Title Page

Contents

I

This is a Part

CHAPTER 11

Guided Design for Efficient On-device Object Detection Model

The LOW-POWER COMPUTER VISION (LPCV) challenge is an annual competition for the best technologies in image classification and object detection measured by both efficiency (execution time and energy consumption) and accuracy (precision/recall). Our Amazon team has won three awards from LPCV challenges: 1st prize for interactive object detection challenge in 2018 and 2019 and 2nd prize for interactive image classification challenge in 2018. This paper is to share our award-winning methods, which can be summarized as four major steps. First, 8-bit quantization friendly model is one of the key winning points to achieve the short execution time while maintaining the high accuracy on edge devices. Second, network architecture optimization is another winning keypoint. We optimized the network architecture to meet the 100ms latency requirement on Pixel2 phone. The third one is dataset filtering. We removed the images with small objects from the training dataset after deeply analyzing the training curves, which significantly improved the overall accuracy. And the forth one is non-maximum suppression optimization. By combining all the above steps together with the other training techniques, for example, cosine learning function and transfer learning, our final solutions were able to win the top prizes out of large number of submitted solutions across worldwide.

11.1 INTRODUCTION

Competitions encourage diligent development of technologies. Historical examples include Ansari XPRIZE competitions for suborbital spaceflight, numerous Kaggle competitions such as identifying salt deposits beneath the Earth's surface from seismic images, and the PASCAL VOC, ILSVRC (a.k.a ImageNet), and COCO [8] competitions for accurate computer vision techniques. The Low-Power Image Recognition Challenge (LPIRC) aims to accelerate the development of computer vision solutions that are fast, accurate, and low-power for edge devices.

Started in 2015, LPIRC [2], [11] is an annual competition identifying the best computer vision solutions for classifying and detecting objects in images, while using as little energy as possible. Although many competitions are held every year, LPIRC is the only one integrating both state-of-the-art computer vision techniques and low power requirement. In June 2018, the competition was held very successfully co-located with CVPR 2018 in Salt Lake City. Due to the large number of contestants and strong interests of sponsors, the LPIRC was held again in November 2018 (called LPIRC-II), and the winners of which were announced at NeurIPS 2018. In June 2019, the competition was held co-located with CVPR 2019 in Long Beach, California. LPIRC offered two tracks in 2018, Track 1 is sponsored by Google focusing on the evaluation of accuracy and execution time using TensorFlow and TFLite running on Pixel2 phone. Track 2 is sponsored by Facebook focusing on the evaluation of energy consumption using the Caffe2 framework running on NVIDIA Jetson TX2. We as Amazon team participated the Track 1 in LPIRC-II in 2018 and LPIRC-2019 due to the main interest on the edge phone devices. Here is the description of LPIRC Track 1.

11.1.1 LPIRC Track 1 in 2018 and 2019

The goal of LPIRC is to achieve the best accuracy within the wall-time constraint by evaluating the accuracy and the execution time using TensorFlow models. Although no power or energy constraint is explicitly measured for this track, latency correlates reasonably with energy consumption. This track is further divided into three categories (with separate awards):

Real-time image classification: This is the original task where focusing on ImageNet classification models operating at 30 ms/image. Submissions are evaluated based on classification accuracy/time while focusing on the real-time regime running on Google's Pixel2 phone.

Interactive image classification: Similar to the above category but the latency budget is extended to 100 ms/image.

Interactive object detection: The newly introduced category in 2018 focusing on COCO object detection models at 100 ms/image. Submissions are evaluated based on detection mAP and latency running on Google's Pixel2 phone.

11.1.2 Three Awards for Amazon team

After diligent hard-work, our Amazon team won the two awards out of all submitted solutions at 2018: the first place for interactive object detection challenge and the second place for interactive image classification challenge. Then in LPIRC-2019, our team won the first place again for interactive object detection challenge. We would like to describe the details of our winning methods in this paper. The rest of the paper will be organized as follows: section 11.2 will talk about some technical details

about getting an efficient deep learning model for edge devices. In section 11.3, we illustrate the winning methods and show the experimental results. And section 11.4 concludes this paper.

