embedded VISION SUMMIT 2018

Deep Quantization for Energy Efficient Inference at the Edge

Hoon Choi

May 23, 2018

LATTICE SEMICONDUCTOR

Agenda

- Requirement of Intelligence at Edge (IaE)
- Our solution – Deep Quantization
- Benefit of deep quantization
- Issue of deep quantization
- Deep quantization strategy
- Real world example

Intelligence at Edge - Requirements

Power/energy consumption

- Server side – unlimited power source with good cooling system
- Edge side – limited power source (e.g., battery) with min/no cooling system

Problem complexity

- Server side – complex problems, e.g., 10K classification, segmentation
- Edge side – relatively simple, e.g., 10 classification for surveillance camera

Accuracy

- Server side – expects state of the art accuracy
- Edge side – mainly works as a smart gate controlling the start of the main computation in server, which allows some degree of false positives

Intelligence at Edge – Requirements (Cont'd)

Latency

- Server side – more focused on throughput using batch process
- Edge side – mostly real time applications, thus requires low latency (tens of ms)

Cost and Size

- Server side – more focused on performance, e.g., large # of pins for huge bandwidth to DRAM
- Edge side – low cost and small form factor; limited # of pins in package, thus limited bandwidth to DRAM

Our solution – Deep Quantization

- Architectural requirements
  - Minimize/remove the access to DRAM to save power & live with small packages
  - Use energy efficient MAC equivalent operation to save power & improve latency
  - Just accurate enough to work for edge applications and save cost and power
- Need to start from a compact optimized neural network. Many methods are used together to squeeze further
  - Matrix decomposition such as singular value decomposition (weights)
  - Weight pruning (weights)
  - Quantization & compression with code book (weights)
  - Deep quantization (weights & activations)
- In this talk, we focus on the deep quantization of weights and activations

Benefit of Deep Quantization

- Resolves memory issue and improves the performance & energy efficiency
  - Reduce the storage for activations by 16x (for Binarized NN, aka. BNN, compared to 16-bit fixed point) – minimize or remove DRAM access
  - XOR and # of 1 counter instead of MAC – boosts performance by 16x and reduce the energy by 16x – energy saving & improve latency
    - 1-bit x 1-bit => XOR, SUM(1-bit's) = # of 1 counter
- Some loss of accuracy but still good for a smart gate applications
- Enables very small sized devices to cover real world problems with optimized power numbers

Benefit of Deep Quantization (Cont'd)

