Quantization Mimic: Towards Very Tiny CNN for Object Detection

Yi Wei[1(✉)], Xinyu Pan[2], Hongwei Qin[3], Wanli Ouyang[4], and Junjie Yan[3]

[1] Tsinghua University, Beijing, China
wei-y15@mails.tsinghua.edu.cn
[2] The Chinese University of Hong Kong, Hong Kong, China
THUSEpxy@gmail.com
[3] SenseTime, Beijing, China
{qinhongwei,yanjunjie}@sensetime.com
[4] The University of Sydney, SenseTime Computer Vision Research Group, Sydney, New South Wales, Australia
wanli.ouyang@sydney.edu.au

Abstract. In this paper, we propose a simple and general framework for training very tiny CNNs (*e.g.* VGG with the number of channels reduced to $\frac{1}{32}$) for object detection. Due to limited representation ability, it is challenging to train very tiny networks for complicated tasks like detection. To the best of our knowledge, our method, called Quantization Mimic, is the first one focusing on very tiny networks. We utilize two types of acceleration methods: mimic and quantization. Mimic improves the performance of a student network by transfering knowledge from a teacher network. Quantization converts a full-precision network to a quantized one without large degradation of performance. If the teacher network is quantized, the search scope of the student network will be smaller. Using this feature of the quantization, we propose Quantization Mimic. It first quantizes the large network, then mimic a quantized small network. The quantization operation can help student network to better match the feature maps from teacher network. To evaluate our approach, we carry out experiments on various popular CNNs including VGG and Resnet, as well as different detection frameworks including Faster R-CNN and R-FCN. Experiments on Pascal VOC and WIDER FACE verify that our Quantization Mimic algorithm can be applied on various settings and outperforms state-of-the-art model acceleration methods given limited computing resouces.

Keywords: Model acceleration · Model compression
Quantization · Mimic · Object detection

1 Introduction

In recent years, CNN achieved great success on various computer vision tasks. However, due to their huge model size and computation complexity, many CNN

Y. Wei and X. Pan—The work was done during an internship at SenseTime

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11212, pp. 274–290, 2018.
https://doi.org/10.1007/978-3-030-01237-3_17

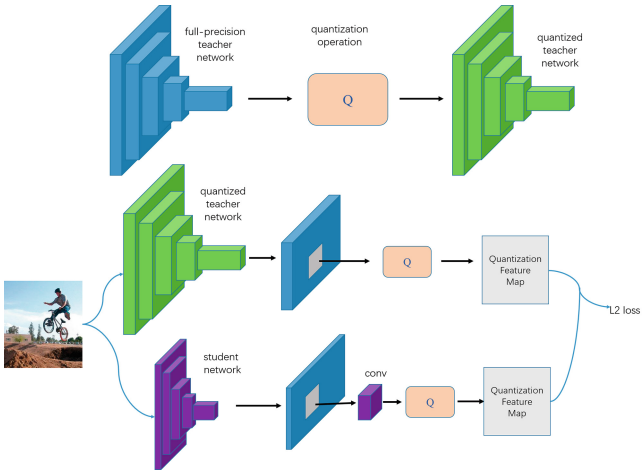

Fig. 1. The pipeline of our method. First we train a full-precision teacher network. Then we operate quantization on the feature map of full-precision teacher network and we get a quantized network. Finally we use this quantized network as teacher model to teach a quantized student network. We emphasize that we do both quantization operation on feature maps of student and teacher networks in training stages.

models cannot be applied on real world devices directly. Many previous works focus on how to accelerate CNNs. They can be roughly divided to four categories: quantization (*e.g.* BinaryNet [1]), group convolution based method (*e.g.* MobileNet [2]), pruning (*e.g.* channel pruning [3]) and mimic (*e.g.* Li *et al.* [4]).

Although most of these works can accelerate models without degradation of performance, their speed-up ratios are limited (*e.g.* compress VGG to VGG-1-4[1]). Few methods are experimented on very tiny models (*e.g.* compress VGG to VGG-1-16). "Very tiny" is a relative concept and we define it as a model whose channel numbers of every layer is less than or equal to $\frac{1}{16}$ compared with original model. Our experiments show that our method outperform other approaches for very tiny models.

As two kinds of model acceleration methods, quantization and mimic are widely used to compress model. Quantization methods can transfer a full-precision model to a quantized model[2] while maintaining similar accuracy. However, using quantization method to directly speed up models usually need extra specific implementation (*e.g.* FPGA) and specific instruction set. Mimic methods can be used on different frameworks and easy to implement. The essence of these methods is knowledge transfer in which student networks learn the high-level representations from teacher networks. However, when applied on very tiny

[1] In this paper *-1-n* network means a network whose channel numbers of every layer is reduced to $\frac{1}{n}$ compared with original network.

