

# TOWARDS LIGHTWEIGHT CONVOLUTIONAL NEURAL NETWORKS FOR OBJECT DETECTION

**Dmitriy Anisimov, Tatiana Khanova**

Intel

Nizhny Novgorod, Russia

{dmitry.anisimov,tatiana.khanova}@intel.com

## ABSTRACT

We propose model with larger spatial size of feature maps and evaluate it on object detection task. With the goal to choose the best feature extraction network for our model we compare several popular lightweight networks. After that we conduct a set of experiments with channels reduction algorithms in order to accelerate execution. Our vehicle detection models are accurate, fast and therefore suit for embedded visual applications. With only 1.5 GFLOPs our best model gives 93.39 AP on validation subset of challenging DETRAC dataset. The smallest of our models is the first to achieve real-time inference speed on CPU with reasonable accuracy drop to 91.43 AP.

## 1 INTRODUCTION

Object detection in general and vehicle detection specifically has many applications including surveillance, autonomous driving, etc. Moreover, to be practical, object detector must operate on embedded processors that have far less compute capabilities than powerful GPUs used for benchmarking on typical computer vision datasets. Hence we focus on design of fast object detection model, which still retains high quality. We perform our experiments for vehicle detection sub-task on DETRAC dataset (Wen et al., 2015).

### OUR MOTIVATION

To reduce number of operations it is natural to try aggressively decreasing spatial size of feature maps. But from the other side, it is known that for tasks such as semantic segmentation or object detection it is important to preserve spatial information. So we investigate possibility to train accurate model with relatively large feature maps and still keep it fast.

Often network weights are initialized from a model, pretrained on classification of a large number of classes, e.g. ImageNet with 1000 classes (Russakovsky et al., 2014). We hypothesize that for a lot of practical tasks with few classes a considerable number of channels in convolution layers from such models are redundant. So in order to achieve high processing speed we focus on reducing the number of channels.

### OUR CONTRIBUTION

- Our findings show that it is possible to get good quality and fast detection model by increasing size of feature maps and decreasing number of channels.
- We demonstrate how to get lightweight model from a large one, by simply picking some number of channels and after that fine-tuning the resulting model on the given task.
- Straightforward layer-by-layer architecture like ResNet is shown to give good quality on par with more sophisticated architectures.
- We present the set of accurate models where the smallest one works in real-time on CPU at 34 fps.

**Table 1:** Feature extraction network accuracy on ImageNet (\* - our implementation)

Model	Top-1	Top-5	GFLOPs	MParams
SqueezeNet1.0	57.50	80.30	0.859	1.2
SqueezeNet1.0_bn	57.67	81.29	0.859	1.2
ResNet10	63.87	85.16	0.893	5.4
MobileNet*	69.29	89.25	0.579	4.3
PVANet	<b>72.34</b>	<b>91.16</b>	0.616	100.9

**Table 2:** Detection models with different feature extraction networks (on DETRAC validation split)

Model	AP	GFLOPs	MParams
MobileNet_SSD	86.92	1.7	5.2
PVANet_SSD	88.24	1.8	5.4
SqueezeNet1.0_SSD	88.61	1.8	3.5
MobileNet_light_SSD	88.93	0.745	2.3
ResNet10_SSD	<b>89.94</b>	1.8	4.0
SqueezeNet1.0_bn_SSD	<i>91.10</i>	1.8	3.5

## 2 RELATED WORK

Currently there are two popular approaches to object detection, namely: Faster R-CNN (Ren et al., 2015) and SSD (Liu et al., 2015). As it was shown in Huang et al. (2016) SSD framework has the best quality/speed trade-off, so we choose it for our experiments.

Besides detection framework it is important to use lightweight feature extraction network to preserve reasonable number of computations. For this reason we compared several such networks: ResNet10 (He et al., 2015), SqueezeNet (Iandola et al., 2016), MobileNet (Howard et al., 2017), PVANet (Kim et al., 2016).

Dilation (i.e. input stride in convolution), which we use in our model, first was proposed for segmentation task in Chen et al. (2016). They also sub-sampled two last fully connected layers in VGG network (Simonyan & Zisserman, 2014) in order to reduce computations. In SSD framework the same reduced VGG model was used. We extend this idea to sub-sampling all layers in feature extraction network.

