A Retrospective Evaluation of Energy-Efficient Object Detection Solutions on Embedded Devices

Ying Wang, Zhenyu Quan, Jiajun Li, Yinhe Han, Huawei Li, Xiaowei Li
State Key Laboratory of Computer Architecture
Institute of Computing Technology Chinese Academy of Sciences, Beijing, P.R. China
{wangying2009, zlei, yinhes, lihuawei, lxw}@ict.ac.cn

Abstract—**The field of image and video recognition has been propelled by the rapid development of deep learning in recent years. With its fascinating accuracy and generalization ability, deep CNNs have shown remarkable performance in large-scale and real-life image dataset. However, accommodating computation-intensive CNN-based image detection frameworks on power-constrained devices is considered more challenging than desktop or warehouse computing systems. Instead of emphasizing purely on detection accuracy, Low Power Image Recognition Challenge (LPIRC) is initiated to highlight the energy-efficiency of different image recognition solutions, and it witnesses the advancement of cost-effective image recognition technology in aspects of both algorithmic and architecture innovation. This paper introduces the cost-effective CNN-based object detection solutions that reached an improved tradeoff between energy and accuracy for mobile CPU+GPU SoCs, which is the winner of LPIRC2016, and it also analyzes the implications of both recent hardware and algorithm advancement on such a technique. It is demonstrated in our evaluation that the performance growth of embedded SoCs and CNN models have clearly contributed to a sheer growth of mAP/WH in current CNN-based object detection solutions, and also shifted the balance between accuracy and energy-cost in the contest solution design when we seek to maximize the efficiency score defined by LPIRC through design parameter exploration.**

Keywords—Image Recognition; CNN; Energy; GPU

I. Introduction

The recent advancement of computer vision and image processing has gained substantial momentum from the development of deep learning technology. Deep convolutional neural networks are enabling the computers to exceed the accuracy level of human raters in ImageNet classification. It is expected to change the video-related applications in many areas, including security surveillance and natural interaction.

Although convolutional neural networks (CNNs) are showing increasing power and accuracy on a variety of image databases, the growth of network scale and depth, agitated by the complexity of image content, still poses a significant challenge to the consolidation of CNN-based image recognition solutions on power-constrained computing devices, which struggle to offer real-time CNN inference capability with limited computational and memory resources. Thus, a plenty of real-time CNN-based detection frameworks have been proposed to enable fast image recognition in GPGPU platform [1] [2]. However, these proposals are still requiring the hardware to provide high-throughput processing capability and sufficient memory space to deal with the inference task. For mobile devices that emphasize on energy consumption, the computational overhead of such CNN-based solutions have to be reduced for improved system availability and sustainable battery life.

For low-end mobile devices that have less computation strength and power budget, an appropriate CNN-based detection framework should be able to exploit the capability of hardware better and increase the power utility using architectural and algorithmic optimization. Low Power Image Recognition Challenge (LPIRC) is initiated to promote such research on energy-efficient object detection technology, and it particularly highlights the tradeoff between accuracy and power consumption in the detection solutions [3]. In contrast to state-of-the-art CNN-based detection frameworks focused on accuracy improvement, detection solutions that are aware of the network computational overhead are expected to have a better score in terms of mAP/WH in this contest. In LPIRC2016, we implemented a pipelined energy-aware object detection system on NVDIA Jetson TX1 to reach an improved tradeoff between energy and accuracy.

In the on-site contest of LPIRC2016, two different object detection architectures: BING+FAST-RCNN and Faster-RCNN are employed to complete the image recognition missions of track-1 and track-3 respectively. These two frameworks are all based on the popular convolutional neural network and implemented on the NVIDIA Jetson-TX1 development board. The design philosophy in building this low-power but cost-effective object detection system is quite simple: to trade-off between model accuracy and computational complexity, and to maximize the utility of the heterogeneous CPU+GPU SoC, through an exhaustive design space exploration stage of CNN parameters. Design parameter exploration method is essential to increase the energy-efficiency of CNN-based object-detection solutions on mobile systems. For example, we can adjust the hyper-parameters of the used CNN model, sacrifice accuracy for speed-up by means of dimension reduction techniques and other input-level approximation methods [4]. It helps create a massive space of design parameters for the target CNN-based object detection framework. With a sufficiently large design space, we can evaluate the accuracy and power consumption of different implementations, and search for a proper design point that yields the highest score measured in mAP/WH. With this method, we successfully boost the energy-efficiency of the Fast R-CNN and Faster R-CNN implementations on NVIDIA TX1, which are respectively the winners in track-1 and track-3 of LPIRC2016.