[2] The quantized network in this paper means a network whose output feature map is quantized but not means parameter is quantized

networks, mimic method does not work well either. This is also caused by the very limited representation capacity.

It is a natural hypothesis that if we use quantization method to discretize the feature map of the teacher model, the search scope of the student network will get shrinked and it will be easier to transfer knowledge. And quantization on student network can increase the matching ratio on the discrete feature map from teacher network. In this paper, we propose a new approach utilizing the advantages of quantization and mimic methods to train very tiny networks. Figure 1 illustrates the pipeline. Quantization operation is applied to the feature map of the teacher model and the student model. The quantized feature map of the teacher model is used as supervision of the student model. We propose that this quantization operation can facilitate feature map matching between two networks and make knowledge transfer easier.

To summarize, the contributions of this paper are as follows:

- We propose an effective algorithm to train very tiny networks. To the best of our knowledge, this is the first work focusing on very tiny networks.
- We utilize quantized feature maps to facilitate knowledge distilling, *i.e.* quantization and mimic.
- We use a complicated task object detection instead of image classification to verify our method. Sufficient experiments on various CNNs, frameworks and datasets validate our approach effective.
- The method is easy to implement and has no special limitation during training and inference.

2 Related Work

2.1 Object Detections

The target of object detection [5–10] is to locate and classify the objects in images. Before the success of convolutional neural network, some traditional pattern recognition algorithms (HOG [11], DPM [12] *et al.*) are used on this task. Recently, R-CNN [13] and its variants become the popular method for object detection task. The SPP-Net [14] and Fast R-CNN [15] reuse feature maps to speed up R-CNN framework. Beyond the pipeline of Fast R-CNN, Faster R-CNN add region proposal networks and use joint-train method during training. R-FCN utilize position-sensitive score maps to reduce more computation. YOLO [16] and SSD [17] are the typical algorithms of region-free methods. Although the frameworks used in this paper are from region proposal solutions family, Quantization Mimic can easily transform to YOLO and SSD methods.

2.2 Model Compression and Acceleration

Group Convolution Based Methods: The main point of this kind of methods is to use group convolution for acceleration. Mobilenet [2] and Googlenet Xception [18] utilize Depthwise Convolution to extract features and Pointwise

Convolution to merge features. Beyond these works, Zhang *et al.* [19] propose a general group convolution algorithm and show that Xception is the special case of their method. Group operation will block the information flow between different group convolutions and most recently, Shufflenet [20] introduces channel shuffle approach to solve this problem.

Quantization: Quantization methods [21,22] can reduce the size of models efficiently and speed up for special implementation. BinaryConnect [23], binarized neural network (BNN) [1] and LBCNN [24] replace floating convolutional filter with binary filter. Furthermore, INQ [25] introduce a training method to quantize model whose weights are constrained to be either powers of two or zero without a decrease on performance. Despite these advantages, quantization models can only be used to speed up on special devices.

Pruning and Sparse Connection: [26,27] set sparse constraint during training for pruning. [28,29] focus on the importance of different filter weights and do pruning operation according to weights' importance. And these methods are training-based, which are more costly. Recently He *et al.* [3] propose an inference-time pruning method, using LASSO regression and least square construction to select channels in classification and detection task. Furthermore, Molchanov *et al.* [30] combine transfer learning and greedy criteria-based pruning. We use He *et al.* [3] and Molchanov *et al.* [30] for comparing our alogrithm and we will show that it is difficult for them to prune a large network (such as VGG) to a very tiny network (such as VGG-1-32). Sparse connection [31–34] can be considered as parameter-wise pruning method, eliminating connection between neurons.

Mimic: The principle of mimic is Knowledge Transfer. As a pioneering work, Knowledge Distillation (KD) [35] defines soft targets as outputs of the teacher network. Compared with labels, soft targets provide extra information about inter-class similarities. FitNet [36] develops Knowledge Transfer as whole feature map mimic learning to compress wide and shallow networks to thin and deep networks. Li *et al.* [4] extend mimic techniques for object detection task. We use their joint-train version as our baseline.

3 Our Approach

In this section, we first introduce the quantization method and mimic method we use separately, then combine them and propose the pipeline of Quanzition Mimic algorithm. In Sect. 3.4 we show the theoretical analysis of our approach.