11.2 BACKGROUND

Quantization is crucial for deep learning inference on edge devices, which have very limited budget for power and memory consumption. Such platforms often rely on fixed-point computational hardware blocks, such as Digital Signal Processor (DSP), to achieve higher power efficiency over floating-point processor, such as CPU and GPU. Although quantization may not impact inference accuracy for many over-parameterized deep leanring models, such as VGGNet [7], InceptionNet [13], ResNet [3] and etc. deploying those models on edge devices is not quite feasible due to large computational complexity and memory footprint.

Recently, there has been tramendous progress on the innovation of many lightweight deep learning networks. These models can trade off accuracy with efficiency by replacing conventional convolution with depthwise separable convolution. For example, the MobileNets [4] [5] drastically shrink the parameter size and memory footprint, thus are getting increasingly popular on edge devices. The downside is that the separable convolution core layer in MobileNets causes large quantization loss, thus results in significant feature representation degradation in the 8-bit fixed-point inference pipeline.

To demonstrate the quantization issue, we selected the TensorFlow implementation of MobileNets [4] [5] and InceptionV3 [13], and compared their accuracies on floating-point pipeline against 8-bit fixed-point pipeline. The results are summarized in Table 11.1. The top-1 accuracy of InceptionV3 drops slightly after applying the 8-bit quantization, while the accuracy loss is significant for MobileNetV1, MobileNetV2, and MobileNetV1-SSD [9]. Although the accuracy of InceptionV3 is much less impacted by 8-bit quantization, its computational complexity is eight times more than MobileNetV1, which means it is not well suited to use InceptionV3 on edge devices. Therefore, how to make the lightweight deep learning models quantization friendly is essentical to achieve a good tradeoff between accuracy and latency. There are two approaches: one is the quantization-aware training proposed by [6], and the other one is the re-architectured MobileNet network proposed by [12]. Due to the availablity of TFLite quantization toolkit, we utilize the quantization-aware training approach in both interactive classification challenge and interactive detection challenge.

11.3 AWARD-WINNING METHODS

In this section, we will describe the steps to achieve the award-winning models: First, we built the proper deep learning networks and trained the networks by 8-bit quantization-aware framework. Second, we optimized the network architecture to meet the latency requirement while maintaining the high accuracy. Third, we applied dataset filtering to training images. Forth, we optimized the non-maximum suppression in detection model by choosing the best threshold. Besides the above steps,

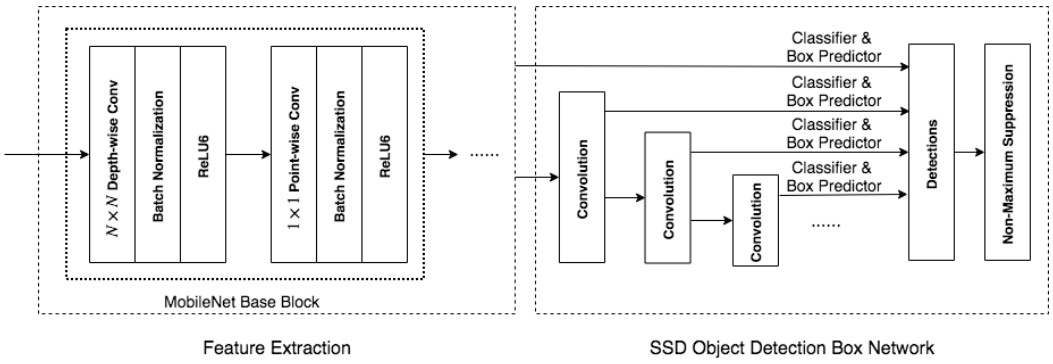

Figure 11.1 MobileNet SSD Object Detection Network Architecture where MobileNet V1/V2 is used as backbone, and SSD is used as detection head. We chose MobileNet-SSD to meet the latency requirement while maintaining the high accuracy on Pixel2 phone.

we also applied the other training techniques to further improve the accuracy, for example, cosine learning function and transfer learning.

11.3.1 Quantization Friendly Model

As mentioned in section 11.2, we built our deep learning models for both classification and detection based on MobileNets [4] [5]. For the classification task, we chose the MobileNetV2 [5] architecture with best configuration of the input resolution and depth multiplier. For the detection task, we used MobileNetV1 [4] as the base network in 2018, and MobileNetV2 [5] as the base network in 2019, and add SSD [9] layers after the base network for object detection as illustrated in Figure 11.1. We trained the models by the quantization-aware framework proposed by [6] to achieve high efficiency for both tasks on Pixel2 phone.