Concurrent with our work, authors in Yu et al. (2017) applied dilations with similar purpose to classification and segmentation tasks. In comparison with their approach, after removing pooling operations, we additionally sample channels in convolution layers and thus preserve small number of computations.

In addition to good quality, real-time performance is required for most practical applications. There are a lot of methods dealing with high computation demand of neural networks. Roughly they can be divided into two groups, based on dependency on hardware support.

The first group includes such methods as quantization (Gysel et al., 2016; Zhou et al., 2017) and weight sparsifying (Han et al., 2015). But these approaches require delicate hardware customization to get practical speedup. For example, quantization relies on hardware support of low-bit operations and weights pruning leads to sparse computations.

Our target is to get fast model that does not depend on particular hardware, so we focus on the second hardware independent group. Those methods decrease number of computations using standard DNN building blocks. Different sorts of decompositions are used, e.g. searching for basis using PCA decomposition (Wen et al., 2017). Another possible solution is pruning of channels. While weight-level pruning requires sparse computation support, pruning also can be done on per-channel basis by eliminating less important channels in convolution filters (Li et al., 2016).

## 3 ARCHITECTURE

### 3.1 FEATURE EXTRACTION NETWORK

We choose four networks for our experiments. Their accuracies on ImageNet classification task are shown in Table 1. It was shown earlier, for example, for visual odometry (Agrawal et al., 2015), that features trained on two different tasks with the same network architecture lead to different quality on third task. The same may hold for fine-tuning from different architectures. So we can't choose feature extraction network for detection relying solely on classification accuracies. Below we demonstrate that the best accuracy on classification task doesn't necessarily lead to better accuracy for object detection.

---

Next we study influence of feature extraction network on detection accuracy. We modify chosen networks to have similar FLOPs and number of parameters for fair comparison. Brief description of the networks and our modifications is given below.

**SqueezeNet1.0** Network consists of "fire" blocks, each block contains two stages: first convolution layer makes "bottleneck" by reducing number of channels. Next there are two parallel inception-like (Szegedy et al., 2014) convolution layers with 1x1 and 3x3 kernels accordingly. We add two "fire" blocks at the end of network in the same way as it was done in SqueezeDet paper (Wu et al., 2016).

**MobileNet** In recently presented lightweight architecture authors replace convolution layer with depthwise convolution applying a single 3x3 filter to each input channel, followed by conventional convolution with 1x1 kernels. This network already contains required number of FLOPs and parameters, so we keep it as is.

**PVANet** PVANet is a relatively sophisticated architecture. It consists of CReLU layers (Shang et al., 2016) at the beginning of the network, followed by inception blocks. At the end there are two large fully connected classification layers with 4096 channels each, that's why it has a large amount of weights. To reduce number of weights we sample last fully connected layers in the same way as it was done for VGG model in DeepLab and SSD frameworks:<sup>1</sup>

- 6x6 kernels are sampled to 3x3 and dilation 6 is added to the following layer in order to preserve receptive field (to compensate kernel sampling and removing previous max pooling).
- 4096 channels are sampled to 256. We use 256 instead of 1024 to be able to compare with other models.

**ResNet10** Often only large versions of ResNet architecture are used, e.g. with 50 layers and more, but we show that small 10-layer model gives good quality as well. We use model with pre-activations trained in Caffe (Jia et al., 2014) by Computer Vision Group Jena (Simon et al., 2016). We sample last two layers to 256 channels for comparison with other models.

### 3.2 SSD MODIFICATIONS FOR DETRAC

Since ground truth bounding boxes for DETRAC test dataset are not publicly available, we prepare our train/validation split. We select 10 videos (totally 15K frames) as validation and the rest 68K as train trying to preserve ratio of night/day videos and keep similar locations in the same split.

To adapt SSD framework to DETRAC dataset we make the following changes:

- We keep three prior box scale levels based on the bounding box sizes on the dataset. We use the same parameters, except for the first "small" scale, for which we decrease range to 15/50.
- Our experiments show that down-sampling images to 320x256 resolution gives good quality/speed trade-off.

Results of this configuration with four feature extraction networks are shown in Table 2. One can see that ResNet10 provides the best accuracy, so we choose it for further experiments.

In the third version of this paper we fix issue with proper handling ignore regions during evaluation on DETRAC validation split. All conclusions remain the same, the only change is that metrics become around 2 AP lower.