3.1 Quantization

[21–23] use quantization method to compress models directly. Unlike them, we use quantization to limit the range and help mimic learning. In details, the

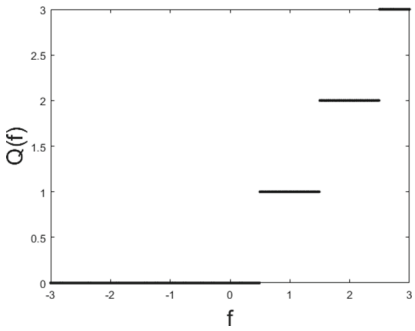

Fig. 2. Quantized ReLU function. The new activation function is defined as $\widetilde{f} = Q(f)$, where f is the original activation function.

quantization for teacher network is to discretize its output and in the meanwhile we can guarantee the accuracy of teacher network when doing quantization. And quantizing the output of student network can help it match the discrete output of teacher network, which is the goal of mimic learning. In our work, we do quantization operation on the last activation layer of the teacher network.

INQ [25] constrains the output to be either zero or power of two. Different from them, we use uniform quantization for the following reason. R-FCN [37] and Faster R-CNN [38] use RoI pooling operation which is a kind of max pooling operation. The output of RoI pooling layer is determined by the max response of every block in RoIs. So it is important to describe strong response of feature maps more accurately. Uniform quantization can better describe large value than power of two quantization. We define the element-wise quantization function Q as:

$$Q(f) = \beta \quad \text{if } \frac{\alpha+\beta}{2} < f \leq \frac{\gamma+\beta}{2} \tag{1}$$

where α,β and γ are the adjacent entries in the code dictionary D:

$$D = \{0, s, 2s, 3s..\} \tag{2}$$

where s is the stride of uniform quantization. We use function Q to convert full-precision feature maps to quantized feature maps:

$$\widetilde{f} = Q(f) \tag{3}$$

where f is the feature map. Figure 2 illustrates quantized ReLU function. As for backward propagation, inspired by BNN [1], we use the full-precision gradient. We find that quantized gradient will cause the student network difficult to converge.

3.2 Mimic

In popular CNN detectors, the feature map from feature extractors (*e.g.* VGG, Resnet) will affect both localization and classification accuracy. We use L2 regression to let student networks learn the feature map from the teacher networks

and utilize Li *et al.* [4] joint-train version as our backbone. Unlike soft target [35] whose dimension is equal to the number of categories, the dimension of feature maps is related to the size of inputs and networks architecture. Sometimes number can be millions. Simply mimicking the whole feature maps is difficult for student network to converge. Faster R-CNN [38] and R-FCN [37] are region-based detectors and both of them use RoI-Pooling operation. So the region of interest plays more important role than other regions. We use mimic learning between the region of interest on student's and teacher's feature maps. The whole loss function of mimic learning is described as follows.

$$L = L^r_{cls} + L^r_{reg} + L^d_{cls} + L^d_{reg} + \lambda L_m \tag{4}$$

$$L_m = \frac{1}{2N} \sum_i \left\| f^i_t - r\left(f^i_s\right) \right\|^2_2 \tag{5}$$

where L^r_{cls}, L^r_{reg} are the loss function of region proposal networks [15] while L^d_{cls}, L^d_{reg} are the function of R-FCN or Faster R-CNN detectors. We define L_m as the mimic-loss and λ is the loss weight. N is the number of region proposals. f^i_t and f^i_s represent the ith region proposal on teacher and student network's feature maps. Function r transfers the feature map from student network to the same size of teacher network. The mimic learning is on the last year of feature extractor networks.

Though RoI mimic learning reduces the dimension of feature maps and helps student network convergence, very tiny network is sensitive to mimic loss weight λ. If λ is small, it will weaken the effectiveness of mimic learning. In the contrast, large λ will also bring bad results. Due to the poor learning capacity of very tiny network, large λ will cause it focus on the learning of teacher network's feature map at the begining of training. In this way, it will ignore other loss. We name this phenomenon as 'gradient focus' and we set λ as 0.1, 1 and 10 for experiments.

3.3 Quantization Mimic

The pipeline of our algorithm is as follows: First we train a full-precision teacher network. Then we use function Q to compress full-precision teacher network to a quantized network. To get high performance compressed model, we finetune on full-precision network. Finally, we utilize quantized teacher network to teach student network using mimic loss as supervision. And during training, we both quantize the feature map of teacher and student network. Figure 3 illustrates our method.