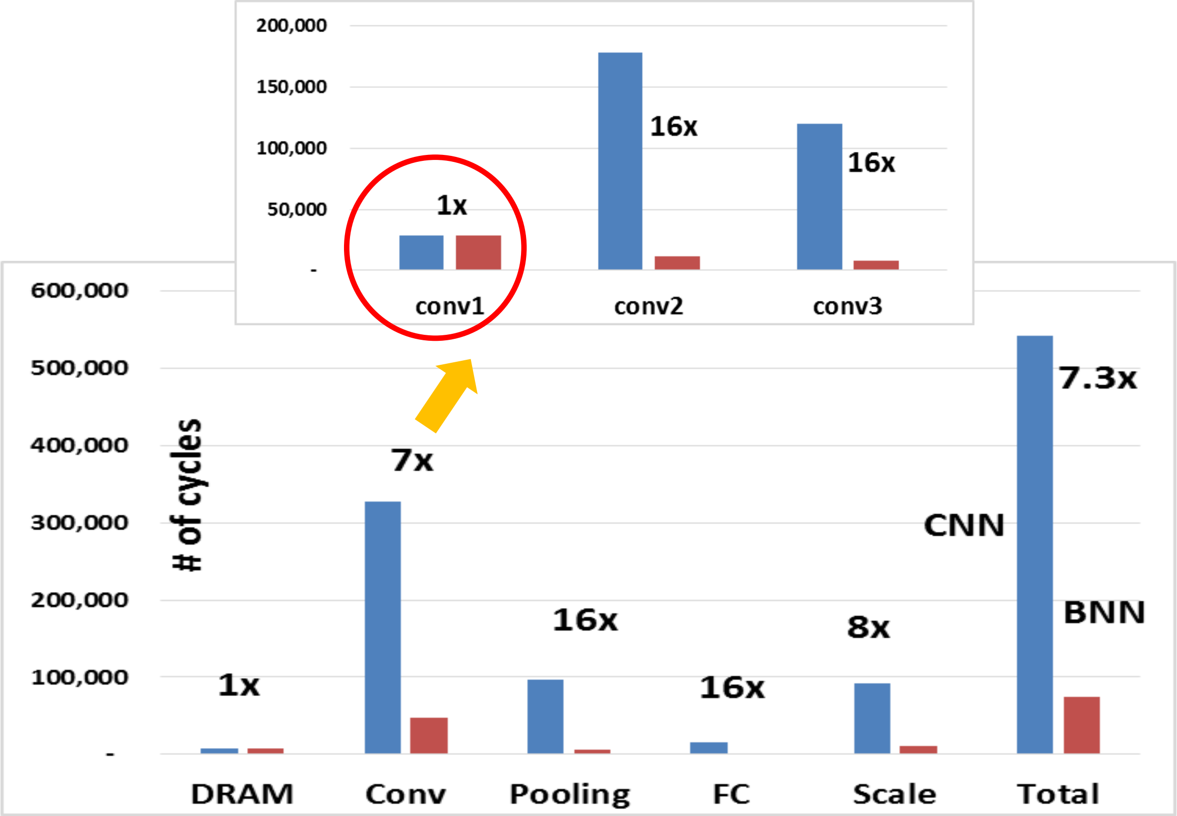
- Reduction in activation size and weight size

Application	Layers	# of MAC / XORCNT	Actiation size in KB			Weight size in KB		
			Quantizations			Quantizations		
			16b	8b	1b	16b	8b	1b
Face detection	3CBP, 1FC	16	231	116	17	233	116	15
Gender detection	3CBP, 3FC	271	1,754	877	245	22,040	11,020	1,377
Finding Waldo	8CBP	390	4,459	2,230	422	12,605	6,303	788
Car/pedestrian	8CBP, 1FC	396	4,469	2,234	423	17,285	8,642	1,080

CBP: convolution-batch normalization-activation-pooling; FC: fully connected layer

Benefit of Deep Quantization (Cont'd)

- 16x memory reduction and <u>7.3x reduction in the # of cycles & energy</u>
- More reduction as the # of layers grows

