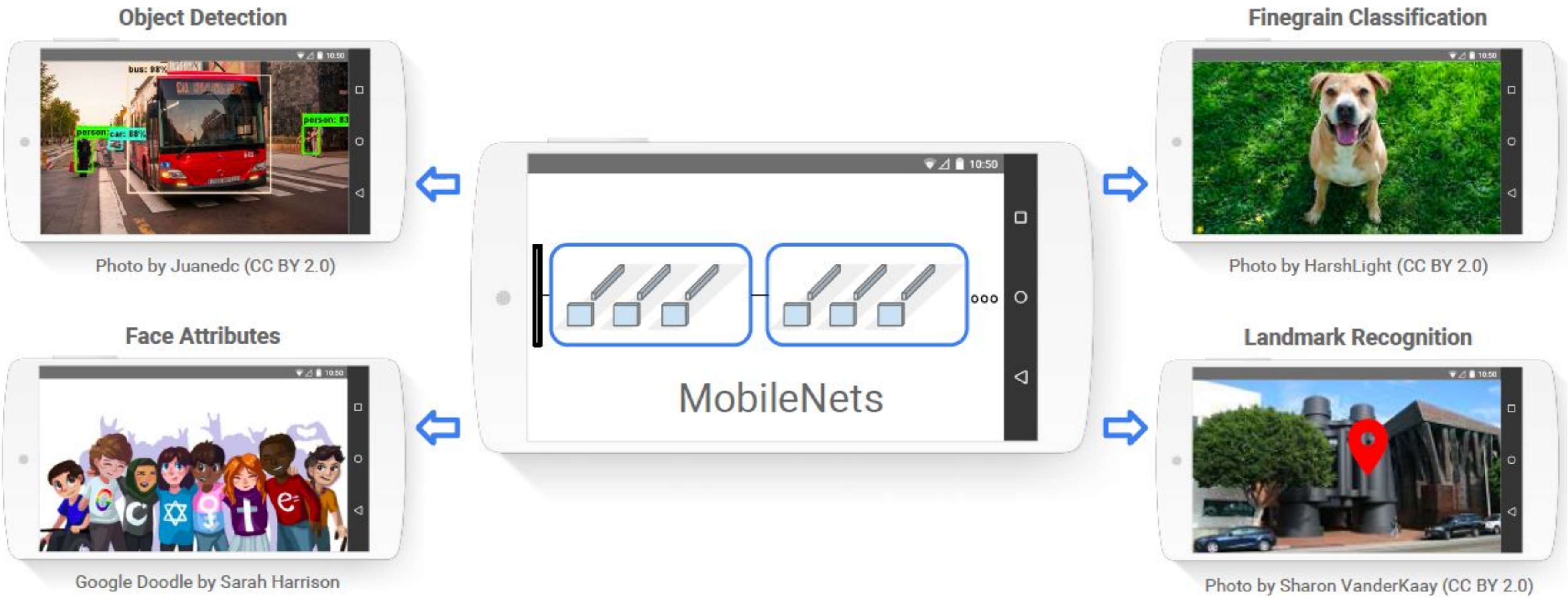
Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding

Song Han, Huizi Man, William J. Dally

Presented by Glazkova Ekaterina

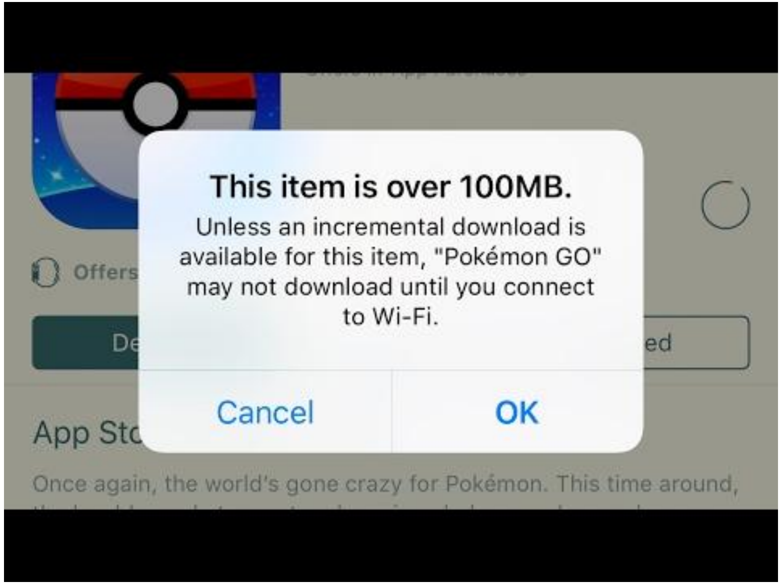
Deep Learning on Mobile. Tasks



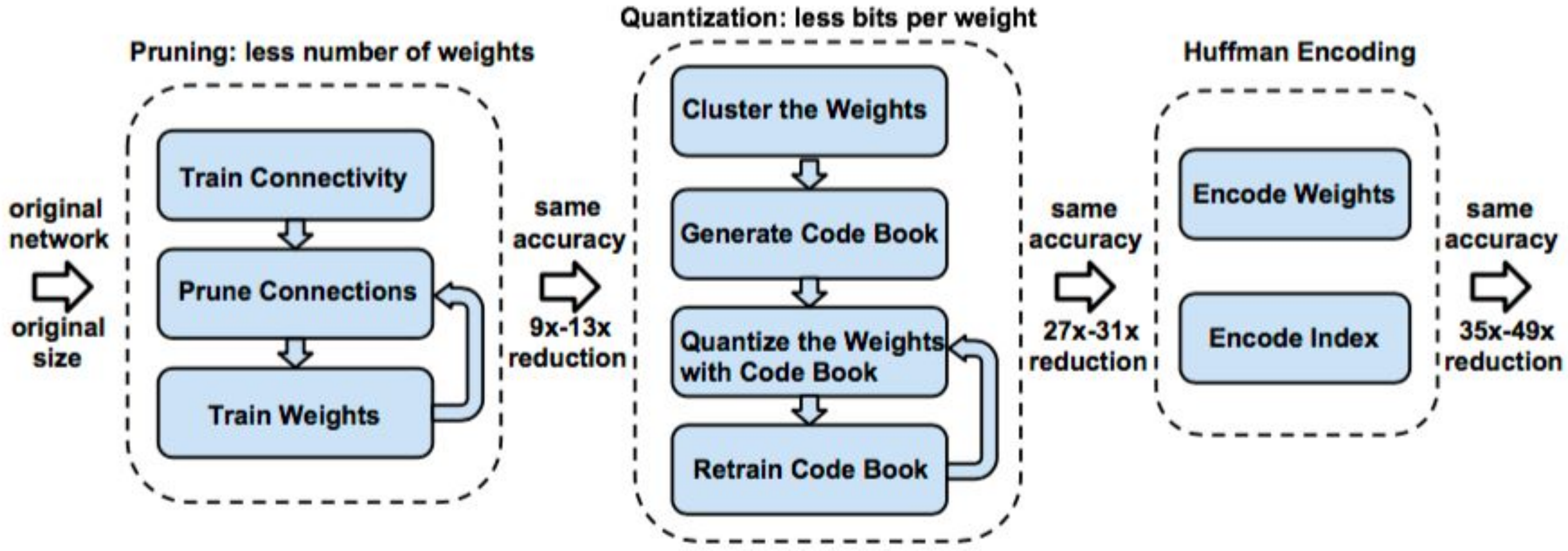
[Image Source: [Howard et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications]]

Deep Learning on Mobile. Problems

- Memory Limit
  - Device memory limits
  - App markets limits
- Energy Consumption
- Execution Time