Because of quantization operation, the mimic loss L_m is redefined as:

$$L_m = \frac{1}{2N} \sum_i \left\| Q\left(f^i_t\right) - Q\left(r\left(f^i_s\right)\right) \right\|^2_2 \tag{6}$$

where quantization function Q is defined in Eq. 1

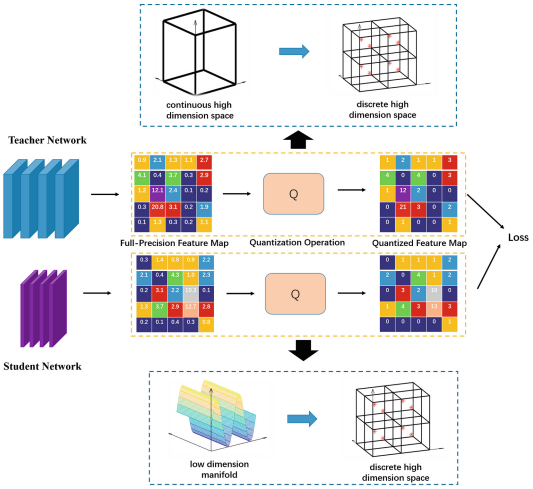

Fig. 3. The effect of quantizatuon operation. We use quantized teacher network to guide quantized student network. The quantization on teacher network can discretize its feature maps and convert a continous high dimension space to a discrete high dimension space. And for student network, quantization helps low dimension manifold to match a discrete high dimension feature map. In this way, mimic learning becomes easier.

3.4 Analysis

We will show that the quantization of both teacher and student networks will facilitate feature maps matching between student and teacher networks and help student network learn better. Figure 3 shows the effect of quantization operation. We assume that f_t^n is the feature map of full-precision teacher network with the input I_n. The width, height and channel numbers of f_t^n are W_t^n, H_t^n and C_t^n.We squeeze f_t^n as a column vector y_n whose dimension is $W_t^n H_t^n C_t^n$. The target of mimic learning is to get approximate solution of the following equation:

$$Y = w_s I \tag{7}$$

$$Y = [y_1, y_2, ..., y_n] \tag{8}$$

$$I = [I_1, I_2, ..., I_n] \tag{9}$$

where w_s is the weights of student network. However, due to the high dimensionality of y_n and large image numbers, the rank of Y can be very high. On the other hand, very tiny networks have few parameters and the rank of w_s is low. Therefore, it is difficult for very tiny student networks to mimic high dimension feature maps directly. The target of Quantization Mimic is changed as:

$$Q\left(Y\right) = Q\left(w_s I\right) \tag{10}$$

where Q is quantization function. The quantization operation on the output of teacher network discretizes its feature maps. Furthermore, because of the range

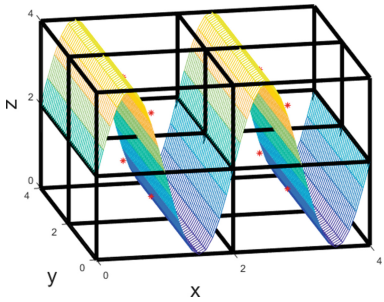

Fig. 4. A manifold in 3-dimension space. The manifold intersect all 8 cubes. The point '*' represent the center of cube, which is the vector after quantization operation.

of element in feature maps is bounded, the value of every entry in matrix $Q(Y)$ is discrete and finite. For example, if the range of element in f_t^n is $[0, 40]$ and the stride of uniform quantization is 8, the possible value of entry in $Q(Y)$ is from $\{0, 8, 16, 24, 32, 40\}$. In this way, we convert continuous high dimension space to discrete high dimension space.

The quantization on student networks makes it easier to match the $Q(f_t^n)$. Every axis of target space for student network can be separated by entries in code dictionary. And the whole space is separated by several high dimension cubes.

For simplicity, we assume the dimension of target space ϕ is 3, *i.e.* , the dimension of y_n is 3. The code dictionary is selected as $\{1, 3\}$. Because of quantization operation, this 3-dimension space is separated by 8 cubes (See Fig. 4). If a vector v is in cube c , after quantization operation, it will be the center of cube c. For example, $v = [1.2, 2.2, 1.8]^{\mathrm{T}}$, $Q(v) = [1, 3, 1]^{\mathrm{T}}$, and $[1, 3, 1]^{\mathrm{T}}$ is the center of a cube.