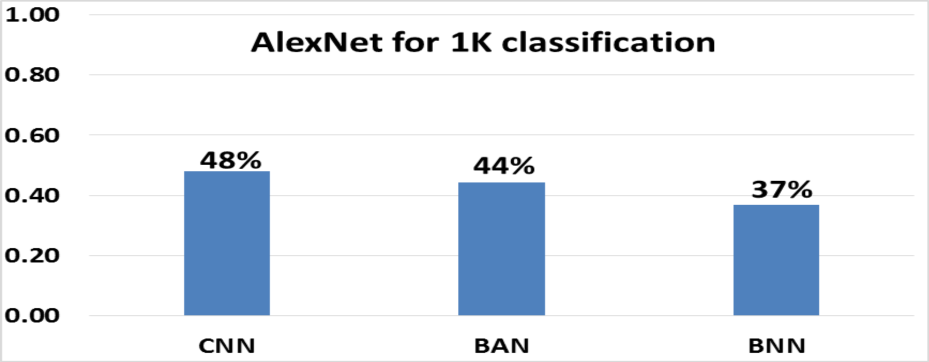


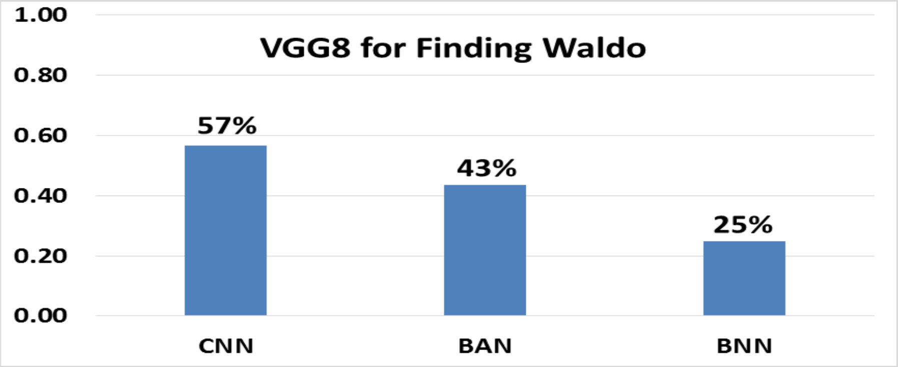
Deep Quantization issue

- Floating point to 16-bit/8-bit fixed point is a usual quantization. No special treatment is needed except performance simulation & possible minor retraining
  - In this level of quantization, each layer of quantized network mimics the behavior of original layer with some errors in numbers
- Quantization to 2-bit and 1-bit is a different story. It's not a simple quantization but a totally different network
  - A layer of the new network does not follow the behavior of the original one
  - Generally accuracy degrades and requires more layers and/or wider layer to get back the accuracy
  - Requires whole new training from scratch

Deep Quantization issue – Accuracy (Cont'd)

- For most of edge applications, problem complexity is relatively lower and simple binarization is still OK to meet the accuracy goal
  - For gender detection (3CBP+3FC), CNN=99%, BAN (Binary Activation Network; 16-bit weight, 1-bit activation)=99% and BNN=96%
- However, for complex applications and networks (e.g., 1K classifications), the accuracy of simple binarization goes down too much





Deep Quantization issue – Accuracy (Cont'd)

- Degradation of accuracy is mainly due to the severe vanishing of gradient & saturation caused by binarization activation function
  - Tanh has flat or very slow slope in both wing sides and it hinders the gradient propagation – similar to CNN before ReLu
- Our approach in training to get back the accuracy
  - Residual network – Adding residual paths is a very well known technique to help gradient back-propagation. Similar help in BNN
  - Wider network – A well known technique (wide residual network) in CNN. Similar help in BNN
  - Batch normalization – Helps isolation of each layer in training for CNN. In BNN, it also prevents/reduces the saturation
  - Wider at the first layer and higher precision at the last layer

Deep Quantization issue – Accuracy (Cont'd)

- Wider at the first layer
  - The 1st layer is the most important layer since it's the only layer that can see all the information on input data. All the following layers are deeply quantized and already lost some degree of information
  - Assign more channels to the 1st layer to keep more information for the following layers
- Higher precision at the last layer
  - The last FC layer is important especially if we are dealing with regression problem. Regression requires continuous values as output and more bits in weights helps
  - All middle layers can be optimized with deep quantization. Lost in accuracy can be compensated by more middle layers and/or thick layers

Deep Quantization issue – Accuracy (Cont'd)
- Effect of wider (first) layers and higher precision

embedded VISION SUMMIT 2018

of channels at each layer

Wider first layer

Network	1	2	3	4	5	6	7	8
Conv	64	64	64	32	64	128	64	128
Conv	64	64	32	64	128	64	32	64
Pool								
Conv	128	64	64	32	48	48	128	48
Conv	128	64	128	64	64	96	64	96
Pool								
Conv	256	64	32	32	32	32	32	32
Conv	256	64	16	16	16	16	16	16
Pool								
Conv	64	32	32	16				
Conv	32	16	16	8				
Pool								
FC				1b weight				16b weight
Accuracy	89	84	74	56	78	83	81	90