The past year has witnessed the rapid advancement in the area of CNN-based object detection and also energy-efficient architectures that are believed to deliver higher inference performance for these CNN-based frameworks. New detection frameworks and more powerful mobile devices like NVIDIA TX2 and Movidius stick are bringing more design options for power-efficient object detector implementation. In this paper, we not only introduce the pipelined energy-aware object detection frameworks for LPIRC2016, but also investigate the implications of technological progress on the energy-efficiency of low-power object detectors and how it influences the tradeoff between detection energy and accuracy of LPIRC solutions.

The rest of the paper is organized as follows. We introduce the background of low power object detection methods in Section II, and in Section III present the CNN design parameter exploration method for energy-efficiency boost in the track-1 contest of LPIRC. Section IV shows how it performs in track-3 of LPIRC. Section V concludes this paper.

II. Real-Time CNN-based Object Detection Method

A. Background and preliminaries

Object detection aims to hypothesize the object location and categorization from images or videos. An accurate object detector is expected to generate the precise bounding boxes that cover the region of the detected objects (for example, faces, or vehicles) and produce the according categorization labels. Low Power Image Recognition Challenge (LPIRC) also uses the dataset from ImageNet Large Scale Visual Recognition Challenge (ILSVRC) to test the efficiency of participants' solution [5]. However, compared to ILSVRC, LPIRC not only measures the detection accuracy (mean average precision, *mAP*) of contest solutions, but also considers the important factor of energy consumption in on-site evaluation. The final score of the detection solution under evaluation will be calculated by an on-site referee system, as illustrated in Eq. 1, where mAP gauges the detection accuracy and E (measured in watt · hour) is the total energy consumption measured during the 10-minute test.

$$\text{Score} = \frac{\text{mAP}}{\text{E}} \quad (1)$$

B. Two-Stage CNN-based object detection

Deep convolutional neural networks (CNNs) have been showing prevailing accuracy in object detection tasks compared to traditional methods. Early region-based object detection systems include two stages: hypothesizing bounding boxes, extracting features of each box and applying a classifier to them. The first stage often relies on inexpensive coarse-grained proposal search methods, such as Selective Search, EdgeBox, BING, etc. [6] [7]. With the region proposal results, the CNN mainly plays as a classifier, and it does not need to predict the object bound coordinates in the image all by itself. These flows are defined as two-stage object detection methods. The original Region CNN (R-CNN) is a typical two-stage object detector and very computationally expensive because it has to perform a ConvNet forward pass for every single object proposal [8]. To accelerate this procedure, later work such as OverFeat computes the shared convolutional features from an image pyramid for detection results [9]. Similarly, SPPnets were proposed to accelerate the R-CNN by sharing some convolutional feature maps across the regions [10]. Fast R-CNN [2] also trains the detector to work on shared convolutional features, and it even claims to classify object proposals and refine their bounding-box coordinates simultaneously. In general, the two-stage CNN-based object detectors are thought as expensive and less efficient because it requires additional computation stage of bounding-box search, however, they can be used to construct power-efficient object detection solutions by properly balancing the two stages in heterogeneous multi-core SoC when the accuracy is not the sole optimization target of the system [11].