We suppose that feature maps of student network consist a low dimension manifold. The goal of mimic learning is to use this manifold to fit all 8 cube centers, *i.e.* , we want these 8 centers on the manifold. However, after introducing quantization on student network, if the manifold intersect a cube, the manifold can achieve the center of this cube. Thus, instead of matching all centers, we only need the manifold to intersect 8 cubes, which weaken matching conditions. And in this way, there are more suitable manifolds , which promotes feature maps matching between two networks. Experiments in Sect. 4.1 shows that our approach is still effective in high dimension case. Figure 4 illustrates a manifold in 3-dimension space which intersect all cubes.

3.5 Implementation Details

We train networks with Caffe [39] using C++ on 8 Nvidia GPU Titan X Pascal. We use stochastic gradient descent (SGD) algorithm. The weight decay is 0.0005 and momentum is 0.9. We set uniform quantization stride as 1 for all experiments.

VGG with R-FCN: In this experiment we rescale the images such that their shorter side is 600 and we use original images for test. We use gray images as input. The learning rate is 0.001 for the first 50K iterations and 0.0001 for the next 30K iterations. The teacher network is VGG-1-4 with R-FCN and we set mimic loss weight λ as 1. For RPN anchors, we use one aspect ratio and 4 scales with box areas of 4^2, 8^2, 16^2, 32^2. 2000 RoIs are used to sample the features on the feature maps of teacher and student networks. The ROI output size of R-FCN detector is set as 3×3. We utilize OHEM [40] algorithm to help training.

Resnet with Faster R-CNN: We rescale all the images such that shorter side is 600 for both training and test. We totally train 40K iterations. The learning rate is 0.001 for the first 30K iterations and 0.001 for the last 10k iterations. We set λ as 0.1, 1 and 10 for Resnet experiment respectively. And for RPN anchors, we use 2 aspect ratios (2:1, 3:1) and 3 scales with box areas of 4^2, 8^2 and 16^2. 128 RoIs are used to sample the features on the feature maps of teacher and student networks. The ROI output size of Faster R-CNN detector is set as 7×7.

4 Experiments

To prove the generalization ability of our method, we evaluate our approach for different frameworks on different datasets. In detail, we use VGG with R-FCN and Resnet with Faster R-CNN as our backbones. Results are reported on WIDER FACE [41] and Pascal VOC [42].

4.1 Experiments on WIDER FACE Dataset

WIDER FACE dataset [41] contains about 32K images with 394K annotated faces. The size of faces in WIDER FACE dataset vary a lot. The validation and Test set are divided into *easy* , *medium* and *hard* subsets. We find that VGG and VGG-1-4 have similar performance on WIDER FACE dataset (See Table 3) and we use VGG-1-4 with R-FCN detector as our teacher network (large model). To show the superiority of our algorithm, VGG-1-32 with R-FCN detector is selected as the student network (small model). Table 1 illustrate the speed and size of our very tiny model student network compared with large models. It has extremely small size and fast speed.

Main Results. We implement our algorithm on VGG-1-32 with R-FCN detector. We compare our method with Li *et al.* [4], He *et al.* [3] and group convolution based accelerating method including using Depthwise Convolution and Group Convolution. The results are shown in Table 2 (we set input as 1000×600 and compute the complexity). For fair comparison, we use the same implementation details for all experiments. We involve Depthwise Convolution and Group Convolution into VGG-1-32 structures, guaranteeing the similar complexity with the original network. For example, we extend the channel numbers of every convolution layers c to $\lceil \sqrt{3}c \rceil$ and we set group number as 3. We also compare with

Table 1. The comparision between VGG, VGG-1-4, VGG-1-32 with R-FCN detector on speed and size. The size is calculated theoretically. VGG-1-32 with R-FCN has tiny size and amazing speed, which can be applied on embedded devices. Tested on Titan X GPU with a single image of which the longer side is resized to 1024.

Method	Speed	Size
VGG with R-FCN	103.6 ms	79.8M
VGG-1-4 with R-FCN	30.2 ms	5.04 M
VGG-1-32 with R-FCN	**9.6 ms**	**0.13 2M**

Table 2. Comparison with other methods. The results show that our method outperforms others (higher is better). Group convolution based approach(Depthwise Convolution and Group Convolution) don't work well on the very tiny model. Quantization Mimic also outperforms than Li *et al.*, who only uses mimic learning.