In Section 3.6 we get the best result with MobileNet feature extraction network, so we conduct additional experiment with it. We sample channels to 256 for all layers with large number of channels. This model shows better accuracy comparing to original MobileNet and has smaller number of parameters, see Table 2.

---

<sup>1</sup>We reduce learning rate for detection model training two times, because it diverges with original value.

**Table 3:** Comparing original ResNet10 feature extraction network with our detection model

name	ResNet10			SSDR			_1.5	_0.75	_0.47
	type	spatial dim.	channels	type	spatial dim.	channels			
conv1	7x7/2	128x160	64	7x7/2	128x160	64	49	41	
pool1	3x3	64x80	64	3x3	64x80	64	49	41	
res_block	3x3	64x80	64	3x3	64x80	64	49	41	
res_block	3x3/2	32x40	128	3x3/2	32x40	64	49	41	
res_block	3x3/2	16x20	256	3x3 dil 1,2	32x40	128	76	49	
res_block	3x3/2	8x10	512	3x3 dil 2,4	32x40	128	76	49	
conv16_det				3x3/2	16x20	128	76	49	
conv32_det				3x3/2	8x10	128	76	49	

After submission to the conference we did the same with SqueezeNet1.0\_bn network (SqueezeNet1.0 with batch normalizations). Since it achieves the best quality we expect that our approach applied to it would lead to even better results.

### 3.3 MODEL

In this section we study possibility of getting accurate and fast detection model by increasing size of feature maps and decreasing number of channels, we propose the following model. To keep size of feature maps we remove two last spatial reductions, that reduce image resolution by 16 and 32 times. In order to preserve receptive field of following convolution layers we add dilations 2 and 4 to them accordingly. We also sample channels in most convolution layers to retain reasonable number of FLOPs.<sup>2</sup> Resulting architecture is shown in Table 3, model **SSDR\_1.5**.

### 3.4 STUDYING EVEN LIGHTER MODELS

In order to accelerate existing model further and achieve real-time performance on CPU, we continue experiments with methods for eliminating channels, see Table 3, models **SSDR\_0.75** and **SSDR\_0.47**.

**One-shot random sampling** First, we take our best **SSDR\_1.5** model, decrease number of channels following random sampling strategy and fine-tune it. Resulting 473 MFLOPs model gives 91.09 AP on DETRAC validation split.

**One-shot pruning** Pruning scheme, proposed in Li et al. (2016) appears to be straightforward improvement of the random sampling approach:

1. For each convolution layer in network, except for the layers in detection part:
  - (a) For each filter calculate  $L_1$  metric to measure usefulness.
  - (b) Prune specified percentage of filters with the smallest metrics. Percentage of filters to prune is chosen empirically to be 5% and 10% for the first and the last half respectively.
2. Fine-tune the whole network on the training dataset with smaller learning rate.

**Iterative pruning** The above algorithm can be repeated until desired quality/speed trade-off is achieved leading to iterative pruning as an opposite to one-shot approach.

Contrary to Li et al. (2016), for pruning of pre-elementwise sum convolutions we find that using 3x3 filters to calculate metric instead of shortcut convolutions gives better result.

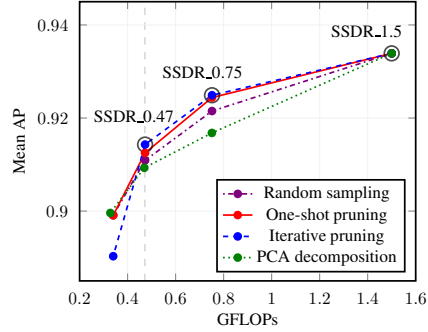
**PCA decomposition** Another approach we employ was proposed in Wen et al. (2017) and consists of applying PCA to filters in order to determine basis for each convolution and then decompose layer into two consecutive convolutions.

<sup>2</sup>We experiment with sampling channels by  $L_1$  metric (see Section 3.4), but it doesn't improve results.

**Table 4:** Comparison of detection models (on DETRAC validation split)

Model	AP	GFLOPs	MParams
ResNet10_Clustered	89.07	1.7	3.9
ResNet10_SSD	89.94	1.8	4.0
ResNet10_FPN	91.50	1.4	2.8
SSDR_1.5 (ours)	<b>93.39</b>	1.5	1.1
SSDR_0.75 (ours)	92.49	0.75	0.47
SSDR_0.47 (ours)	91.43	<b>0.47</b>	0.24

**Figure 1:** Comparison of channel reduction techniques on DETRAC validation split. Real-time performance on CPU is labeled with vertical line.