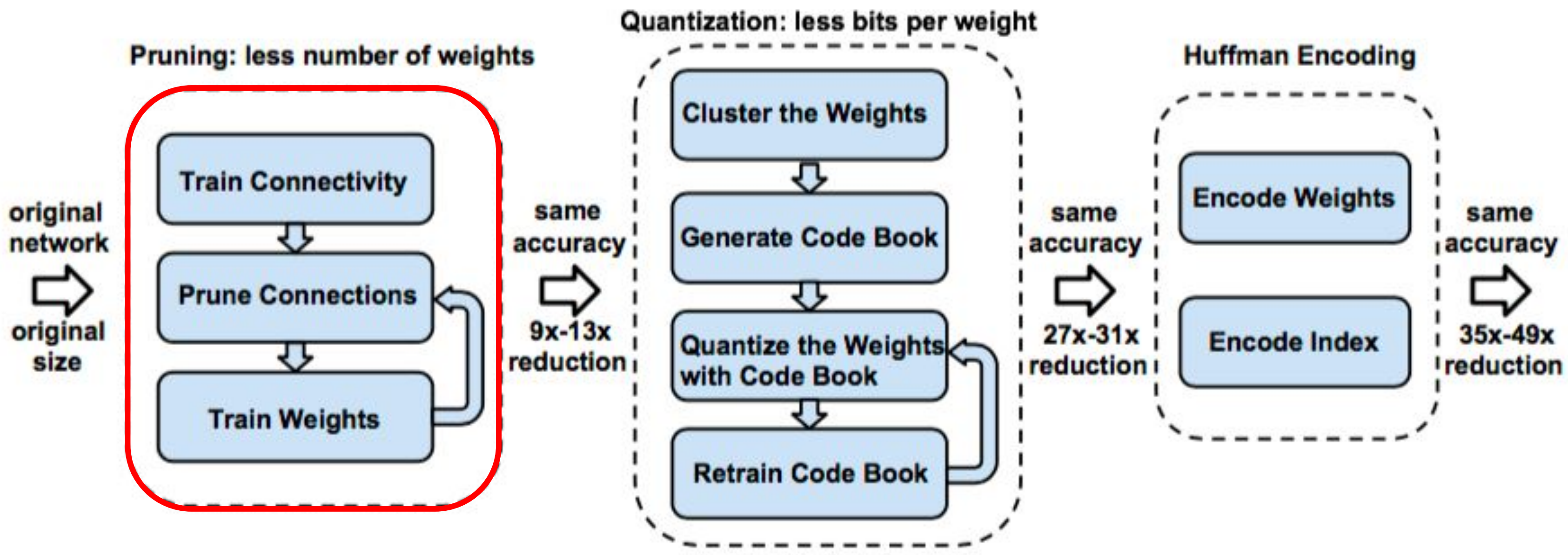


Deep Compression Pipeline



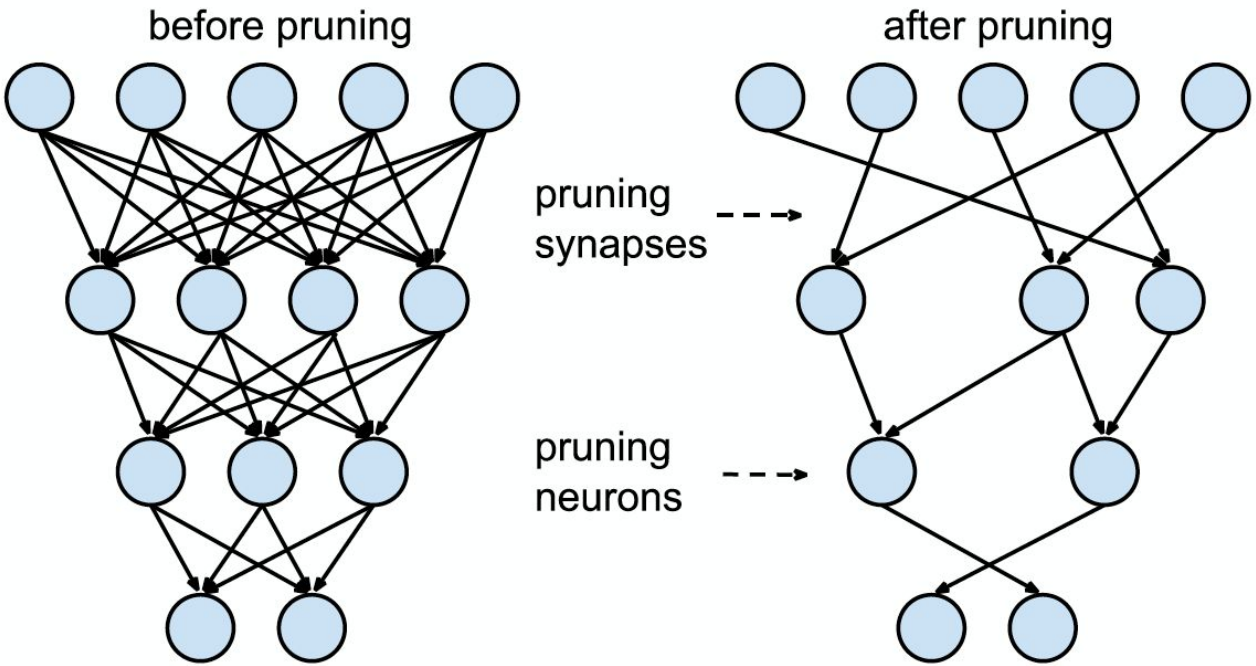
[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Deep Compression Pipeline. Pruning



[Image Source:
[Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]]

Pruning. Reduce Number of Weights



Synapses and neurons before and after pruning
[Image Source: Han et al. Learning both Weights and Connections for Efficient Neural Networks]

Pruning. Weights Representation

Span Exceeds 8=2^3

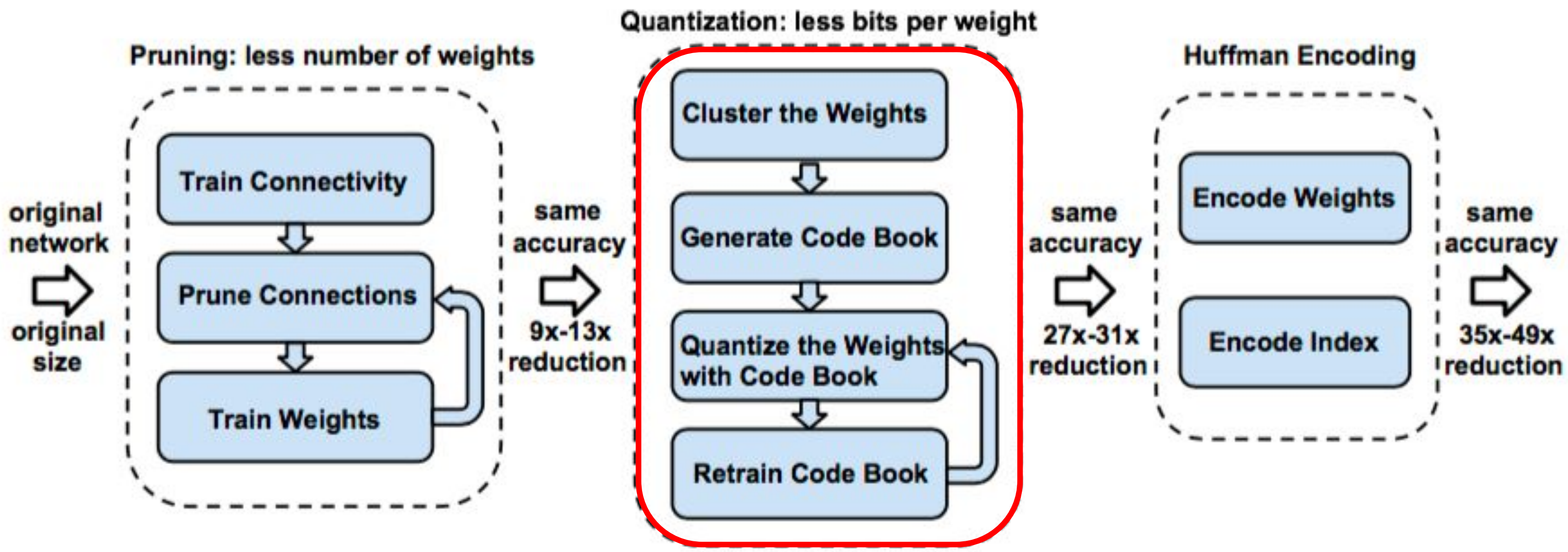
idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
diff		1			3								8			3
value		3.4			0.9								0			1.7

