Shoot to Know What: An Application of Deep Networks on Mobile Devices

Jiaxiang Wu, Qinghao Hu, Cong Leng, and Jian Cheng
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
Beijing, China, 100190
{jiaxiang.wu, qinghao.hu, cong.leng, jcheng}@nlpr.ia.ac.cn

Abstract

Convolutional neural networks (CNNs) have achieved impressive performance in a wide range of computer vision areas. However, the application on mobile devices remains intractable due to the high computation complexity. In this demo, we propose the Quantized CNN (Q-CNN), an efficient framework for CNN models, to fulfill efficient and accurate image classification on mobile devices. Our Q-CNN framework dramatically accelerates the computation and reduces the storage/memory consumption, so that mobile devices can independently run an ImageNet-scale CNN model. Experiments on the ILSVRC-12 dataset demonstrate $4 \sim 6\times$ speed-up and $15 \sim 20\times$ compression, with merely one percentage drop in the classification accuracy. Based on the Q-CNN framework, even mobile devices can accurately classify images within one second.

Introduction

In recent years, convolutional neural networks (CNNs) have become the state-of-the-art methods in various computer vision tasks, especially large-scale image classification. From AlexNet (Krizhevsky, Sutskever, and Hinton 2012) to VGG-16 (Simonyan and Zisserman 2015), the classification accuracy consistently improves as the network grows deeper. Nevertheless, the model complexity also increases correspondingly. Even as a relatively small convolutional network, AlexNet involves 60 million parameters and over 729 million FLOPs[1] are required to classify a single image. Such costly computation complexity makes the deployment of CNN models unaffordable for common PCs and mobile devices. Hence, it is urgent to develop acceleration and compression techniques for CNN models.

In this demo, we adopt an efficient framework, namely Quantized CNN (Q-CNN), to realize fast and accurate on-site image classification on mobile platforms. It is often difficult, if not impossible, to run a CNN model on mobile devices, due to the limited computation ability, storage, and memory space. However, with our proposed Q-CNN framework, the required computation resource of convolutional networks can be dramatically reduced, with only minor degradation in the classification accuracy.

Copyright © 2016, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

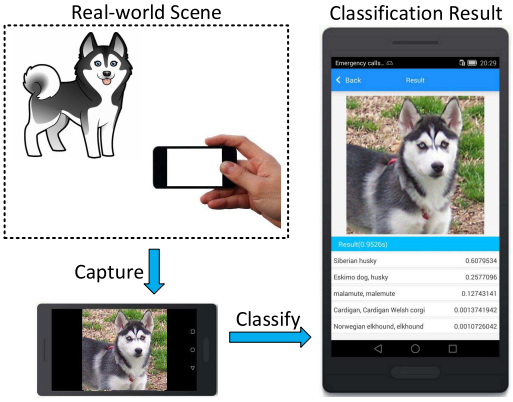

Figure 1: Our demo system allows the user to accurately classify arbitrary photos captured by the camera with a convolutional neural network.

To verify the effectiveness of our Q-CNN framework, we evaluate its classification performance on the ILSVRC-12 benchmark (Russakovsky et al. 2015), based on AlexNet (Krizhevsky, Sutskever, and Hinton 2012), CNN-S (Chatfield et al. 2014), and VGG-16 (Simonyan and Zisserman 2015). For all three networks, our approach offers $4 \sim 6\times$ speed-up and $15 \sim 20\times$ compression, at a cost of less than 1% drop in the top-5 classification accuracy.

We develop an Android application to demonstrate the on-site CNN-based image classification on mobile devices, as depicted in Figure 1. To the best of our knowledge, this should be the very first attempt to deploy an ImageNet-scale network on mobile devices. With the efficient Q-CNN framework, image classification with AlexNet on a common smartphone only takes 0.95s and the storage/memory consumption is merely 13MB/75MB.

System Framework

In our demo system, the user can arbitrarily take a photo (or load an existing photo) with the smartphone, and then recognize objects within the photo using convolutional networks. In Figure 2, we depict the workflow of our image classification, which mainly involves two procedures: pre-processing, and image classification with the Q-CNN framework.

[1] FLOPs: number of FLoating-point OPerations.