**Table 5:** Comparison of our model with the best models from leader-board (on DETRAC test split)

Model	full set	easy set	medium set	hard set	cloudy set	night set	rainy set	sunny set
RTN	74.15%	91.52%	79.16%	61.73%	77.02%	77.20%	65.27%	84.14%
EB	67.96%	89.65%	73.12%	53.64%	72.42%	73.93%	53.40%	83.73%
NANO	63.01%	80.33%	68.04%	50.73%	67.00%	62.20%	55.89%	73.89%
SSDR_0.75 (ours)	59.07%	77.84%	64.41%	45.98%	62.79%	60.88%	48.55%	74.32%
SSDR_1.5 (ours)	58.68%	79.55%	63.74%	44.93%	61.59%	62.19%	47.47%	74.42%
FasterRCNN2	58.45%	82.75%	63.05%	44.25%	66.29%	69.85%	45.16%	62.34%
YOLO2	57.72%	83.28%	62.25%	42.44%	57.97%	64.53%	47.84%	69.75%
SSDR_0.47 (ours)	57.07%	76.67%	62.22%	43.89%	62.41%	58.48%	45.26%	72.55%

Comparison of aforementioned techniques applied to ours best model is shown in Figure 1. Iterative pruning strategy gives the best 91.43 AP for 473 MFLOPs model and 92.49 AP for 752 MFLOPs model. But the results we get with other methods are close to the best one.

### 3.5 EXPERIMENTS

We compare our models with several recently proposed architectures, namely: base SSD with three scale levels (Liu et al., 2015), SSD with clustered priors (Erhan et al., 2013; Wu et al., 2016) and Feature Pyramid Networks (Lin et al., 2016). Our solution gives superior result comparing to all aforementioned models either on accuracy or processing speed, see Table 4.

**Clustered priors** Following Erhan et al. (2013); Wu et al. (2016) we cluster DETRAC ground truth bounding boxes to get dataset specific priors. First, all bounding boxes were clustered into 3 scale groups, and then each of them was clustered into 4 groups to achieve comparable number of priors with the original SSD. While model with clustered priors does not show the best result (Table 4), clustering approach may provide valuable insight on choice of scale/aspect ratio parameters for priors generated by other methods.

### ACCURACY AND TIME

Table 5 compares accuracies of our models with other results on different subsets of DETRAC test set. On time of submission our model is ranked 4th in official leader-board<sup>3</sup> by quality and it is at least an order of magnitude faster than all others. Noticeably, on the test set smaller **SSDR\_0.75** model gives even better accuracy than the base one.

We measure inference speed of our smallest **SSDR\_0.47** model on Intel®Core™i7-6700K CPU @ 4.00GHz x 8 using Intel®MKL library and Caffe and get ~34 fps.<sup>4</sup>

<sup>3</sup><http://detrac-db.rit.albany.edu/DetRet>

<sup>4</sup>Note that we merged batch normalization mean/variance in previous convolution weights to eliminate redundant computations.

**Table 6:** Detection models with different feature extraction networks (on VOC 2007 test)

Model	AP	GFLOPs	MParams
SqueezeNet1.0 SSD	38.45	2.8	7.0
ResNet10 SSD	64.83	2.3	6.7
SqueezeNet1.0.bn SSD	65.61	2.8	7.0
PVANet SSD	67.69	2.3	8.1
MobileNet SSD	70.04	2.6	8.8
SSDM_7.5 (ours)	<b>73.08</b>	7.5	10.1

**Table 7:** Comparison of ResNet10 based detection models (on VOC 2007 test)

Model	AP	GFLOPs	MParams
ResNet10.FPN	63.76	2.0	5.3
ResNet10 SSD	64.83	2.3	6.7
SSDR_5.5 (ours)	<b>68.73</b>	5.5	8.5

### 3.6 PASCAL VOC

In the previous sections we apply our approach to concrete vehicle detection task. In order to extensively study capabilities of proposed techniques we conduct similar set of experiments on general object detection PASCAL VOC dataset (Everingham et al., 2015). We don’t experiment with channel reduction for them and use the original SSD 300x300 configuration. The results obtained from this study are provided in Tables 6 and 7. Since we get the best result for SSD with Mobilenet feature extraction network, we make use of our modification with replacing two max pooling with dilations to it as well, see Table 6, SSDM\_7.5 model.