11.3.2 Network Architecture Optimization

To speed up the training process, we adopted the idea of transfer learning by starting the training with a pre-trained model in tensorflow [1]. The pre-trained weights was obtained on the 300×300 image resolution. However, the quantized MobileNetV1 SSD model with 300×300 input resolution took 145 ms on Pixel2 phone, which exceeded the speed budget by around 45 ms. Therefore, we need to optimize the detection network to meet the latency requirement. We also found that it's important to make the layer geometry as multiple of eight due to the underneath 8-bit runtime optimazation in the convolution kernels. In the following sections, we will introduce the training approaches and the optimal model architecture to achieve the best results on the LPIRC competition in both 2018 and 2019.

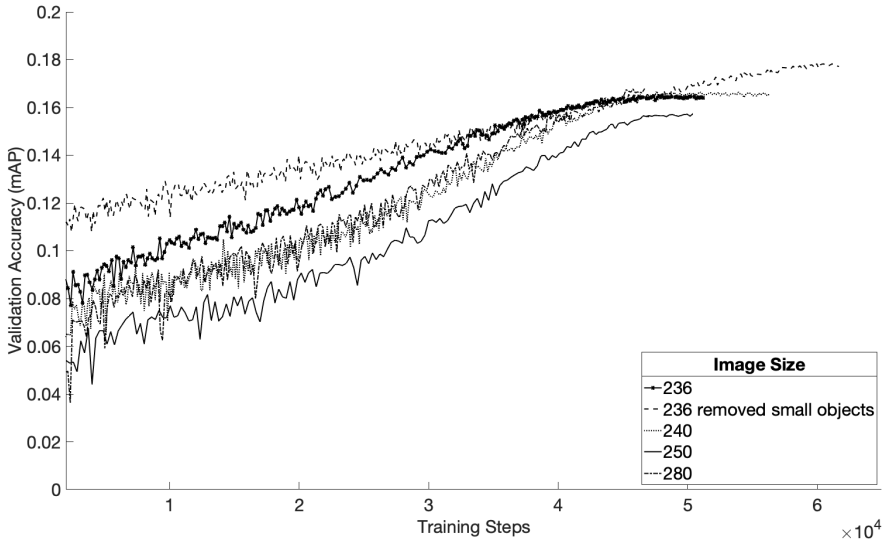

Figure 11.2 Object detection accuracy (mAP) vs. input image size. The 236×236 model reached almost the same accuracy as the 280×280 model, which means we can significantly reduce the computation complexity, while maintaining the similar accuracy. The other interesting observation is that 250×250 detection model has even lower accuracy than 240×240 detection model and 236×236 detection model. Our understanding is that the layer geometry of the network needs to be multiple of eight to reduce the possible negative impact of padding or striding. The accuracy of 236×236 model got further improved after removing the training images with small objects.

11.3.3 Training Hyper-parameters

For detection task, when we started to train the quantized MobileNetV1 SSD model using COCO dataset [8] and Tensorflow [1] framework with GPUs, it is very critical to choose the proper training parameters so that the network can converge to an optimal state in the fastest speed. Based on recent literature [10], stochastic gradient descent with warm restart has gained a lot of attention and been proven to be fast and stable in many empirical studies. In particular, we chose to use a variant of this technique - cosine decay learning rate for our fine-tune network training.

11.3.4 Optimal Model Architecture

The fastest way to improve the latency of a deep learning network is to reduce the input image resolution. However, changing the input image resolution without re-training the network will significantly degrade the model accuracy. Hypothetically, if the change of resolution is not drastic, fine-tuning the network to adapt to the new image resolution should be viable. To validate, we have run the experiments to fine-tune 300×300 detecion model to smaller image sizes. The mAP accuracy of pre-

trained 300 × 300 detection model is 18.0%. From Figure 11.2, the object detection mAP accuracy has decreased significantly for smaller image sizes before fine-tuning. As expected, fine-tuning the networks eventually improved the mAP accuracy over the training steps. Figure 11.2 and Table 11.2 also show the trade offs between speed and accuracy. As shown in Figure 11.2, changing input image resolution with fine-tuning does not have significant impact to its accuracy. The 236 × 236 detection model reached almost the same accuracy as the 280 × 280 detection model with only 0.02 accuracy loss. It suggests that we can significantly reduce the computatial complexity, while maintaining the similar accuracy. The other interesting observation is that 250 × 250 detection model has even lower accuracy than 240 × 240 detection model and 236 × 236 detection model. Our understanding is that the layer geometry of the network needs to be multiple of eight to reduce the possible negative impact of padding or striding. By fine-tuning the network to 236 × 236 pixels, we achieved the speed of 93 ms on Pixel2 phone while maintaining a good accuracy in 2018.

