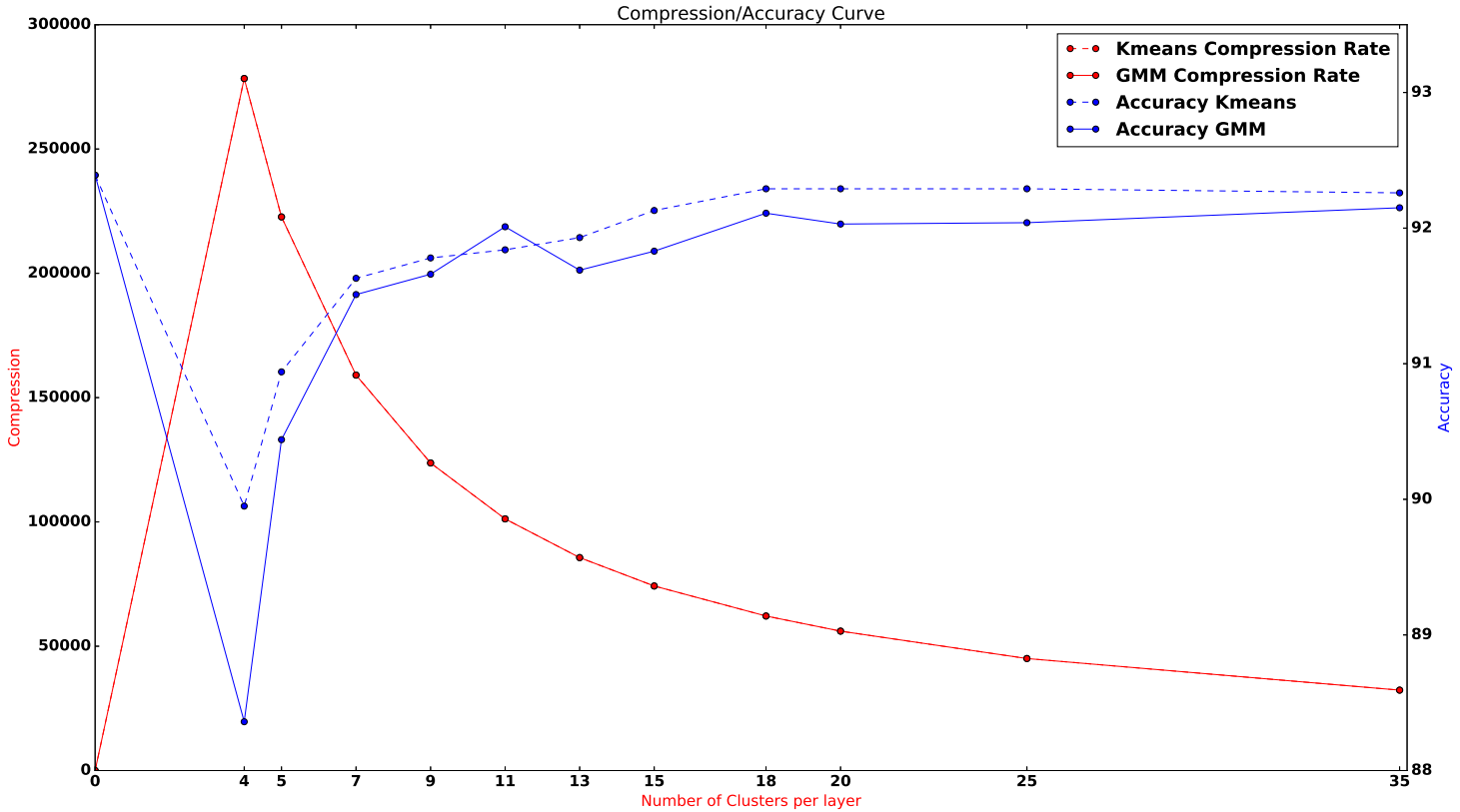


Figure 1: Compression vs Accuracy for a VGG19. unique_non-zero_weights / all_parameters