C. Single-Stage CNN-based object detection

The region proposal steps in two-stage methods consume considerable execution time and possibly becomes the performance bottleneck when it runs in a relatively less-powerful CPU core. Faster R-CNN firstly introduces novel Region Proposal Networks (RPNs) that replace the region proposal predictors, and it stacks a few additional convolutional layers on the shared convolutional feature maps to reduce computational overhead [1]. Compared to Region-based CNN approaches, state-of-the-art single stage object detectors like YOLO and SSD provide cost-effective alternatives to them for real-time processing purpose [12] [13]. These single-stage object detection methods often treat the stage of proposal search as a classical regression problem and merge it into deep convolutional neural network. YOLO is such a framework that uses features from the entire image to predict each bounding box. It also predicts all bounding boxes across all classes for one frame at high speed. To improve the location accuracy, SSD combines the notion of fixed-size prior boxes from Faster R-CNN and multi-scale features in proposal search, but it removes the stages of feature resampling in proposal boxes to achieve speed improvement. YOLOv2 also adopts the concept of prior anchor boxes to improve the recognition accuracy [14]. As the most recent region-based detection method, R-FCN adopts the fully convolutional image classifier and the position-sensitivity score maps to incorporate translation invariance in both classification and detection [2]. Later single-stage object detection methods keep improving the detection accuracy or develop towards the direction of prediction speed-up [15] [16]. In brief, single-stage approaches are becoming the mainstream technical solution to object detection for the conciseness of end-to-end prediction and greater space for performance improvement. However, when we were devising our contest solution for LPIRC2016, these solutions are not widely adopted or even under development.

D. Object Detection Acceleration on mobile systems

Power-efficient object detection must be fast enough to reduce the overall energy consumption spent during the image processing stage. However, state-of-the-art object schemes are generally computationally expensive due to the deep existence of convolutional layers. Therefore, applying lightweight convolutional neural networks to the object detection framework is expected to improve the detection efficiency. Such attempts are primarily classified into three categories. Firstly, using mobile CNN models such as Squeezenet, Mobilenet and Shufflenet, instead of resorting to the classic deep networks with numerous parameters and intensive operations, can directly improve the memory access performance and eliminate the majority of MAC operations in deep feature extraction. Such lightweight models often have compact feature maps and shallow network structure, enabling real-time network inference on low-power devices [17] [18] [19]. The second type of optimization technique is to reduce the parameter size and the involved computation operations of CNN models through compression or quantization technologies. For example, sparse matrix encoding and product quantization can be leveraged to reduce the footprint of CNN parameters, which promotes the memory and bandwidth utilization of mobile CPU or GPGPU [20] [23]. Among them, sparsification is a common approach used to eliminate ineffectual data and operations in a network by pruning its connections at training stage. Nguyen et al. proposed LSDN+ShrinkConnect to reduce the number of network activations for domain-specific working-set compression and acceleration [21]. Deep compression [22] further reduces the weight storage overhead to run large networks by means of pruning, quantization and Huffman coding. Also, matrix Low Rank Approximation and Singular-Value Decompostion techniques are also falling into this category by reducing the dimension of matrix multiplications involved in CNN inference [23]. Other approaches like *winograd fft* are also proved to be useful in CNN inference acceleration [24]. Some of these techniques have also been adopted in the primitive library of popular deep learning frameworks, and some of them are evaluated as a design option in our prior solution to the LPIRC2016 contest. Except software level techniques, employing specialized hardware implemented with ASICs and FPGA to improve the performance and efficiency of CNN inference is also a promising and intensively studied area related to fast object detection [25] [26] [27].

III. Speed-Accuracy Tradeoff for Energy-Efficient batching Object detection

The evaluation metric for track-1 of 2nd LPIRC is mAP/WH, thus the system design must pursue a balanced tradeoff between accuracy and energy consumption to maximize the score. To resolve the maximization problem, we have to find sufficient adjustable design parameters in our detection framework to obtain the optimal solution.

A. Design Space Exploration for pipelined Fast R-CNN

Our previous champion solution of LPIRC employs Fast R-CNN as the object detection framework to deal with the on-site image recognition task. As we have mentioned, it aimed to boost the energy-efficiency of the detection solution through an extensive exploration of design parameters in this framework. There are many design choices in the implementation of Fast R-CNN in the target hardware NVIDIA Jetson TX1 SoC, including the hyper-parameter of CNN, data representation precision, region proposal methods and input scaling factor. The design parameter exploration phase is to achieve two essential targets:

1. To make sure that the marginal profit of tuning each parameter knob is equivalent to the others', which means changing any of the parameters will decrease the contest score measured in mAP/MH. In ideally balanced cases, neither the accuracy nor the power consumption is over-emphasized, so that the mAP/WH will be maximized.