11.3.5 Neural Architecture Search

Neural architecture search (NAS) [14] have been shown to be able to automatically acquire a neural architecture that achieves competitive performance. Despite its great success, most of the NAS approaches are focused on searching for a universal model for all computer vision applications using a proxy task (e.g., searching a model on ImageNet classification but applying it on semantic segmentation task). Moreover, most NAS approaches adopt proxy latency such as FLOPS instead of real on-device latency during neural architecture searching to tradeoff the model accuracy and overhead. Unfortunately, it has been demonstrated that using proxy task and proxy latency is inaccurate and the searched model may achieve poor performance when being applied on different tasks and platforms. In LPIRC-2019, we performed neural architecture search directly on object detection task and used on-device (Pixel2 Phone) latency for searching to ensure that the searched model meets the latency requirement while achieving the optimal accuracy.

11.3.6 Dataset Filtering

COCO [8] dataset is a very challenging dataset for object detection, because it contains objects in a wide range of scales. It can be divided into three categories of training images with regard to the object size: large, medium, and small. Let A be the area size of the object in each category, the categories are defined as following:

$$\begin{aligned} Large &: A > 96 \times 96 \\ Medium &: 32 \times 32 < A < 96 \times 96 \\ Small &: A < 32 \times 32 \end{aligned} \tag{11.1}$$

The size 96/32 is determined by the COCO [8] dataset. As seen in Figure 11.3, the impact to objects in each category is very different. The mAP accuracy of large objects is significantly better than medium and small objects. The mAP accuracy improvement in small objects is too small and it has almost no contribution to the

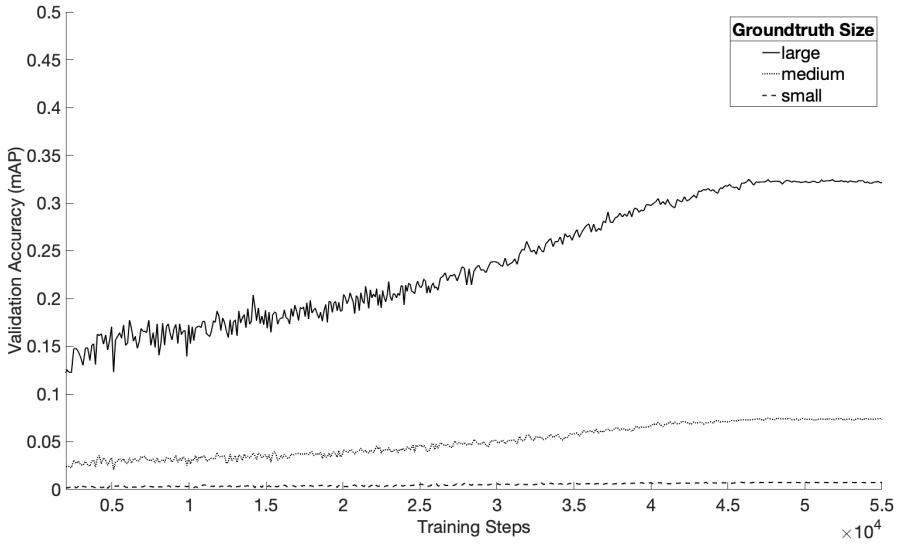

Figure 11.3 Object detection accuracy of objects with different sizes: Entire training steps

overall mAP. However, the error back-propagation from small objects is weighted equally as other objects. In the extreme case, the small objects are likely to be chosen as the hard examples and largely contribute to the back-propagated error. Due to the noise generated by image down-sampling, small objects would have a very low signal-to-noise ratio and the quality of objective error measured from small objects are significantly degraded. Figure 11.4 shows an example of the increase in mAP of small objects would negatively impact the mAP of medium and large objects. Therefore, we removed the images with small objects from our training dataset to focus more on large and medium objects. As shown in Figure 11.2, this dataset filtering optimization improved the object detection accuracy (mAP) by more than 7.8% relatively, and increased the overall mAP from 16.5% to 17.8%.

