Staged Training to ternarize neural network weights

Kenji Ogura

April 01 2020

1 Abstract

For small device or Embedded systems without GPU, realtime object detection task via Neural Networks should be worked. I was inspired below papers,

- Training a Binary Weight Object Detector by-Knowledge Transfer for Autonomous Driving[1]
- Ternary weight networks[2]
- XNOR-Net: ImageNet Classification Using BinaryConvolutional Neural Networks[3]

Generally the inference task using full ternary weights -1,0,+1 is considered as low accuracy than full precision weights. But for mobile device such as raspberryPI small weights will be efficiency choice. Many quantization methods for model weights are proposed now such as FP16, bfloat, fixed point 16bits, 8bit, ternary 2bits and XNor 1bit too. I consider that re-training after quatization of weights is needed. How to train using some quantization methods? from ground, finetune?

2 Staged training method

In this paper I propose the staged training for yolov2-voc.cfg, yolov3-voc.cfg on Darknet website[4]. You can get full ternarized weights within 3 points accuracy drops for yolov2, within 4.5 points accuracy dorops for yolov3 using this method. 2bits ternary weights representaion may be x16 smaller than 32bit floating point.

Staged training generates Ternarized weights for yolov2-voc, yolov3-voc. This method sprits a training step into 4 stages.

- Stage-0 : first few layers are ternarized.
- Stage-1 : last few layers are ternarized.
- Stage-2 : all layers without last layer are ternarized.
- Stage-3 : full ternarized.

Weights used on each stages is imported from previous stage, such as stage-2 weights from stage-1. Figure.1 shows staging for yolov2-voc.cfg into 4 training stages, and Figure.2 for yolov3-voc.cfg. On each figures F denote full precision weights and T denotes ternarized weights. To estimate this staged training method I implement ternary keyword for each convolution layer in cfg file like 'ternary=1' and I also support ternarizing weights function and ternarized weights memory area in darknet framework.

I trained yolov2-voc.cfg on 4 jobs, and checked each loss curves on Excell graph. And I use AlexeyAB[5] darknet to get mAP of experiments. In fact, our experience denotes that accuracy via ternary weights drops about 4 points against full precision weights inferece.

3 Result of Training

Tables denote the result of my staged training. VOC 2012, 2007 Dataset is used with training and VOC 2007 for estimate of mAP. Number of iteration is 41000 to 100400, steps x0.1 at 80%, 90%, lr=0.001, burnin=1000 on Darknet framework.

Iteration 41000(2000/class), steps x0.1 80% 90%, lr=0.001 at all stages

no	layer	filters	size	input -> output	M0	M1	M2	M3
0	conv	32	3x3/1	416x416x3->416x416x32	F	F	F	T
1	max	-	2x2/2	416x416x32->208x208x32				
2	conv	64	3x3/1	208x208x32->208x208x64	F	F	T	T
3	max	-	2x2/2	208x208x64->104x104x64				
4	conv	128	3x3/1	104x104x64->104x104x128	F	F	T	T
5	conv	64	1x1/1	104x104x128->104x104x64	F	F	T	T
6	conv	128	3x3/1	104x104x64->104x104x128	F	F	T	T
7	max	-	2x2/2	104x104x128->52x52x128				
8	conv	256	3x3/1	52x52x128->52x52x256	F	F	T	T
9	conv	128	1x1/1	52x52x256->52x52x128	F	F	T	T
10	conv	256	3x3/1	52x52x128->52x52x256	F	F	T	T
11	max	-	2x2/2	52x52x256->26x26x256				
12	conv	512	3x3/1	26x26x256->26x26x512	F	T	T	T
13	conv	256	1x1/1	26x26x512->26x26x256	F	T	T	T
14	conv	512	3x3/1	26x26x256->26x26x512	F	T	T	T
15	conv	256	1x1/1	26x26x512->26x26x256	F	T	T	T
16	conv	512	3x3/1	26x26x256->26x26x512	F	T	T	T
17	max	-	2x2/2	26x26x512->13x13x512				
18	conv	1024	3x3/1	13x13x512->13x13x1024	F	T	T	T
19	conv	512	1x1/1	13x13x1024->13x13x512	F	T	T	T
20	conv	1024	3x3/1	13x13x512->13x13x1024	F	T	T	T
21	conv	512	1x1/1	13x13x1024->13x13x512	F	T	T	T
22	conv	1024	3x3/1	13x13x512->13x13x1024	F	T	T	T
23	conv	1024	3x3/1	13x13x1024->13x13x1024	T	T	T	T
24	conv	1024	3x3/1	13x13x1024->13x13x1024	T	T	T	T
25	route	16	-	-				
26	conv	64	1x1/1	26x26x512->26x26x64	T	T	T	T
27	reorg/2			26x26x64->13x13x256				
28	route	27	24	-				
29	conv	1024	3x3/1	13x13x1280->13x13x1024	T	T	T	T
30	conv	125	1x1/1	13x13x1024->13x13x125	F	F	F	T
31	detection	-	-	-				