Experiments

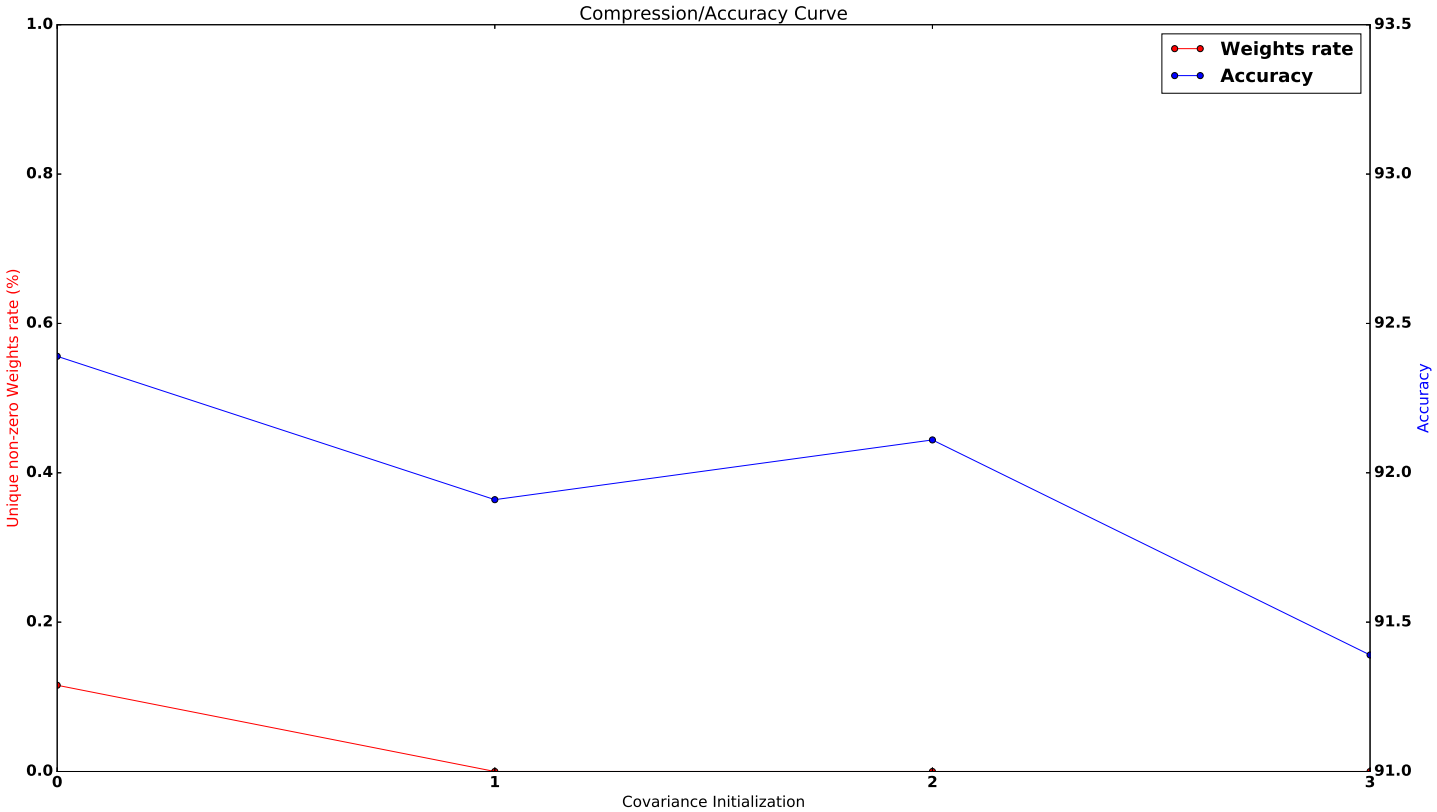


Figure 2: Compression vs Accuracy for a VGG19. unique_non-zero_weights / all_parameters