Wider layers

Higher precision at FC

Deep Quantization issue – Accuracy (Cont'd)
- Effect of wider, deeper, residual path and precision

embedded VISION SUMMIT 2018

CIFAR 100 test

Network	Accuracy (%)	Computation time (%)	Memory usage(%)	
F-6-m1-k1	100	100	100	Reference point base line 6 layer CNN
B-6-m1-k1	20	6	6	6 layer BNN after simple binarization
B-6-m1-k3	33	56	19	Effect of wider layers (3x, 5x, 8x, and 10x)
B-6-m1-k5	56	156	31	
B-6-m1-k8	80	400	50	Fast increase in computation cost
B-6-m1-k10	100	625	63	
B-12-m1-k1	24	13	6	Deeper (2x) without residual; 3% gain over 6 layer
B-12-m1-k1-r	33	13	6	Residual helps accuracy; 10% gain over 6 layer
B-12-m1-k2-r	60	50	13	Effect of wider layers (2x and 4x)
B-12-m1-k4-r	80	200	25	
B-12-m2-k2-r-fl	80	175	13	Last layer with 16b weight; about 15% extra gain
B-18-m2-k2-r-fl	88	200	13	Effect of deeper network with residual
B-24-m2-k2-r-fl	96	225	13	
B-30-m2-k2-r-fl	103	250	13	

F = 16-bit CNN, B = BNN, number = # of layers, m = 1st layer width multiplier, k = other layer width multiplier, r = residual path, fl = last layer with 16-bit weight

All numbers are relative to F-6-m1-k1

Deep Quantization Strategy

- **Quantization priority**: Quantize to make “input activation for conv. layer fit inside, output activation of each layer fit inside, and then weight”
  - Input activation for each conv. layer is used again and again, thus keep it inside of chip is important to reduce the DRAM access
- **Priority among more bits (higher precision), more layers (depth), and width**: more layers, more bits, and then wider layer
  - More layers: no change in layer by layer memory, computation time goes up
  - More bits for activation: memory & computation time go up, no more XOR + Counter
  - Wider layer: memory increase by n, computation time by n^2

Deep Quantization Strategy (Cont'd)

- **BNN vs. small (shallow and narrow) CNN**
  - BNN is not yet an cure-all for all edge applications
  - Though it becomes better and better, still for some applications it's much harder to meet the accuracy with BNN and it results in a much wider and deeper network
  - So, in most cases, we need to try both of BNN and CNN and select one per the power, performance, and size requirement
    - Applications especially that do not require much abstractions, a shallow and narrow CNN can be good enough in accuracy and small enough for small sized devices

Real World Example – Face detection

- BNN; 2 class classification (face vs. no-face); 32x32 RGB (32x32x3)
- 16M MACs becomes 16M of XOR & # of one counting operations
- 26KB of memory: all weights + MAX (one layer's in & out activations)
  - 16b case ~ 400KB; 8b case ~ 200KB

Face detection

Face det	MAC	Activation		Weight	
Layers	# (M)	# (K)	Mem (KB)	# (K)	Mem (KB)
Input		3	3		
Conv1	2	66	8	2	0.22
Pool1		16	2		
Conv2	9	16	2	37	4.61
Pool2		4	1		
Conv3	5	8	1	74	9.22
Pool3		2	0		
FC9	0	0	0	4	0.51
Total	16	116	17	116	15