Filler Zero

Representing the matrix sparsity with relative index. Padding filler zero to prevent overflow

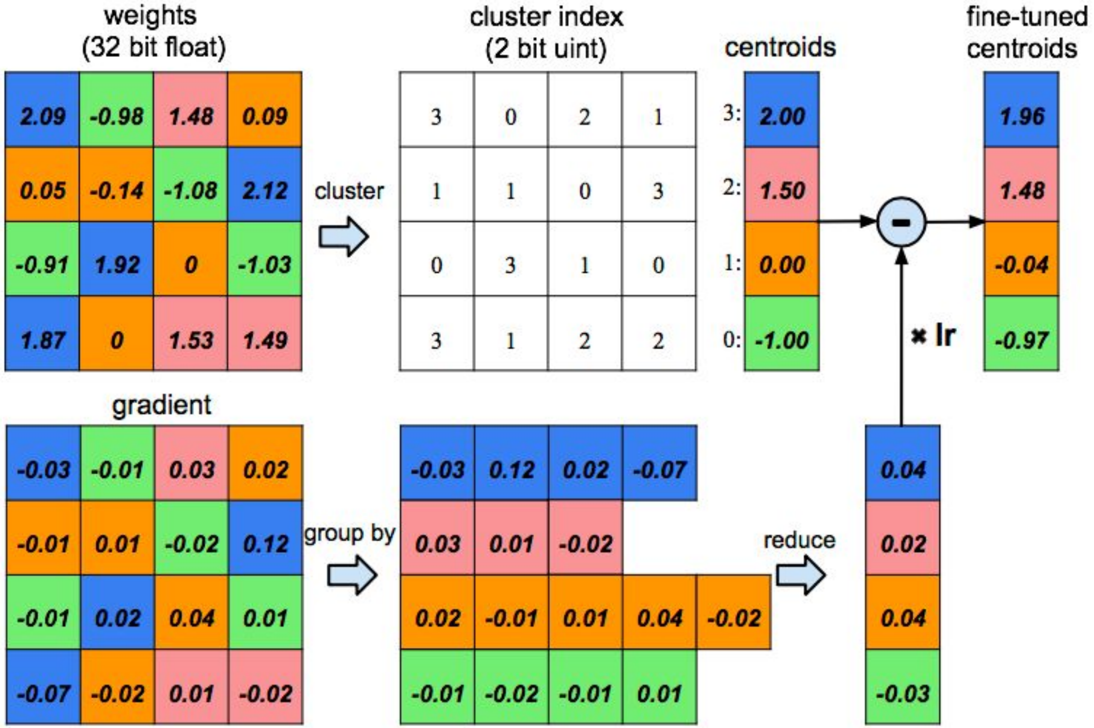
[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Deep Compression Pipeline. Quantization



[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Trained Quantization and Weight Sharing