Figure 1: Staging for yolov2-voc.cfg

no	layer	filters	size	input->output	M0	M1	M2	M3
0	conv	32	3x3/1	416x416x3->416x416x32	T	T	T	T
-	-	-	-	-	T	T	T	T
79	conv	512	1x1/1	13x13x1024->13x13x512	T	T	T	T
80	conv	1024	3x3/1	13x13x512->13x13x1024	F	F	F	T
81	conv	75	1x1/1	13x13x1024->13x13x75	F	F	F	T
82	yolo	-	-	-				
83	route	79	-	-				
84	conv	256	1x1/1	13x13x512->13x13x256	F	T	T	T
85	upsample			2x13x13x256->26x26x256				
86	route	85	61	-	F	T	T	T
87	conv	256	1x1/1	26x26x768->26x26x256	F	T	T	T
88	conv	512	3x3/1	26x26x256->26x26x512	F	T	T	T
89	conv	256	1x1/1	26x26x512->26x26x256	F	T	T	T
90	conv	512	3x3/1	26x26x256->26x26x512	F	T	T	T
91	conv	256	1x1/1	26x26x512->26x26x256	F	T	T	T
92	conv	512	3x3/1	26x26x256->26x26x512	F	F	F	T
93	conv	75	1x1/1	26x26x512->26x26x75	F	F	F	T
94	yolo	-	-	-				
95	route	91	-	-				
96	conv	128	1x1/1	26x26x256->26x26x128	F	T	T	T
97	upsample		F	2x26x26x128->52x52x128				
98	route	97	36	-				
99	conv	128	1x1/1	52x52x384->52x52x128	F	F	T	T
100	conv	256	3x3/1	52x52x128->52x52x256	F	F	T	T
101	conv	128	1x1/1	52x52x256->52x52x128	F	F	T	T
102	conv	256	3x3/1	52x52x128->52x52x256	F	F	T	T
103	conv	128	1x1/1	52x52x256->52x52x128	F	F	T	T
104	conv	256	3x3/1	52x52x128->52x52x256	F	F	F	T
105	conv	75	1x1/1	52x52x256->52x52x75	F	F	F	T
106	yolo	-	-	-				

Figure 2: Staging for yolov3-voc.cfg

Stage	mAP	IOU	comments
-	76.85	54.67	official Weights
M0	77.09	57.04	-
M1	76.44	56.18	-
M2	75.06	57.71	-
M3	73.82	54.90	full ternary

Table 1: result regard to yolov2

Stage	mAP	IOU	comments
-	75.54	62.78	FT from darknet53.conv.75
M0	75.02	63.04	-
M1	73.69	63.75	-
M2	73.76	63.54	-
M3	71.26	61.61	full ternary

Table 2: result regard to yolov3

Iteration 100400(5000/class), steps x0.1 80% 90%, lr=0.001 at M0 and M1 Iteration 60400(3000/class), steps x0.1 80% 90%, lr=0.001 at M2 and M3

References

[1] Jiaolong Xu, Peng Wang, Heng Yang‡and Antonio et.al. "Training a Binary Weight Object Detector byKnowledge Transfer for Auonomous Driving" https://arxiv.org/pdf/1804.06332.pdf

[2] Fengfu Li and Bo Zhang, Bin Liu "Ternary weight networks" https://arxiv.org/pdf/1605.04711.pdf

[3] ohammad Rastegari, Vicente Ordonez, Joseph Redmon, Ali Farhadi Allen Institute for AI, University of Washington "XNOR-Net: ImageNet Classification Using BinaryConvolutional Neural Networks" https://arxiv.org/pdf/1603.05279.pdf

[4] https://github.com/pjreddie/darknet

[5] https://github.com/AlexeyAB/darknet