Real World Example – Human detection

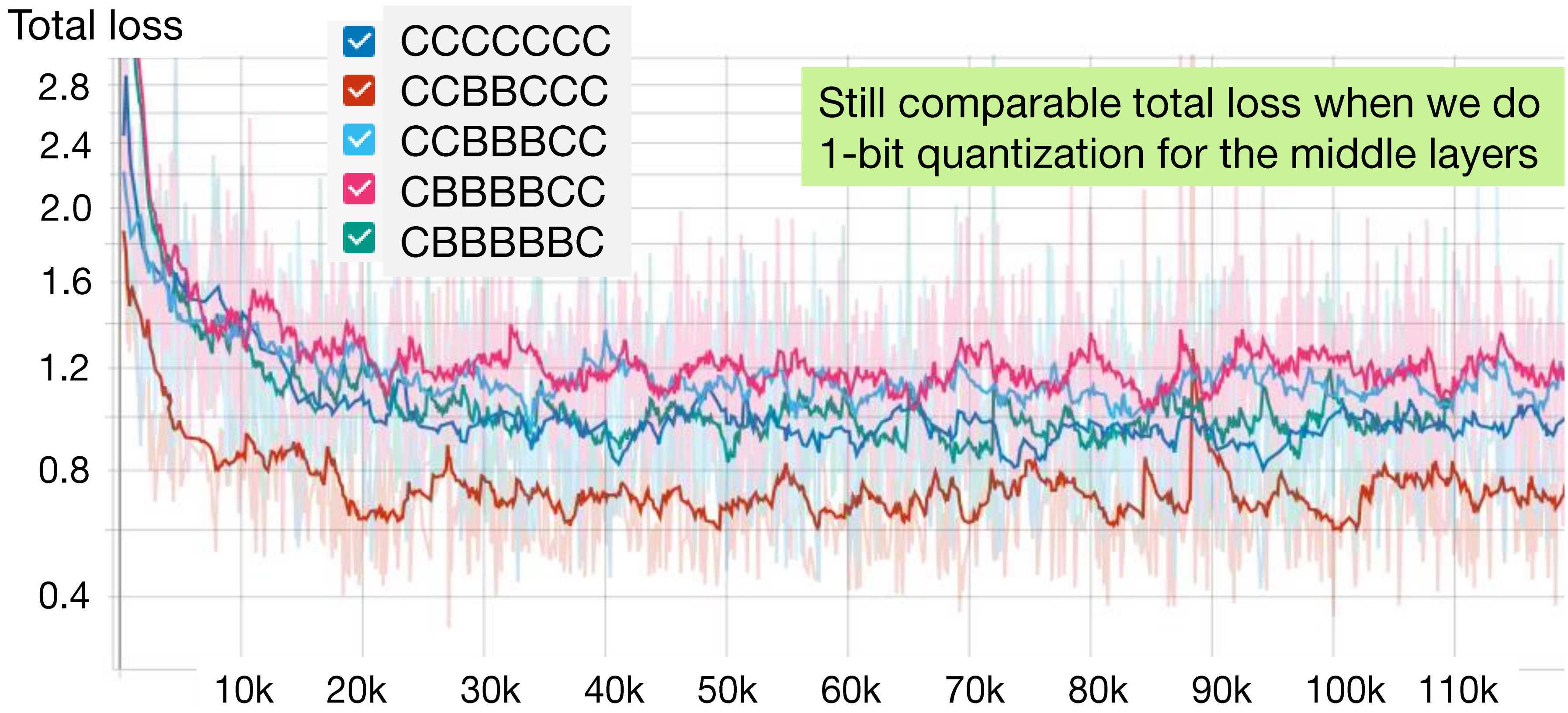
- VGG type 7 convolution layers and 4 pooling layers
- Mix of 16-bit quantized layers and 1-bit quantized layers
  - 5 different mixes with different memory sizes for weights
  - Activation size is determined by the 1st layer in this case

	Activation size (KB)		Weight size (KB)	
	Quantization		Quantization	
Layer	16b	1b	16b	1b
Conv	2048	128	3.4	0.2
Pool	512	32		
Conv	512	32	72	4.5
Conv	512	32	72	4.5
Pool	128	8		
Conv	256	16	144	9
Conv	256	16	288	18
Pool	64	4		
Conv	128	8	576	36
Pool	32	2		
Conv	6	0.4	216	13.5
Total	**4454**	**278**	**1371**	**86**

Quant of layer							Weight size (KB)
1	2	3	4	5	6	7	
C	C	C	C	C	C	C	1371
C	C	B	B	C	C	C	1169
C	C	B	B	B	C	C	899
C	B	B	B	B	C	C	831
C	B	B	B	B	B	C	291

C: 16-bit quantization, B: 1-bit

Real World Example – Human detection (Cont'd)