Experiments

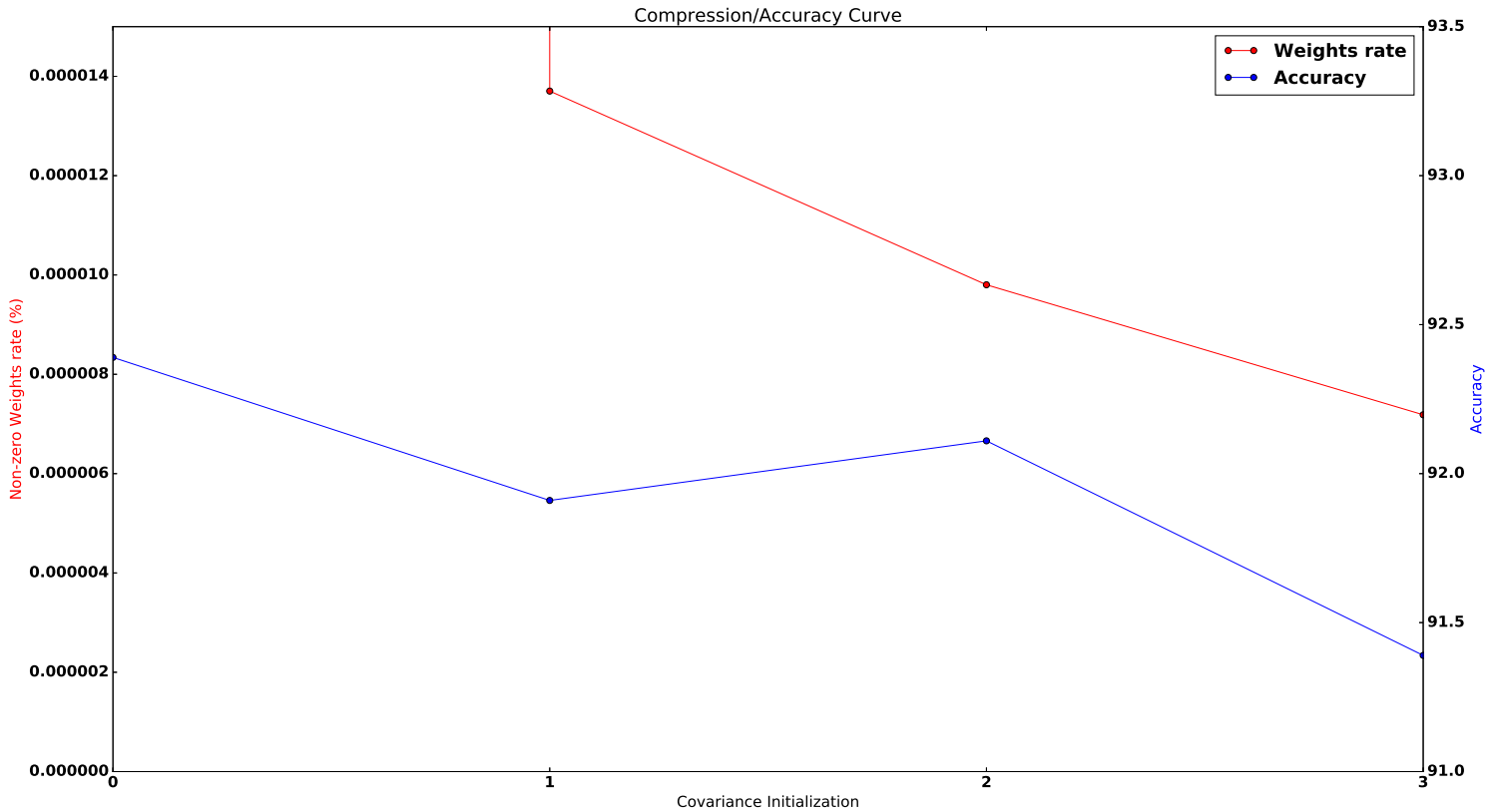
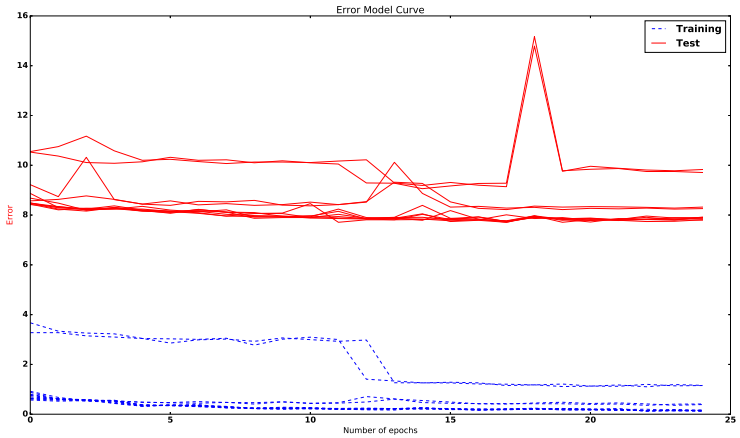
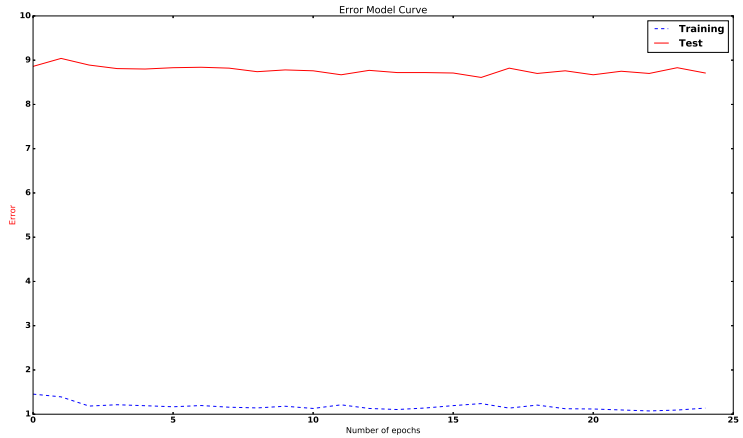