Interestingly, performance of detection models with different feature extraction networks on VOC test set shares tendency in accuracies with Imagenet classification task (Table 1) contrary to what we get for DETRAC dataset. We argue that this is because of the difference in the number of classes between these datasets: only one class in DETRAC and 20/1000 in PASCAL VOC and Imagenet accordingly. Original SqueezeNet1.0 SSD model shows poor result on this dataset, but works quite well after we add batch normalizations to it, see Table 6, SqueezeNet1.0.bn SSD model.

## 4 CONCLUSION

We have presented a set of detection models, which are accurate, fast and therefore suit for real-world applications. We keep high spatial resolution in feature extraction part, rather than losing spatial structure information caused by progressively reducing the resolution of internal representations. Even though the proposed network is designed for vehicle detection, we believe our design choices can be applicable to other practical tasks, particularly where detailed understanding of the scene is important.

Our network design is independent of network compression and quantization, so those methods are applicable to our network as well to further increase the actual performance in real tasks. We have shown that relatively simple pruning technique can give real-time performance on CPU with less than 2% AP loss on test set.

## REFERENCES

- Pulkit Agrawal, João Carreira, and Jitendra Malik. Learning to see by moving. *CoRR*, abs/1505.01596, 2015. URL <http://arxiv.org/abs/1505.01596>.
- Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016. URL <http://arxiv.org/abs/1606.00915>.
- Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. *CoRR*, abs/1312.2249, 2013. URL <http://arxiv.org/abs/1312.2249>.
- M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, January 2015.

- 
- Philipp Gysel, Mohammad Motamedi, and Soheil Ghiasi. Hardware-oriented approximation of convolutional neural networks. *CoRR*, abs/1604.03168, 2016. URL <http://arxiv.org/abs/1604.03168>.
- Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. *CoRR*, abs/1506.02626, 2015. URL <http://arxiv.org/abs/1506.02626>.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL <http://arxiv.org/abs/1512.03385>.
- Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017. URL <http://arxiv.org/abs/1704.04861>.
- Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016. URL <http://arxiv.org/abs/1611.10012>.
- Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016. URL <http://arxiv.org/abs/1602.07360>.
- Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093, 2014. URL <http://arxiv.org/abs/1408.5093>.
- Kye-Hyeon Kim, Yeongjae Cheon, Sanghoon Hong, Byung-Seok Roh, and Minje Park. PVANET: deep but lightweight neural networks for real-time object detection. *CoRR*, abs/1608.08021, 2016. URL <http://arxiv.org/abs/1608.08021>.
- Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *CoRR*, abs/1608.08710, 2016. URL <http://arxiv.org/abs/1608.08710>.
- Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. *CoRR*, abs/1612.03144, 2016. URL <http://arxiv.org/abs/1612.03144>.
- Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015. URL <http://arxiv.org/abs/1512.02325>.
- Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015. URL <http://arxiv.org/abs/1506.01497>.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. URL <http://arxiv.org/abs/1409.0575>.
- Wenling Shang, Kihyuk Sohn, Diogo Almeida, and Honglak Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. *CoRR*, abs/1603.05201, 2016. URL <http://arxiv.org/abs/1603.05201>.
- Marcel Simon, Erik Rodner, and Joachim Denzler. Imagenet pre-trained models with batch normalization. *CoRR*, abs/1612.01452, 2016. URL <http://arxiv.org/abs/1612.01452>.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL <http://arxiv.org/abs/1409.1556>.

- 
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL <http://arxiv.org/abs/1409.4842>.
- Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. DETRAC: A new benchmark and protocol for multi-object tracking. *CoRR*, abs/1511.04136, 2015. URL <http://arxiv.org/abs/1511.04136>.
- Wei Wen, Cong Xu, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Coordinating filters for faster deep neural networks. *CoRR*, abs/1703.09746, 2017. URL <http://arxiv.org/abs/1703.09746>.
- Bichen Wu, Forrest N. Iandola, Peter H. Jin, and Kurt Keutzer. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. *CoRR*, abs/1612.01051, 2016. URL <http://arxiv.org/abs/1612.01051>.
- Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. *ArXiv:1705.09914*, 2017. URL <https://arxiv.org/abs/1705.09914>.
- Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *CoRR*, abs/1702.03044, 2017. URL <http://arxiv.org/abs/1702.03044>.