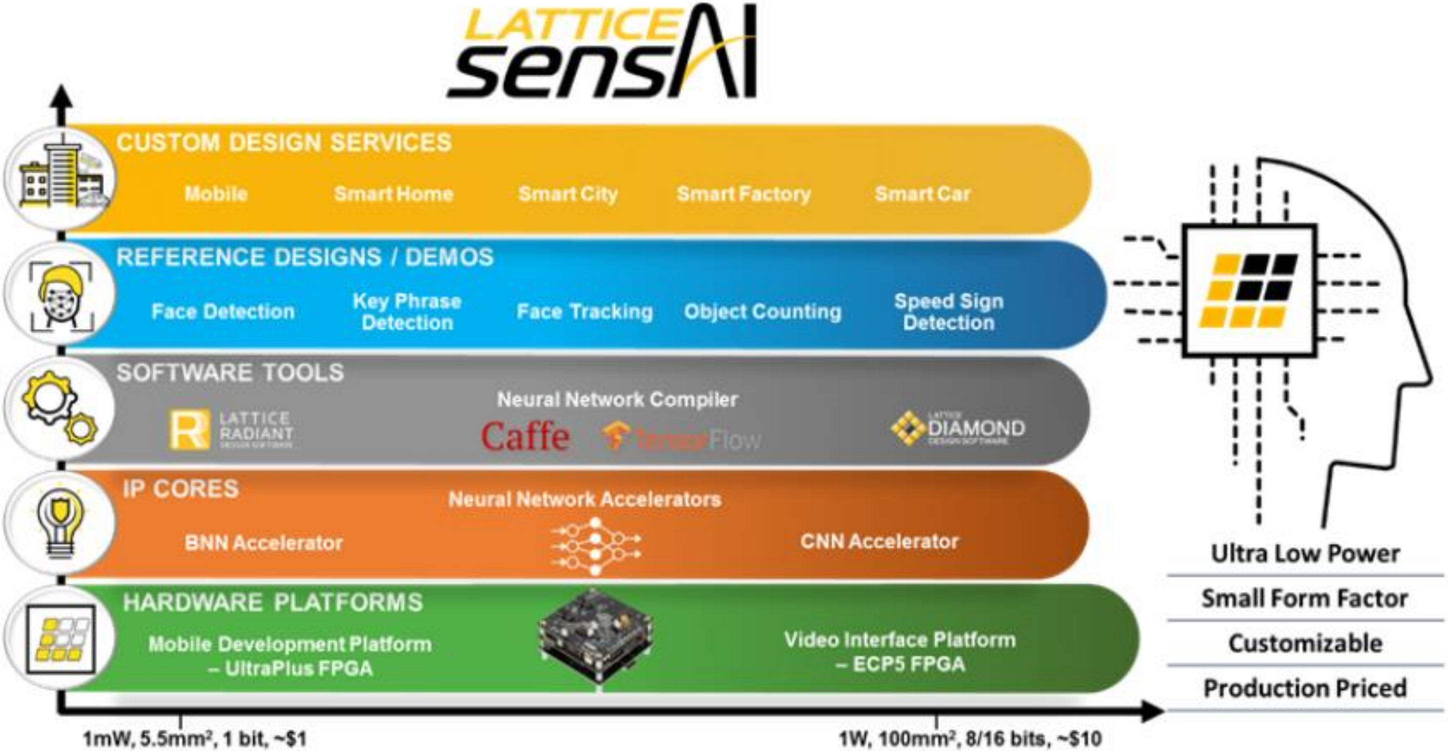


Still comparable total loss when we do 1-bit quantization for the middle layers

Conclusion

- Intelligence at the edge requires low power, low cost, small latency, and small form factor, and reducing the access to external memory is one of the key factor to achieve the goal
- We successfully use deep quantization for IaE applications to achieve the goal with small sized devices
- Accuracy recoup is possible by various techniques including network topology changes such as more layers, wider layer, residual path, etc.
- Deep quantization including BNN is a valuable technology for the edge applications

LATTICE SensAI



Resources

- *Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1*
- *Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations*
- *XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks*
- *A 7.663-TOPS 8.2-W Energy-efficient FPGA Accelerator for Binary Convolutional Neural Networks*
- *Embedded Binarized Neural Networks*

Appendix

Benefit of Deep Quantization (Cont'd)