2. To make a full utility of CPU computational resources and GPU cores, which means the pipelined stages of proposal search and CNN inference are completely balanced in computation time when they are assigned to the CPU and GPU cores respectively. In this case, neither the CPU nor the CPU core will be forced to wait for the response from the other side, so that no static power will be wasted on core idling.

With this target and constraint, we decided to exploit some of the design parameters from both software and hardware perspectives:

- *Bounding-box Searcher* Fast R-CNN need a region proposal search method to pre-select sufficient bounding boxes for feature extraction. Such methods as Selective Search, EdgeBox, BING, have different impacts on the recognition speed and accuracy. In general, Selective Search and EdgeBox are categorized as "slow-but-accurate" solutions, while BING is considered as "fast but less accurate" method. When the general goal of energy-efficiency is prioritized over the other single-object metric, BING is expected to contribute to more speed-up and energy-efficiency gains. Besides the multiple options of region proposal extractor, the number of output box and the input image size are also two important reconfigurable parameters in the pipeline of object detection. For example, when we directly crop the image to half size, the speed of BING is almost increased more than 100 percent. Moreover, if BING is configured to generate the top K scored proposals for further usage in Fast R-CNN instead of the top-scored 1000 boxes (K<<1000), the latency of the detection pipeline will also be shrunk significantly at a minor cost of accuracy loss. In scenarios when the application is more sensitive to speed, it is to be avoided to pass too many of proposals to the stage of region-based CNN classification. Last of all, Faster R-CNN can also be accelerated by reducing the proposals. Conclusively, the option and the detailed parameters of region proposal methods can lead to many tuning knobs that could be adjusted to reach tradeoff between accuracy and speed.
- *CNN model selection* Different CNNs models can be fitted into the framework of Fast R-CNN for the step of feature extraction, classification and box regression, after proposal search. A deep convolutional neural network have many tunable hyper-parameters used to configure the strength and computational cost of the network even

for the same task [28]. Changing such hyper-parameters has a direct impact on the memory access and computation performance of object detection on the SoC. In this work, we assume the hyper-parameter of CNNs, *i.e.,* depth, kernel size, activation functions and other features, as adjustable parameters in the stage of design space exploration for energy-efficient object detector on TX1. The choice of replacing the feature extracting CNNs in Fast R-CNN into a possible customized network can also create an ample space of design points with varied detection accuracy and detection speed.

- *Input-level Approximation* Cropping the under-test image into different sizes also directly changes the number of operations in the entire procedure of CNN inference for object detection. The speed-up effect brought by image down-sizing is evident because the volume of convolutional operations drops almost at the same ratio as the resizing factor. We see in experiments that when image size changes from 600*600 to 300*300, the inference speed almost increase 3× and the detection accuracy of the original Fast R-CNN decrease only about 3 percent.
- *Batch Sizing* GPU is a high-throughput architecture that supports the exploitation of batch-level parallelism in object detection. However, the best batch size for high throughput processing is depending on the scale of CNN model and also the memory capacity of the computing platform. Thus, we can try different batch sizes for a potential detection framework and find a proper one that leads to better performance on the GPU.
- *Representation Precision* GPGPUs like TX1 support half-precision operation to accelerate matrix multiplication. The computation throughput can be elevated at a minor cost of accuracy loss when full-precision operation is replaced with half-precision operation [11]. The recently released *TensorRT* framework even support the execution mode of 8-bit fixed point representation, which could potentially improve the memory performance without the need to re-train the model. This mechanism also poses an option of speedup and accuracy tradeoff.
- *Truncated singular value decomposition* The convolutional, fully-connected and RoI layers that dominate the inference time of the detector [22], are basically decomposed into matrix multiplication in GPGPU execution. Particularly, for Fast R-CNN, localized RoI layers account for nearly half of the inference time when the proposal number and the number of object class are huge. Singular value decomposition (SVD) is a linear-algebra technique that factorizes a matrix and eventually compresses it for less intensive computation on GPUs. Supposing the linear operation in an FC layer is deemed as a product of Wx, x is the feature vector and W is the matrix of weight. SVD is to decompose the W to let W=UDV, where D is the diagonal matrix with singular value on the diagonal, and U and V are orthogonal matrices. To reduce the size of W, W can be approximated by keeping only the d largest elements in D diagonals, so that the approximated W becomes the product of smaller matrices U, D and V. Such truncated SVD is proved to effectively reduce the scale of FC and convolutional weight, also with a controllable amount of accuracy loss. Depending on the proposal number, truncated SVD is proved to reduce detection time by 30%-70% with only a small (0.3 percentage point) drop in *mAP* and without additional fine-tuning after model compression [11].