Solution	Complexity (MFLOPS)	Easy	Medium	Hard
Scratch	227	71.3	55.4	23.8
Depthwise Convolution	232	69.1	51.1	21.6
Group Convolution(group 2)	286	67.8	51.9	22.4
Group Convolution(group 3)	273	65.8	50.8	22.1
He *et al.* [3]	227	68.0	50.7	22.1
Molchanov *et al.* [30]	227	73.2	58.2	25.2
Li *et al.* [4](only mimic)	227	71.9	58.2	25.6
Quantization Mimic	227	**73.9**	**62.1**	**27.6**

pruning methods [3,30]. The pruning ratio is set as 8, which means the model we get after pruning has the same size with VGG-1-32. The results demonstrate that our algorithm outperforms other methods. We find that group convolution based methods are not suitable for very tiny networks. This is mainly because very tiny networks usually have small channel numbers and using group convolutions will block the information flow. Compared with pruning methods [3,30], Quantization Mimic also works better. We argue that pruning methods can get good results on large models (*e.g.* , VGG and Resnet). However, none of these works try to prune a network to $\frac{1}{16}$ times. Compared with mimic method [4], Quantization Mimic outperforms it by 2.0 points, 3.9 points and 1.9 points on *easy*, *medium* and *hard* subsets. We find that quantized teacher network has better performance than full-precision teacher network. Ablation experiments are conducted to diagnose how Quantization Mimic brings improvement.

Table 3 further shows the effectiveness of our approach. We can see that our method can increase AP of very tiny models by 2.6 points, 6.7 points and 3.7 points on *easy*, *medium* and *hard* subsets respectively. Results on *medium* and *hard* subsets, the small model can even achieve comparable results with large model.

Table 3. The comparision between VGG and VGG-1-4 on the WIDER FACE dataset. We suggest that VGG has abundant structures and it has similar performance with VGG-1-4. And we choose VGG-1-4 as teacher model.

Model	Solution	Easy	Medium	Hard
VGG	Full-precision	83.9	61.0	26.8
VGG-1-4	Full-precision	82.4	62.5	26.3
	Quantized	83.7	65.0	27.4
VGG-1-32	Scratch	71.3	55.4	23.8
	Quantization mimic	73.9	62.1	27.6

Ablation Study on Quantization Operation. To verify the effectiveness of quantization operation, we do several experiments. As demonstrated in Table 4, the performance of teacher network directly impact the performance of student network. Also, the quantization operation help mimic learning and improves the performance of student network. For the same quantized teacher network, doing quantization operation on the student network increase AP by 0.9 point, 2.8 points and 2.0 points on three subsets.

Table 4. Quantization *vs.* Nonquantization: The ablation study shows that the performance of student network depends on the performance of teacher network. The results also suggest that quantization method do help mimic learning.

Teacher quantization?	Student quantization?	Easy	Medium	Hard
		71.9	58.2	25.6
✓		73.0	59.3	25.6
✓	✓	**73.9**	**62.1**	**27.6**

We notice that quantization operation has regularization effect on network. To exclude that it is the regularization that bring improvement of performance, we also do experiments with and without quantization on student network. In Table 5, we find that only doing quantization has no influence on the performance, *i.e.* , the improvement comes from Quantization Mimic.

Table 5. The influence of quantization only on small networks. The results suggest quantization only does not bring improvement.

Model	Quantization?	Easy	Medium	Hard
VGG-1-32	✓	71.9	55.2	23.7
		71.3	55.4	23.8

To further show that quantization operation can help student networks learn better, we illustrate the matching ratio of each RoI. In Sect. 3.4 we show that quantization operation promotes feature map matching between two networks. And in Sect. 3.2, we introduce that our mimic learning is based on RoIs. Thus, we consider the matching ratio of each RoI, *i.e.* , the percantage of elements in a RoI whose distance between two feature maps smaller than a threshold. We define the distance between ith entries of two feature maps as $|f_t^i - f_s^i|$, where f_t^i and f_s^i are the ith element of teacher and student feature maps. If this distance is smaller than a threshold (we set 0.3 in this paper), then these two entries match. We evaluate on the validation set of WIDER FACE. We compare the results between full-precision and quantized network. The horizontal axis represents bin of matching ratio, i.e. the percentage of matched entries in a RoI. Figure 5 demonstrates the results. The result shows that quantization operation can increase matching ratio of RoIs and promote feature maps matching process. Thus, quantization operation can help mimic learning.

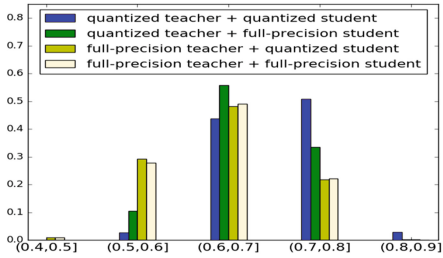

Fig. 5. Histogram of matching ratio. The plot suggests that using quantiation operation both on teacher and student networks can help student network's feature maps to better match teacher network's. The horizontal axis represents bin of matching ratio, *i.e.* the percentage of matched entries in a RoI. The vertical axis represents the frequency of RoIs within this bin.