VGG8

Finding Waldo

VGG8	MAC	Activation (mem in KB)				Weight (mem in KB)			
Layers	# (M)	# (K)	16b	8b	1b	# (K)	16b	8b	1b
Input		164	328	164	164				
Conv1	24	874	1,749	874	109	0	1	0	0
Bn1	1					0	0	0	
Pool1		201	401	201	25				
Conv2	58	401	803	401	50	5	9	5	1
Bn2	0					0	0	0	0
Pool2		100	201	100	13				
Conv3	58	201	401	201	25	18	37	18	2
Bn3	0					0	1	0	0
Pool3		50	100	50	6				
Conv4	58	100	201	100	13	74	147	74	9
Bn4	0					1	1	1	0
Pool4		25	50	25	3				
Conv5	58	50	100	50	6	295	590	295	37
Bn5	0					1	2	1	0
Pool5		13	25	13	2				
Conv6	58	25	50	25	3	1,180	2,359	1,180	147
Bn6	0					2	4	2	0
Pool6		8	16	8	1				
Conv7	75	16	33	16	2	4,719	9,437	4,719	590
Bn7	0					4	8	4	1
Conv8	0	0	0	0	0	4	8	4	1
Pool8	0	0	0	0	0				
Total	390	2,230	4,459	2,230	422	6,303	12,605	6,303	788

Benefit of Deep Quantization (Cont'd)

uYolo

Object detection (car & human) w/ bounding box

uYolo	MAC	Activation (mem in KB)				Weight (mem in KB)			
Layers	# (M)	# (K)	16b	8b	1b	# (K)	16b	8b	1b
Input		164	328	164	164				
Conv1	24	874	1,749	874	109	0	1	0	0
Bn1	1					0	0	0	0
Pool1		201	401	201	25				
Conv2	58	401	803	401	50	5	9	5	1
Bn2	0					0	0	0	0
Pool2		100	201	100	13				
Conv3	58	201	401	201	25	18	37	18	2
Bn3	0					0	1	0	0
Pool3		50	100	50	6				
Conv4	58	100	201	100	13	8	16	8	1
Bn4	0					1	1	1	0
Pool4		25	50	25	3				
Conv5	58	50	100	50	6	33	66	33	4
Bn5	0					1	2	1	0
Pool5		13	25	13	2				
Conv6	58	25	50	25	3	1,180	2,359	1,180	147
Bn6	0					2	4	2	0
Pool6		8	16	8	1				
Conv7	75	16	33	16	2	4,719	9,437	4,719	590
Bn7	0					4	8	4	1
Conv8	4	4	8	4	1	262	524	262	33
Bn8	0					1	2	1	0
FC9	2	1	1	1	0	2,408	4,817	2,408	301
Total	396	2,234	4,469	2,234	423	8,642	17,285	8,642	1,080

Benefit of Deep Quantization (Cont'd)

Gender detection

CaffeNet	MAC	Activation (mem in KB)				Weight (mem in KB)			
Layers	# (M)	# (K)	16b	8b	1b	# (K)	16b	8b	1b
Input		155	309	155	155				
Conv1	44	301	602	301	38	14	28	14	2
Bn1	0					0	1	0	
Pool1		75	151	75	9				
Conv2	173	201	401	201	25	221	442	221	28
Bn2	0					1	2	1	0
Pool2		50	100	50	6				
Conv3	43	75	151	75	9	885	1,769	885	111
Bn3	0					2	3	2	0
Pool5		19	38	19	2				
FC6	10	1	1	1	0	9,634	19,268	9,634	1,204
FC7	0	1	1	1	0	262	524	262	33
FC8	0	0	0	0	0	1	2	1	0
Total	271	877	1,754	877	245	11,020	22,040	11,020	1,377

Face detection

Face det	MAC	Activation (mem in KB)				Weight (mem in KB)			
Layers	# (M)	# (K)	16b	8b	1b	# (K)	16b	8b	1b
Input		3	6	3	3				
Conv1	2	66	131	66	8	2	3.46	2	0.22
Pool1		16	33	16	2				
Conv2	9	16	33	16	2	37	73.73	37	4.61
Pool2		4	8	4	1				
Conv3	5	8	16	8	1	74	147.46	74	9.22
Pool3		2	4	2	0				
FC9	0	0	0	0	0	4	8.19	4	0.51
Total	16	116	231	116	17	116	233	116	15