11.3.7 Non-maximum Suppression Threshold

Lastly, the non-maximum suppression threshold [9] can be tuned to further trade off the speed versus accuracy. The 236×236 detection model has left us a few millisecond room to reach the latency budget 100ms, which is an ideal range for fine-tuning non-max suppression threshold. Table 11.3 summarizes the experimental results of applying different non-max suppression thresholds to the optimized 236×236 MobileNetV1 SSD model. Finally, we chose to use threshold 0.15 which has the best trade-off between accuracy and latency.

Table 11.1 Float vs. 8-bit accuracy/latency comparison on ImageNet2012 and COCO validatation dataset

Networks	Float Pipeline	8-bit Pipeline	Comments
InceptionV3	78.00%/1433ms	76.92%/637ms	Only standard convolution
MobileNetV1	70.50%/160ms	1.8%/70.2ms	Mainly separable convolution
MobileNetV2	71.9%/117ms	1.23%/80.3ms	Mainly separable convolution
MobileNetV1-SSD	21%/331ms	6.7%/145ms	Mainly separable convolution

Table 11.2 Input Image Resolution vs. Latency on Pixel2 Phone

Input Image Resolution (pixels)	300	280	270	240	236
Latency (ms)	145	131	120	95	93

Table 11.3 Non-maximum suppression threshold vs. accuracy on 236 × 236 MobileNetV1 SSD

Non-max suppression threshold	0.30	0.25	0.20	0.15	0.10
Accuracy (mAP %)	17.85	18.11	18.29	18.40	18.43

Table 11.4 Comparison between our optimized 236 × 236 detection model vs. original 300 × 300 detection model

Model	Input Resolution	Accuracy (mAP %)	Latency (ms)
Original detection model	300 × 300	18.0	145
Our optimized detection model	236 × 236	18.4	93