Ablation Study on Quantization Method. Different quantization method will bring different effects. The quantization methods we use in our work is uniform quantization. Another popular quantization method is power of 2 quantization, constraining the output to be either zero or power of 2. Table 6 illustrates the comparison of uniform quantization and power of 2 quantization. Teacher networks using different quantization methods have similar performance. However, the student network using uniform quantization is much better than using power of 2 quantization. We think this is probably because that our mimic learning is based on RoIs and strong responses in these areas are more important. So we should describe large number more accurately. And for power of 2 quantization method, it describes small numbers (*e.g.* the number less than 1) accurately but roughly for large numbers. Thus, uniform quantization method is more reasonable and can bring better results.

Table 6. Uniform Quantization *vs.* Power of 2 Quantization: Using uniform quantization as quantization method can get better result than using power of 2 quantization.

Model	Quantization method	Easy	Medium	Hard
VGG-1-4 (teacher)	Power of 2	83.9	64.8	27.8
	Uniform(stride:1)	83.7	65.0	27.4
VGG-1-32 (student)	Power of 2	73.0	59.5	26.6
	Uniform(stride:1)	73.9	62.1	27.6

4.2 Experiments on Pascal VOC Dataset

We also carry out experiments on more complicated common object detection task. In this section we implement our approach on Resnet18 with Faster R-CNN detector for Pascal VOC object detection benchmark [42]. The experiments show that Quantization Mimic can extend to more complicated tasks.

Following [38], we use Pascal VOC 2007 test set for test and trainval images in VOC 2007 and VOC 2012 for training (07+12). Hyperparameters in Faster R-CNN are same as [38]. Mean Average Precision (mAP) is used as the criterion to evaluate the performance of model. We use Resnet18 with Faster R-CNN framework as teacher networks. And Resnet18-1-16 with Faster R-CNN framework are selected as student networks accordingly. We aim at improving the performance of the student works using Quantization Mimic method.

Main Results. First we compare the model using Quantization Mimic method with the model trained from scratch . Because of the poor learning ability of very tiny models, it is difficult to train them on complicated task, such as classification on Imagenet [43] and common object detections on Pascal VOC. Our method can improve a large margin of performance for very tiny networks on common object detections. Table 7 illustrates the results. We suggest that our method increase mAP 6.5 points for Resnet18-1-16 with Faster R-CNN framework. Relatively, we improve the performance for 16.0%. The experiments also show that Quantization Mimic is easy to implement and can be extended to different frameworks.

We also do experiments compared with other accelerating and compressing methods. Same as the experiments on WIDER FACE dataset, we compare our method with Li *et al.* [4], who only use mimic learning. In Table 8, our method outperforms than Li *et al.* [4]. Our results are 2.4 points higher than our backbone, Li *et al.* [4] on Resnet-1-16, which is a large margin.

Ablation Study on Mimic Loss Weight. We propose that very tiny networks can be sensitive to loss weight in multi-loss task. We do this experiment on Resnet18-1-16 to find a suitable mimic loss weight. In Table 9, we can see that the result of $\lambda = 1$ is much better than the result of $\lambda = 0.1$ and $\lambda = 10$. We suggest that if mimic loss is too small (*e.g.* $\lambda = 0.1$) , the effectiveness of

Table 7. The comparision between Resnet18-1-16 with Faster R-CNN detector fine-tuned on Imagenet dataset and using Quantization Mimic method. Our method can also bring huge improvement for very tiny networks on complicated common object tasks.

Model	Solution	mAP
Resnet18	Full-precision	72.9
	Quantized	73.3
Resnet18-1-16	Scratch	40.5
	Quantization Mimic	47.0

Table 8. Comparison with backbone on Resnet18-1-16 with Faster R-CNN framework. Our method outperforms our backbone Li *et al.* [4] methods for Resnet18 (higher is better).

Model	Solution	mAP
Resnet-1-16	Li *et al.* [4](only mimic)	44.6
	Quantization mimic	**47.0**

mimic learning will decline. However, if we set mimic loss weight too large (*e.g.* $\lambda = 10$), the very tiny network will mainly focus the gradient produced by mimic loss and ignore other gradients. And we call this phenomenon as 'gradient focus' phenomenon.

Table 9. Mimic Loss Weight λ: The results show that very tiny networks are sensitive to the mimic loss weight. Either too large or too small loss weight will decrease the effectiveness of mimic learning.