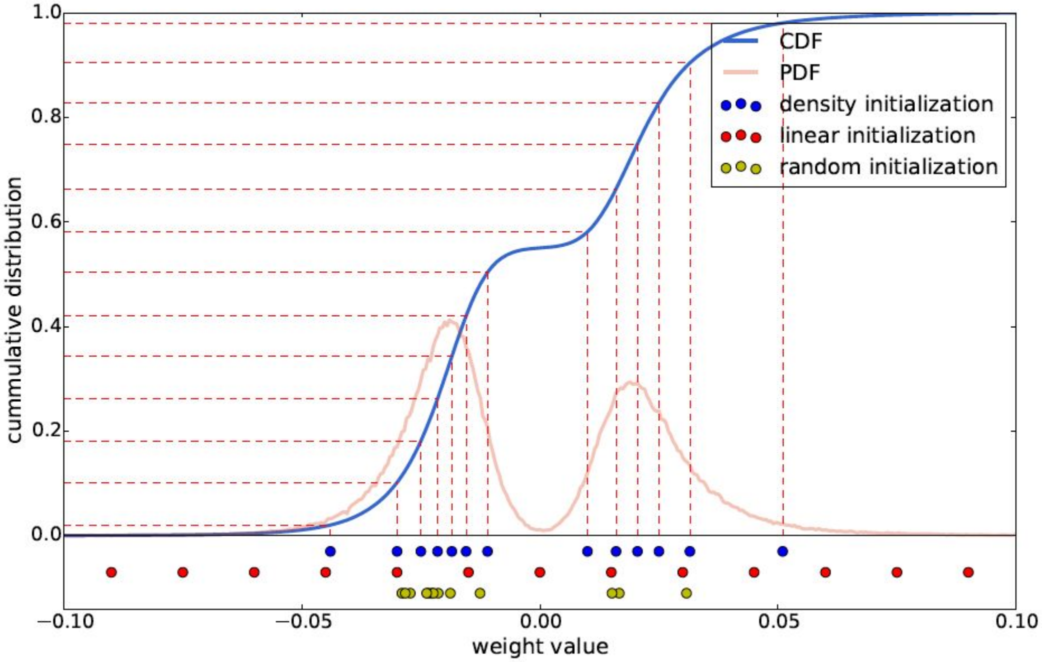
Within-cluster sum of squares (WCSS)

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{w \in c_i} |w - c_i|^2$$

[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Quantization. Initialization of Shared Weights

- Forgy (random)
- Density-based
- Linear



[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Quantization. Feed-Forward and Back-Propagation

$$\frac{\partial \mathcal{L}}{\partial C_k} = \sum_{i,j} \frac{\partial \mathcal{L}}{\partial W_{ij}} \frac{\partial W_{ij}}{\partial C_k} = \sum_{i,j} \frac{\partial \mathcal{L}}{\partial W_{ij}} \mathbb{1}(I_{ij} = k)$$

L - Loss

W - weights

C - cluster centroids

I[i,j] - centroid index of element W[i,j]

Quantization. Compression rate

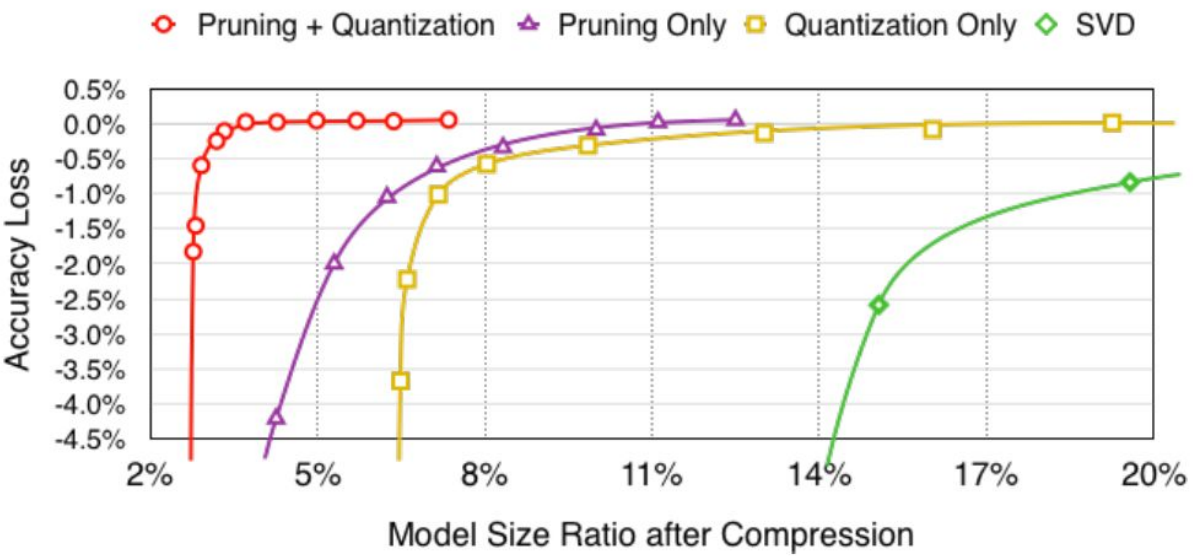
$$r = \frac{nb}{nlog_2(k) + kb}$$

n - number of connections in network

b - one connection representation (in bits)