The adjustable parameters listed above do not cover all the acceleration techniques for CNN-based detection methods, but they are chosen as the basic tuning knobs in our design exploration phase for LPIRC-score maximization, for their simplicity and generality. Fig. 1 shows the varying detection accuracy and speed when some of the parameters are adjusted. It reveals that a larger exploration space exists for speed-accuracy tradeoff. In this figure, the model size of CNN1, CNN2 and CNN3 are decreasing while the images under test are cropped into 224*224, 300*300 and 600*600 inputs.

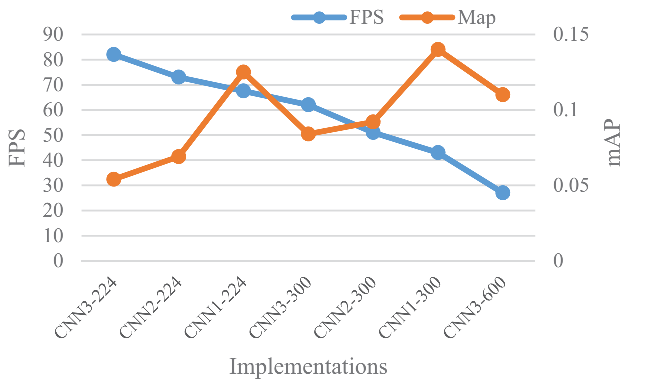

Fig. 1 Accuracy and speed of several implementations

B. *The result of the LPIRC2016 solution*

Since the evaluation metric of track-1 is to be maximized, we can describe this task as an optimization problem whose objective is maximizing mAP/WH. To reach this goal, we have multiple design parameters to explore, such as *p:* proposal number, *s:* size of input image, *b:* batch size, *d:* truncated *d* of SVD and H: hyper-parameter of network. Suppose the energy-efficiency (mAP/WH) is a function of these factors, $f(p,s,b,d,\boldsymbol{H})$. What we are going to search for in this design space is a group of parameters leading to the maximum mAP/WH: $\arg\max\big(f(p,s,b,d,\boldsymbol{H})\big)$.

$$\textit{Obj: } \max\big(f(p,s,b,d,\boldsymbol{H})\big)$$
$$\textit{s.t.}: \quad \mathrm{T}(BING)=\mathrm{T}(RCNN)=T_{pre} \tag{2}$$

In addition, the constraint of this problem is that the speed of BING on the ARM cores and the down-stream Fast R-CNN on the GPGPU are balanced, equal to the image fetching and decomposition time T_{pre}, so that the detection task can be parallelized and the resources/energy on TX1 are fully utilized.

We tried two networks, Fast R-CNN(C) and Fast R-CNN(P), and finalized the other parameters: p is 50, s is 224, b is 50 and *d* is 256. The results of track-1 are shown in Table I. In the competition, the Fast R-CNN (P) solution was ranked the 1st in all nine solutions, which has minor differences from the original Alexnet.