Model	Mimic loss weight	mAP
Resnet18-1-16	10	44.1
	1	**47.0**
	0.1	43.0

5 Conclusion

In this paper, we propose Quantization Mimic to improve the performance of very tiny CNNs. We show quantization operation on both teacher and student networks can promote feature map matching. It becomes easier for the student network to learn after quantization operation. The experiments on WIDER FACE and Pascal VOC dataset demonstrate that quantization mimic outperforms state-of-the-art methods. We hope our approach can facilitate future research on training very tiny CNNs for cutting-edge applications.

References

1. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks: training deep neural networks with weights and activations constrained to +1 or −1 (2016). arXiv preprint arXiv:1602.02830
2. Howard, A.G., et al.: Mobilenets: efficient convolutional neural networks for mobile vision applications (2017). arXiv preprint arXiv:1704.04861
3. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
4. Li, Q., Jin, S., Yan, J.: Mimicking very efficient network for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
5. Zeng, X.: Crafting GBD-Net for object detection. IEEE Trans. Pattern Anal. Mach. Intell. **40**, 2109–2123 (2017)
6. Liu, Y., Li, H., Yan, J., Wei, F., Wang, X., Tang, X.: Recurrent scale approximation for object detection in CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
7. Yan, J., Yu, Y., Zhu, X., Lei, Z., Li, S.Z.: Object detection by labeling superpixels. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
8. Yan, J., Lei, Z., Wen, L., Li, S.Z.: The fastest deformable part model for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
9. Ouyang, W., et al.: DeepID-Net: deformable deep convolutional neural networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
10. Ouyang, W., Wang, K., Zhu, X., Wang, X.: Chained cascade network for object detection. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
11. Wang, X., Han, T.X., Yan, S.: An HOG-LBP human detector with partial occlusion handling. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2009)
12. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
13. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
14. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 346–361. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_23
15. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV)(2015)
16. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
17. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2

18. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
19. Zhang, T., Qi, G.J., Xiao, B., Wang, J.: Interleaved group convolutions for deep neural networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
20. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: An extremely efficient convolutional neural network for mobile devices (2017). arXiv preprint arXiv:1707.01083
21. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet classification using binary convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 525–542. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_32
22. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: DoReFa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
23. Courbariaux, M., Bengio, Y., David, J.P.: BinaryConnect: training deep neural networks with binary weights during propagations. In: Advances in Neural Information Processing Systems (NIPS) (2015)
24. Juefei-Xu, F., Boddeti, V.N., Savvides, M.: Local binary convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
25. Zhou, A., Yao, A., Guo, Y., Xu, L., Chen, Y.: Incremental network quantization: towards lossless CNNs with low-precision weights. In: International Conference of Learning Representation (ICLR) (2017)
26. Alvarez, J.M., Salzmann, M.: Learning the number of neurons in deep networks. In: Advances in Neural Information Processing Systems (NIPS) (2016)
27. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. In: Advances in Neural Information Processing Systems (NIPS) (2016)
28. Anwar, S., Sung, W.: Compact deep convolutional neural networks with coarse pruning (2016). arXiv preprint arXiv:1610.09639
29. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. In: International Conference of Learning Representation (ICLR) (2017)
30. Molchanov, P., Tyree, S., Karras, T., Aila, T., Kautz, J.: Pruning convolutional neural networks for resource efficient inference. In: International Conference of Learning Representation (ICLR) (2017)
31. Guo, Y., Yao, A., Chen, Y.: Dynamic network surgery for efficient DNNs. In: Advances In Neural Information Processing Systems (NIPS) (2016)
32. Han, S., et al.: EIE: efficient inference engine on compressed deep neural network. In: Proceedings of the 43rd International Symposium on Computer Architecture (ISCA) (2016)
33. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: Advances in Neural Information Processing Systems (NIPS) (2015)
34. Yang, T.J., Chen, Y.H., Sze, V.: Designing energy-efficient convolutional neural networks using energy-aware pruning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
35. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network (2015). arXiv preprint arXiv:1503.02531

36. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: FitNets: hints for thin deep nets. In: International Conference of Learning Representation (ICLR) (2015)
37. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: Object detection via region-based fully convolutional networks. In: Advances In Neural Information Processing Systems (NIPS) (2016)
38. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NIPS) (2015)
39. Jia, Y., et al. : Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia (ACMMM) (2014)
40. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
41. Yang, S., Luo, P., Loy, C.C., Tang, X.: Wider face: a face detection benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
42. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The Pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
43. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2009)