Figure 3: Compression vs Accuracy for a VGG19. all_parameters / unique_non-zero_weights

Experiments



(a)



(b)

Figure 4: Fine-tunning training after applying the weight sharing using (a) Kmeans and (b) Non-parametric

Results

```
Cluster:  35
Sharing weights in network
Points:  182   Initial Kmeans:  15    Final Kmeans:  15
iteration:    --     Points:  15    Initial number of clusters:  15    Final number of clusters:  15
Points:  2947   Initial Kmeans:  25    Final Kmeans:  25
Points:  55   Initial Kmeans:  7    Final Kmeans:  7
Points:  11423   Initial Kmeans:  15    Final Kmeans:  15
Points:  119   Initial Kmeans:  4    Final Kmeans:  4
Points:  23201   Initial Kmeans:  10    Final Kmeans:  10
Points:  126   Initial Kmeans:  4    Final Kmeans:  4
Points:  43965   Initial Kmeans:  6    Final Kmeans:  6
Points:  248   Initial Kmeans:  11    Final Kmeans:  11
Points:  74013   Initial Kmeans:  15    Final Kmeans:  15
Points:  234   Initial Kmeans:  13    Final Kmeans:  13
Points:  64329   Initial Kmeans:  20    Final Kmeans:  20
Points:  181   Initial Kmeans:  4    Final Kmeans:  4
Points:  58086   Initial Kmeans:  18    Final Kmeans:  18
Points:  179   Initial Kmeans:  11    Final Kmeans:  11
Points:  132227   Initial Kmeans:  12    Final Kmeans:  12
Points:  342   Initial Kmeans:  3    Final Kmeans:  3
Points:  283105   Initial Kmeans:  6    Final Kmeans:  6
Points:  313   Initial Kmeans:  5    Final Kmeans:  5
Points:  265757   Initial Kmeans:  11    Final Kmeans:  11
Points:  278   Initial Kmeans:  13    Final Kmeans:  13
Points:  234583   Initial Kmeans:  10    Final Kmeans:  10
Points:  276   Initial Kmeans:  10    Final Kmeans:  10
Points:  211674   Initial Kmeans:  18    Final Kmeans:  18
Points:  294   Initial Kmeans:  4    Final Kmeans:  4
Points:  206131   Initial Kmeans:  19    Final Kmeans:  19
Points:  273   Initial Kmeans:  8    Final Kmeans:  8
Points:  182582   Initial Kmeans:  22    Final Kmeans:  22
Points:  226   Initial Kmeans:  3    Final Kmeans:  3
Points:  171899   Initial Kmeans:  8    Final Kmeans:  8
Points:  317   Initial Kmeans:  2    Final Kmeans:  2
Points:  337660   Initial Kmeans:  4    Final Kmeans:  4
Points:  2260654   Initial Kmeans:  17    Final Kmeans:  17
Points:  7389   Initial Kmeans:  14    Final Kmeans:  14
After Sharing weights
 [========================== 40/40 ============================>.]  Step: 8ms | Tot: 2s386ms | Loss: 0.494 | Acc: 90.370% (9037/10000)
(debugging...) sum the absolute weight values (L1):   40895.3774551
total parameters: 38969920,
Total non-weights: 4575283,
Unique weights: 382,
Unique non-zero weights rate: 0.008349210311143596,
Unique non-zero parameters rate in model: 0.0009802432234913491
Accuracy:  90.37
```

Figure 5: Compression in each layer. Weights in convolutional layers, Batch normalization and dense layers

Metric

- Final Parameters:
  Total parameters: 38969920,
  Total non-zero weights: 4575283,
  Unique weights: 382,
  Unique non-zero weights rate (%): 0.00834921031114,
  Unique non-zero parameters rate in model (%): 0.000980243223491 3,
  Compression rate: x102015.50 times

Metric

$n = 4575283, b_1 = 32, b_2 = 8, k = 280$

$$r = \frac{nb_1}{nlog_2(k) + kb_2} = 3.93615 \tag{1}$$

Experiments

Coming back to weight pruning...

- Pruning comparison **with** and **without** L2 regularization was used:
- x8.4616 times vs x20.2394 times.

However weight sharing is a second level compression, so:

- 3.93615x8.4616 = x33.30612684 times
- 3.93615x20.2394 = x79.66531431 times

Troubles

- Why weight sharing compress to much? is it fine?
- Kmeans is the best since 1D dimension is not so complex. No information about number of cluster in thie phase.
- Questions about Equation 1 in the paper
- What does "code book" mean?
- Next Steps. Huffman coding