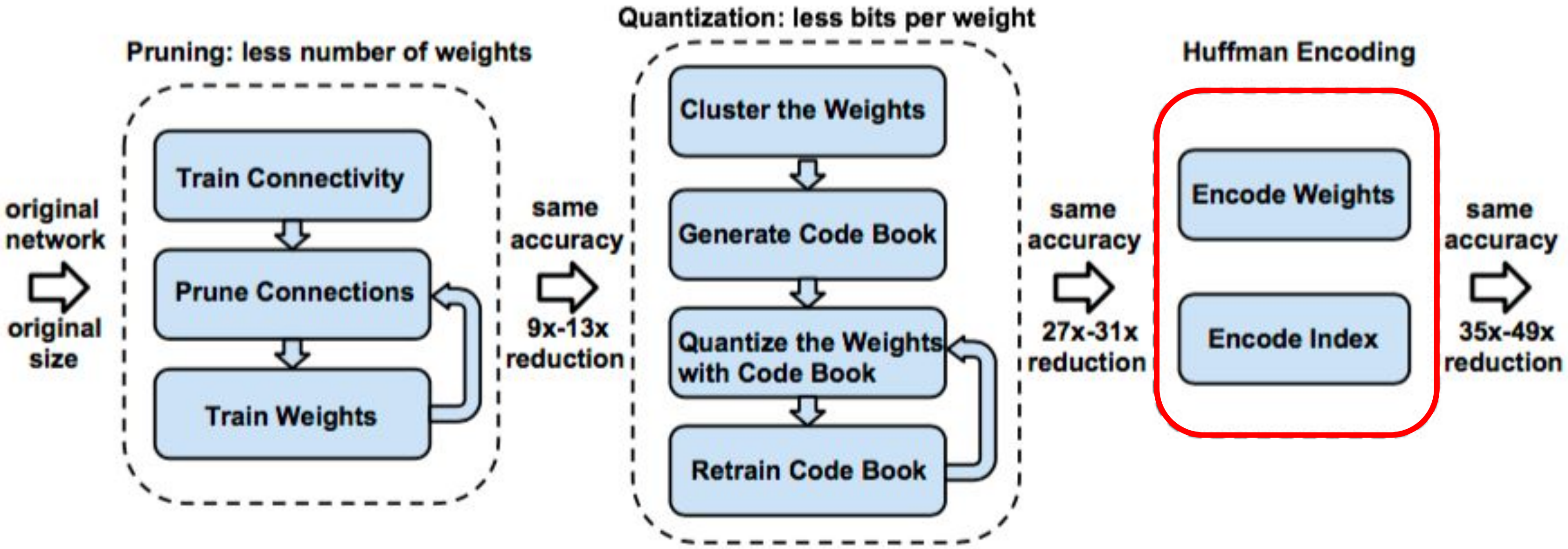
k - number of clusters

Quantization. Working Together with Pruning



[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

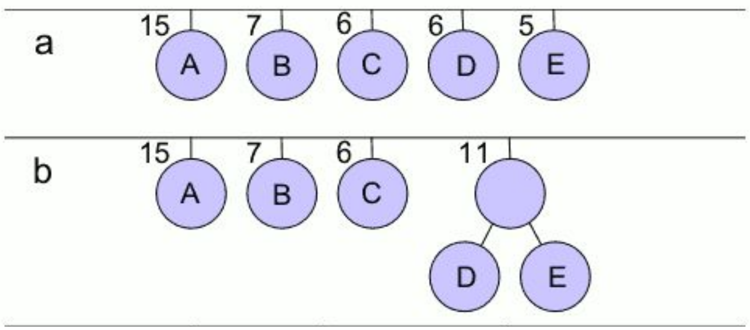
Deep Compression Pipeline. Pruning

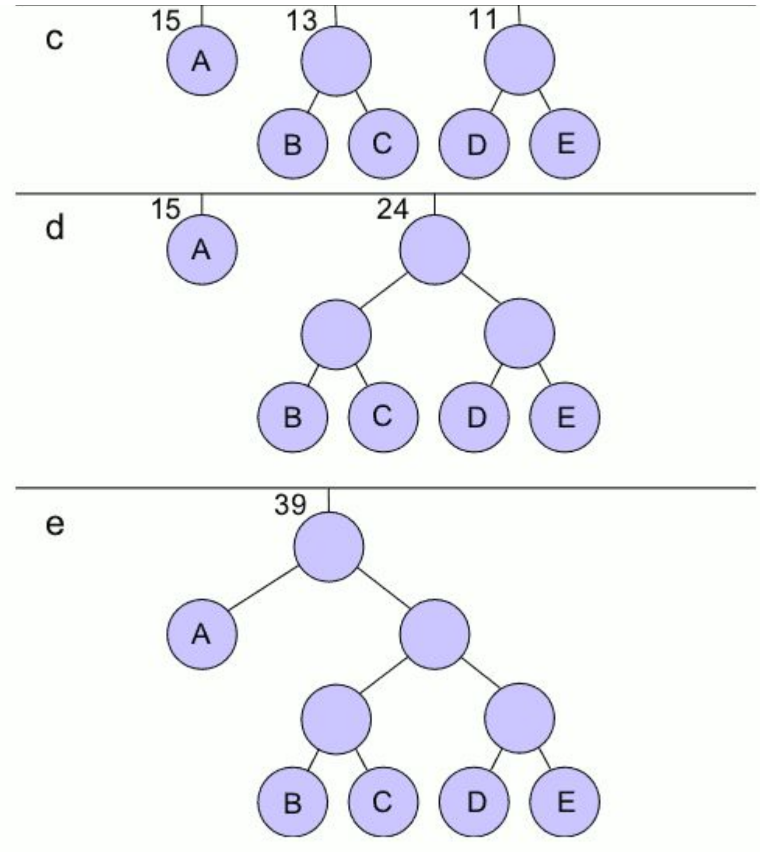


[Image Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Huffman Coding

- Optimal Prefix Code
- Codewords instead of symbols
- More common symbols represented with fewer bits