TABLE I
Results in track-1 of 2nd LPIRC with TX1 [11]

Solution	Accuracy	Energy(WH)	Score
Fast R-CNN(C)	0.0347	0.79	0.044
Fast R-CNN(P)	0.0260	0.94	0.028

C. Hardware Upgrade

Mobile GPGPU The evaluation of LPIRC2016 solutions are all performed on NVIDA Jetson TX1, a Maxwell architecture GPGPU with 256 CUDA cores delivering over 1 TeraFLOPs of performance in theory. The newer version of Jetson board, based on TX2, has been released in this year. TX2 SoC has a Pascal GPU, two big and four small ARM cores, and 8GB DDR4 DRAM on board. Its throughput has been significantly improved over TX1. To evaluate the implications of hardware upgrade, we migrate the implementation of Fast R-CNN to TX2, and select a proper combination of parameters through the design exploration flow as introduced in previous chapters. The performance and efficiency improvement can be seen in Table II.

TABLE II
Results of Fast R-CNN with TX2

Solution	Accuracy	Energy(WH)	Score
Fast R-CNN	0.0392	0.601	0.065

Specialized deep learning accelerator As the recent two years has witnessed the popularity of research and study on deep learning accelerators or processor; we also decided to evaluate their performance and efficiency on CNN-based object detection. However, the Movidius Neural Compute Stick powered by Vision Processing Unit (VPU) is the only device we can find from the market with an available AI programming interface. At the moment, our attempts to run detection frameworks such as Fast R-CNN, SSD and Yolov2, are unsuccessful. Yolov1 is the only framework we can successfully run on the Movidius stick, so that we measured its performance and power consumption through the USB interface approximately. The energy consumption number may not be fair for comparison to the GPGPU implementations because it excludes the power proportion spent by the host desktop computer the Movidius stick is plugged into. However, we can see from the Table III that the approximate score of Movidius is very close to the result of Fast R-CNN on TX1 board. Please notice that the energy cost and the score are for reference because the power spent on image transfer through Ethernet is not included in Movidius stick.

TABLE III
Results of Yolov1 with Movidius

Solution	Accuracy	Energy(WH)	Score
Yolov1-tiny	0.0076	~0.191	~0.0395
Yolov1-resnet	0.0038	~0.188	0.0202

IV. Single-frame Object detection

Track-3 of LPIRC requires the solution to use a camera to capture the screen and detect the objects in the captured image. After detection, the screen is refreshed immediately once a refresh request is sent from the detector to the on-site referee server.

The biggest difference between track-1 and track-3 is that the detection machine will not benefit from a high-throughput chip that could exploit batch-level parallelism in object detection. The reason is due to the limited sampling rate of the camera and also the refresh rate of the monitor. Because the CPU captures the images at a slow frame rate, the detection system has to process them frame by frame instead of waiting for them to become a large batch. Thus, the optimization problem described in formula-(1) has a different constraint in contrast to track-1. Because the constraint changed, we put more emphasis on accuracy instead of on speed and select Faster R-CNN as the framework. In the contest, we replace the ZFnet with CaffeNet in the framework of Faster-RCNN to achieve load balance between the CPU and GPU cores. The final score of track-3 is shown in Table IV.

TABLE IV
Results for track3 tasks [11]

Solution	Accuracy	Energy(WH)	Score	Hardware
Faster R-CNN(C)	0.00251	1.78	0.0014	TX1
Yolov1-tiny	0.00181	~0.198	~0.0091	Movidius
Yolov1-resnet	0.00114	~0.193	~0.0059	Movidius

Because the constraint of the score-maximizing problem has been changed, we also found that the solution based on specialized deep learning accelerator, i.e., Movidius stick, are showing much better performance than previously in track-1, if we disregard the power spent on camera and Ethernet communication. The reason is because the end-to-end image recognition time on Movidius for each frame is already very close to the speed of per frame inference on GPU. When batching execution is not available, the energy-efficiency of embedded deep learning accelerator is increased rapidly. Considering that the Movidius stick consumes less than 20% of TX1 power, the energy-efficiency of per-frame image recognition is even higher in the set-up of track-3 competition. Compared to track-1 that highlights the aggregated throughput of image recognition, the mechanism of track-3 is believed to favor low-latency and low-power object detection solution according to the results. This is an interesting observation worth contemplating. However, the latest LPIRC has removed track-3 from the contest.