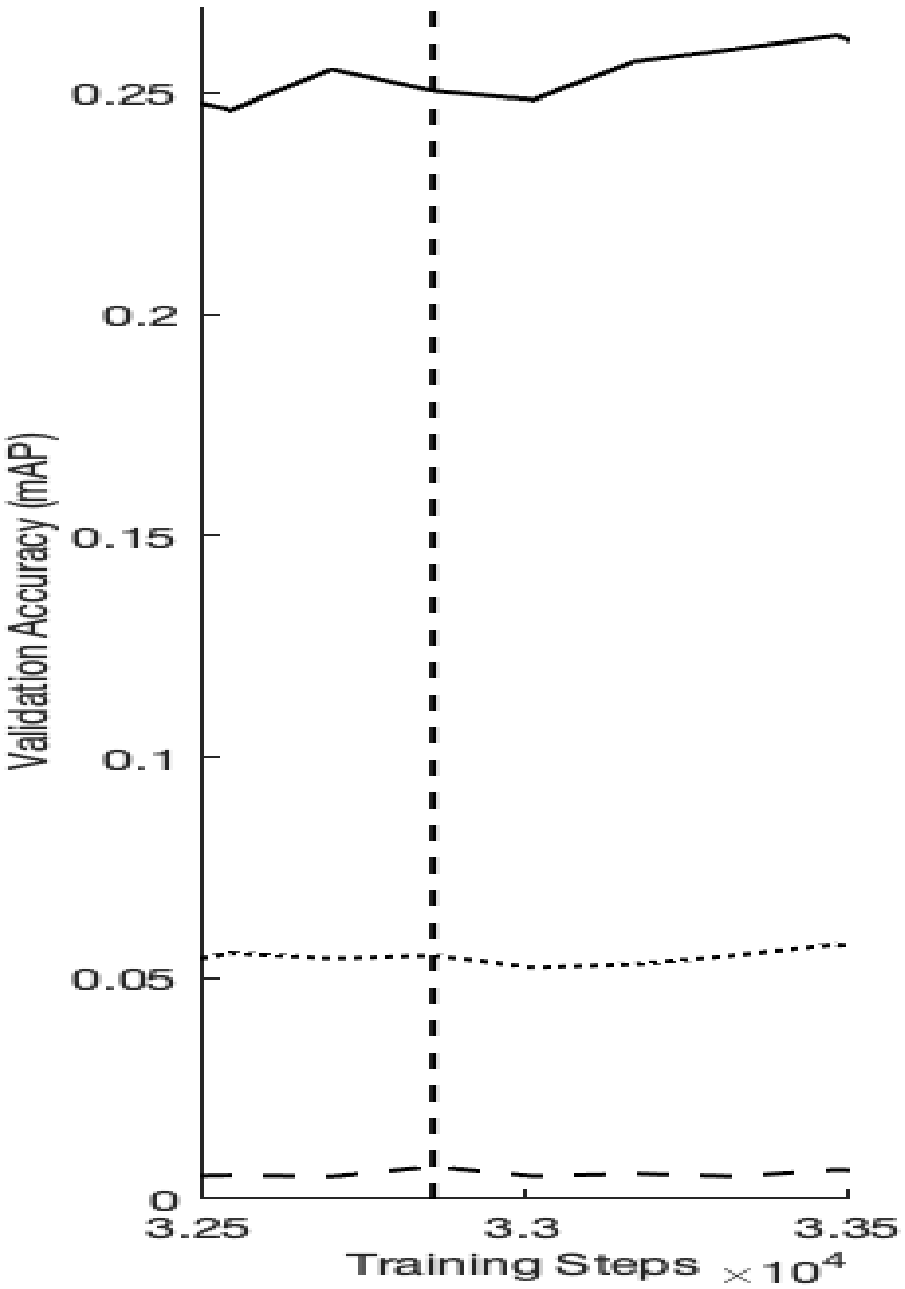

Figure 11.4 Object detection accuracy of objects with different sizes: A zoom-in section

11.3.8 Combination

By combing all the above methods, we can achieve the desired detection model with latency within 100 ms and the highest accuracy. As shown in Table 11.4, our optimized 236×236 detection model can achieve even higher accuracy than original 300×300 detection model with much faster model execution on Pixel2 phone. This detection model won the first place for interactive detection challenge on LPIRC COCO hold-out test dataset in 2018, and further optimized model won the first prize again in 2019. With similar methods applied to ImageNet classification except the dataset filtering and non-maximum suppression optimization, we also trained a very good 8-bit quantization friendly classification model, which won the second place for interactive classification challenge LPIRC-II ImageNet hold-out test dataset in 2018.

11.4 CONCLUSION

This paper explained our award-winning methods from the highly participated public low power image recognition challenge 2018 and 2019. As computer vision is widely used in many battery-powered edge devices, the needs for low-power computer vision will become increasingly important. Our Amazon team will continue to focus on low power computer vision area and engage with LPIRC community.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, and Craig Citro. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Kent Gauen, Ryan Dailey, Yung-Hsiang Lu, Eunbyung Park, Wei Liu, Alexander C. Berg, and Yiran Chen. Three years of low-power image recognition challenge: Introduction to special session. *DATE*, .2018.83420998, 2018.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, arXiv:1512.03385, 2016.

[4] 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. *arXiv*, abs/1704.04861, 2017.

[5] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenetv2: Inverted residuals and linear bottlenecks. *arXiv*, abs/1801.04381, 2018.

[6] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *CoRR*, abs/1712.05877, 2017.

[7] K.Simonyan and A.Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556, 2014.

[8] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed. Ssd: Single shot multibox detector. arXiv:1512.02325, 2015.

[10] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016.

[11] Yung-Hsiang Lu, Alexander C. Berg, and Yiran Chen. Low-power image recognition challenge. *AI Magazine*, Vol 39 No 2: Summer 2018, 2018.

[12] Tao Sheng, Chen Feng, Shaojie Zhuo, Xiaopeng Zhang, Liang Shen, and Mickey Aleksic. A quantization-friendly separable convolution for mobilenets. *arXiv*, arXiv:1803.08607, 2018.

[13] C. Szegedy, W. Liu, and Y. Jia. Going deeper with convolutions. *CVPR*, arXiv:1409.4842, 2015.

[14] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.