[Image Source: Wikipedia]

Huffman Coding. Weights Distribution before HC

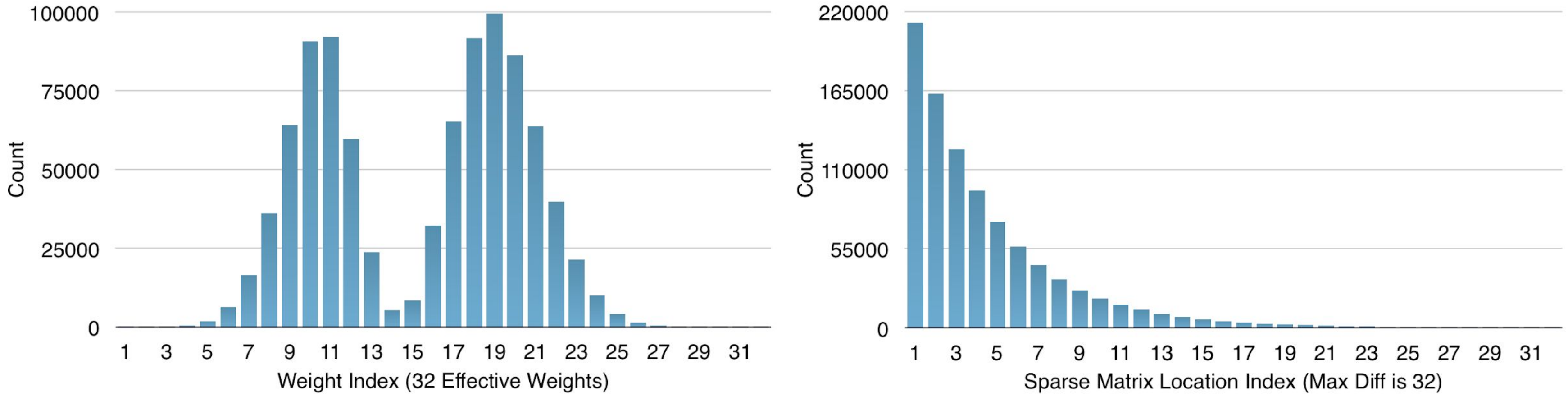


Figure 5: Distribution for weight (Left) and index (Right). The distribution is biased.

[Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Experiments. Compression Statistics. LeNet

Table 2: Compression statistics for LeNet-300-100. P: pruning, Q:quantization, H:Huffman coding.

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
ip1	235K	8%	6	4.4	5	3.7	3.1%	2.32%
ip2	30K	9%	6	4.4	5	4.3	3.8%	3.04%
ip3	1K	26%	6	4.3	5	3.2	15.7%	12.70%
Total	266K	8%(12×)	6	5.1	5	3.7	3.1% (**32**×)	2.49% (**40**×)

Table 3: Compression statistics for LeNet-5. P: pruning, Q:quantization, H:Huffman coding.

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1	0.5K	66%	8	7.2	5	1.5	78.5%	67.45%
conv2	25K	12%	8	7.2	5	3.9	6.0%	5.28%
ip1	400K	8%	5	4.5	5	4.5	2.7%	2.45%
ip2	5K	19%	5	5.2	5	3.7	6.9%	6.13%
Total	431K	8%(12×)	5.3	4.1	5	4.4	3.05% (**33**×)	2.55% (**39**×)

[Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Experiments. Compression Statistics. AlexNet

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1	35K	84%	8	6.3	4	1.2	32.6%	20.53%
conv2	307K	38%	8	5.5	4	2.3	14.5%	9.43%
conv3	885K	35%	8	5.1	4	2.6	13.1%	8.44%
conv4	663K	37%	8	5.2	4	2.5	14.1%	9.11%
conv5	442K	37%	8	5.6	4	2.5	14.0%	9.43%
fc6	38M	9%	5	3.9	4	3.2	3.0%	2.39%
fc7	17M	9%	5	3.6	4	3.7	3.0%	2.46%
fc8	4M	25%	5	4	4	3.2	7.3%	5.85%
Total	61M	11%(9×)	5.4	4	4	3.2	3.7% (**27**×)	2.88% (**35**×)