V. Convlusion

This paper introduces how we finalized the energy-aware object detection solution for 2nd LPIRC in 2016, and also take a retrospective examination of the design exploration method employed to reach accuracy-speed tradeoff in low power object detection. Newer hardware like NVIDIA TX2 and Intel Movidius stick, and detection frameworks are evaluated and compared in the context of LPIRC contest, including both track-1 and track-3 tasks, showing the implications of recent technological advancement on low power object detection systems.

ACKNOWLEDGEMENT

This work was supported in part by National Natural Science Foundation of China under Grant No. 61432017, 61504153, 61532017, 61402146 and 61521092, and in part by National Key Research and Development Program of China under Grant 2016YFF0203500. The corresponding authors are Huawei Li and Xiaowei Li.

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” MIT Press Annual Conference on Neural Information Processing Systems (NIPS), 2015.

[2] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: Objectdetectionvia region-based fully convolutional networks,” MIT Press Annual Conference on Neural Information Processing Systems (NIPS), 2016

[3] Y. H. Lu et al., “Rebooting Computing and Low-Power Image Recognition Challenge,” IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Austin, TX, 2015, pp. 927-932.

[4] T. Chakradhar and A. Raghunathan, “Best-effort computing: re-thinking parallel software and hardware,” Proceeding of ACM Design Automation Conference (DAC), 2010.

[5] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. MIT Press Annual Conference on Neural Information Processing Systems (NIPS), 2012.

[6] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, “Selective search for object recognition,” Springer International Journal of Computer Vision (IJCV), 2013.

[7] M. Cheng, Z. Zhang, W. Lin, P. Torr, “BING: Binarized Normed Gradients for Objectness Estimation at 300fps,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014

[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” International Conference on Learning Representations (ICLR), 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” Springer European Conference on Computer Vision (ECCV), 2014.

[11] C. Wang, Y. Wang, Y. Han, L. Song, Z. Quan, J. Li, X. Li, “CNN-based object detection solutions for embedded heterogeneous multicore SoCs,” in Proc. Of ASP-DAC, 2017.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” Proceeding of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, “SSD: Single Shot MultiBox Detector,” IEEE European Conference on Computer Vision (ECCV), 2016.

[14] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” Proceeding of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017

[15] T. Y. Lin, P.Goyal, R.Girshick, K.He, and P.Dolla ́r, “Focal loss for dense object detection,” arXiv preprint arXiv:1708.02002, 2017.

[16] K.H. Kim, S. Hong, B. Roh, Y. Cheon, and M. Park, “Pvanet: Deep but lightweight neural networks for real-time object detection,” arXiv preprint arXiv:1608.08021, 2016.

[17] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size,”, arXiv:1602.07360, 2016.

[18] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” arXiv preprint arXiv:1707.01083, 2017.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” arXiv preprint arXiv:1704.04861, 2017.

[20] W. Wen, C. Wu, Y. Wang, Y. Chen, H. Li, “Learning Structured Sparsity in Deep Neural Networks,” MIT Press Annual Conference on Neural Information Processing Systems (NIPS), 2016.

[21] V. Nguyen, Hien, K. Zhou, and R. Vemulapalli, “Cross-domain synthesis of medical images using efficient location-sensitive deep network,” Medical Image Computing and Computer-Assisted Intervention (MICCAI), 2015.

[22] S. Han, H. Mao, and W. J. Dally, “Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding,” International Conference on Learning Representations (ICLR), 2016.

[23] Y. Gong, L. Liu, M. Yang, and L. Bourdev, “Compressing deep convolutional networks using vector quantization,” arXiv preprint arXiv:1412.6115, 2014.

[24] H. F. Silverman, “An introduction to programming the Winograd Fourier transform algorithm,” IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-25, no. 2, pp. 152–164, Apr. 1977.

[25] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, “Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning,” ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2014.

[26] Y. Wang, J. Xu, Y. Han, H. Li, and X. Li, “DeepBurning: automatic generation of FPGA-based learning accelerators for the neural network family,” ACM Design Automation Conference (DAC), 2016.

[27] Y. H. Chen, T. Krishna, J. Emer, V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks", IEEE ISSCC, 2016.

[28] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, “Return of the devil in the details: Delving deep into convolutional nets,” British Machine Vision Conference (BMVC), 2014.