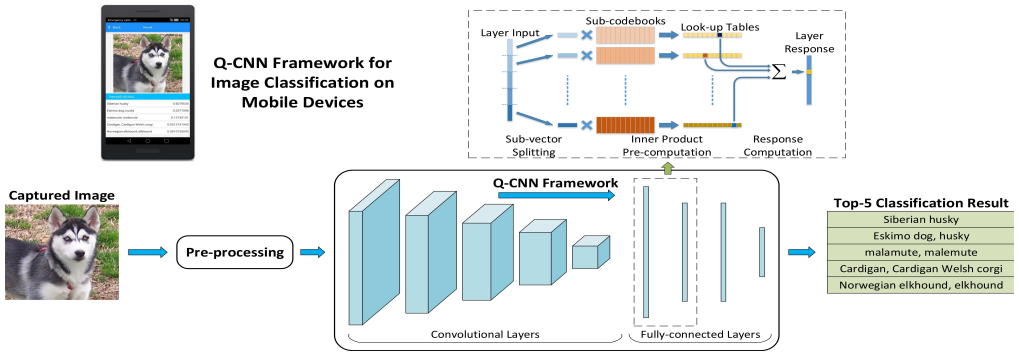

Figure 2: The workflow of the image classification process on mobile devices, based on our Q-CNN framework.

Pre-processing

A few pre-processing operations are required before feeding the captured photo into the convolutional network for classification. Firstly, we resize the image to 256×256, or resize the shorter side to 256, depending on the choice of CNN models. Then we crop out the central patch from the resized image and remove the mean value for each pixel.

Image Classification with Q-CNN

The Q-CNN framework we adopted here takes a CNN model (trained on ImageNet) as the reference model, and then generates a quantized version of it. The major advantage is that Q-CNN can speed-up the computation and reduce the storage/memory consumption dramatically, with only minor performance degradation. Both the filter kernels (convolutional layers) and weighting matrices (fully-connected layers) are quantized with codebooks. The quantization is learned via minimizing the estimation error of each layer's response. After quantization, only codebooks and quantization indexes are stored, which is much more efficient in storage/memory than storing network parameters.

During the test-phase, instead of the standard convolutional and matrix multiplication operations, we approximately compute each element in the layer response as the addition of multiple inner products, which can be directly fetched from a pre-computed look-up table. Therefore, the computation can be significantly accelerated.

Experiments

We evaluate our Q-CNN framework's performance on the ILSVRC-12 benchmark for image classification. We report the top-5 classification error rate of 50k validation images, using single-view testing (central patch only). Three CNN models are used (AlexNet, CNN-S, and VGG-16), and we report the performance on these networks in Table 1.

We have also developed an Android application to integrate our Q-CNN framework into mobile devices, which should be the first attempt to run an ImageNet-scale CNN model on mobile platforms. In Table 2, we compare the performance of the original and quantized CNN models on a Huawei® Mate 7 smartphone, equipped with an 1.8GHz Kirin 925 CPU. The Q-CNN framework requires much less computation resource, while the loss in the classification accuracy is no more than 1%.

Table 1: The speed-up/compression rates and the increase in the top-5 classification error rates of our Q-CNN framework.

Model (Top-5 Err.)	Speed-up	Compression	Top-5 Err. ↑
AlexNet (19.74%)	4.05×	15.40×	0.84%
	4.15×	18.76×	0.97%
CNN-S (15.82%)	5.69×	16.32×	0.81%
	5.78×	20.16×	0.85%
VGG-16 (10.06%)	4.05×	16.55×	0.58%
	4.06×	20.34×	0.65%

Table 2: Comparison on the running time, storage, memory consumption, and classification error rates of CNN models.

Model		Time	Storage	Memory	Top-5 Err.
AlexNet	CNN	2.93s	232.56MB	264.74MB	19.74%
	Q-CNN	**0.95s**	**12.60MB**	**74.65MB**	**20.70%**
CNN-S	CNN	10.58s	392.57MB	468.90MB	15.82%
	Q-CNN	**2.61s**	**20.13MB**	**129.49MB**	**16.68%**

Acknowledgement

This work was supported in part by 863 Program (Grant No. 2014AA015100), and National Natural Science Foundation of China (Grant No. 61332016, 61170127).

References

Chatfield, K.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference (BMVC)*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 1106–1114.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)* 1–42.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*.