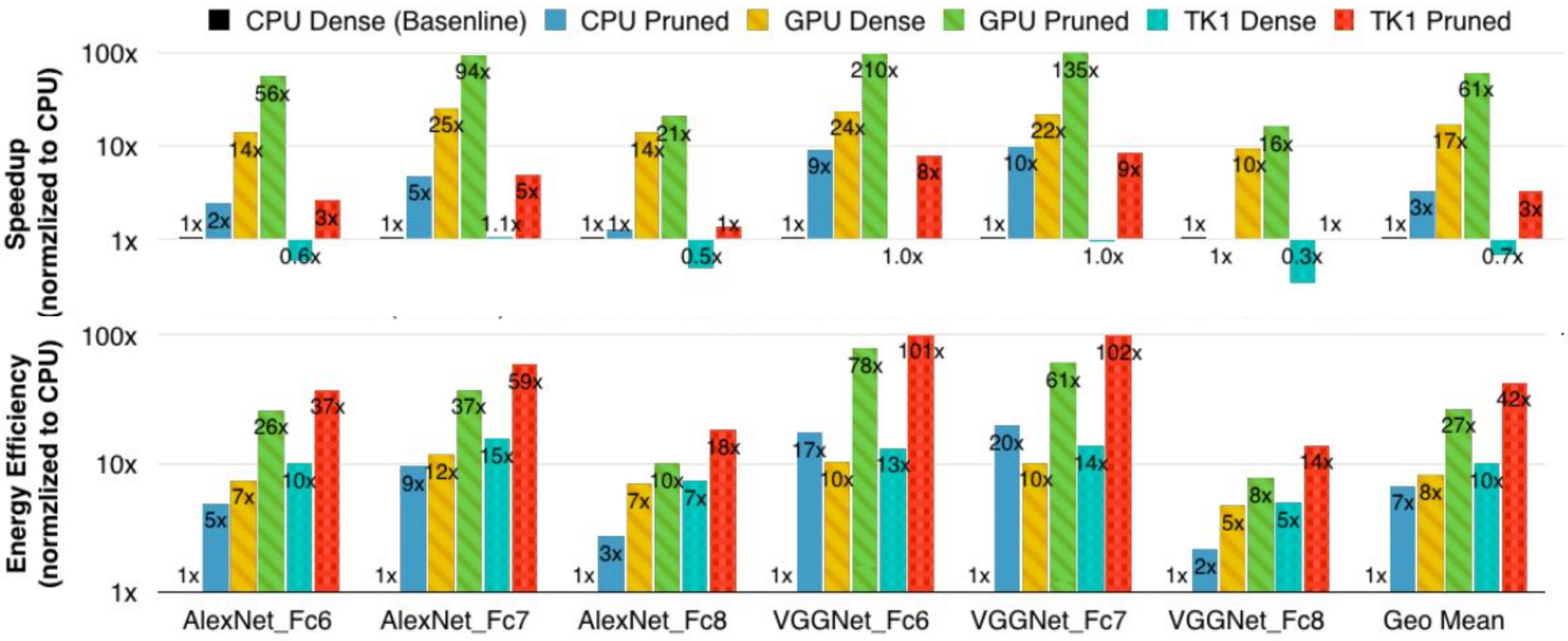
[Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Experiments. Accuracy

Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
LeNet-300-100 Ref	1.64%	-	1070 KB	
LeNet-300-100 Compressed	1.58%	-	**27 KB**	**40**×
LeNet-5 Ref	0.80%	-	1720 KB	
LeNet-5 Compressed	0.74%	-	**44 KB**	**39**×
AlexNet Ref	42.78%	19.73%	240 MB	
AlexNet Compressed	42.78%	19.70%	**6.9 MB**	**35**×
VGG-16 Ref	31.50%	11.32%	552 MB	
VGG-16 Compressed	31.17%	10.91%	**11.3 MB**	**49**×

[Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Experiments. Speedup and Energy Efficiency



[Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Experiments. Other methods on AlexNet

Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
Baseline Caffemodel (BVLC)	42.78%	19.73%	240MB	1×
Fastfood-32-AD (Yang et al., 2014)	41.93%	-	131MB	2×
Fastfood-16-AD (Yang et al., 2014)	42.90%	-	64MB	3.7×
Collins & Kohli (Collins & Kohli, 2014)	44.40%	-	61MB	4×
SVD (Denton et al., 2014)	44.02%	20.56%	47.6MB	5×
Pruning (Han et al., 2015)	42.77%	19.67%	27MB	9×
Pruning+Quantization	42.78%	19.70%	8.9MB	27×
Pruning+Quantization+Huffman	**42.78%**	**19.70%**	**6.9MB**	**35×**

[Source:
Han et al. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding]

Conclusion

- Deep Compression - 3 stage approach
- The same or better accuracy
- Weights are compressed in 35-49 times
- Model works faster and uses less energy

References

1. S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding, 2016. arXiv: 1510.00149
2. S.Han, J.Pool, J.Tran, W.Dally. Learning both Weights and Connections for Efficient Neural Networks, 2015. arXiv: 1506.02626
3. A. Howard, M. Zhu, Bo Chen, D.Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017 arXiv: 1704.04861
4. Deep compression on ICLR 2016 (Video lecture)
5. Wikipedia - Huffman coding (link)