DAC-SDC Low Power Object Detection Challenge for UAV Applications

Xiaowei Xu, *Member, IEEE*, Xinyi Zhang, *Student Member, IEEE*, Bei Yu, *Senior Member, IEEE*, Xiaobo Sharon Hu, *Fellow, IEEE*, Christopher Rowen, *Fellow, IEEE*, Jingtong Hu, *Member, IEEE*, and Yiyu Shi, *Senior Member, IEEE*

Abstract—The 55th Design Automation Conference (DAC) held its first System Design Contest (SDC) in 2018. SDC'18 features a lower power object detection challenge (LPODC) on designing and implementing novel algorithms based object detection in images taken from unmanned aerial vehicles (UAV). The dataset includes 95 categories and 150k images, and the hardware platforms include Nvidia's TX2 and Xilinx's PYNQ Z1. DAC-SDC'18 attracted more than 110 entries from 12 countries. This paper presents in detail the dataset and evaluation procedure. It further discusses the methods developed by some of the entries as well as representative results. The paper concludes with directions for future improvements.

Index Terms—Dataset, benchmark, object detection, unmanned aerial vehicles, low power

1 Introduction

The 55th Design Automation Conference (DAC) held its first System Design Contest (SDC) in 2018 which features a lower power object detection challenge (LPODC) on designing and implementing novel algorithms based object detection in images taken from unmanned aerial vehicles (UAV). This challenge provides a unified platform to develop and compare state-of-the-art object detection algorithms, and discusses the lessons learned from these participated entries.

The LPODC at DAC-SDC'18 focuses on unmanned aerial vehicles applications as such applications have stringent accuracy, real-time, and energy requirements [29]. Specifically, first, the LPODC task is to detect a single object of interest, one of the most important tasks in UAV applications [25]. Second, different from general computer visual challenges, such as ImageNet [22] and PASCAL VOC dataset [9], which focused only on accuracy, LPODC evaluates the final performance based on a combination of throughput, power, and detection accuracy. Thus, LPODC takes into full consideration the features of UAV applications: real-time processing, energy-constrained embedded platform, and detection accuracy. Third, the images of the dataset are all captured from actual UAVs which reflect the real circumstances and problems of UAV applications. Fourth, LPODC provides two hardware platforms: embedded GPU (Jetson TX2 from Nvidia [5]) and FPGA SoC (PYNQ Z-1 board from Xilinx [7]) to all the participating teams to choose from for their implementations. Note that GPUs and FPGAs are widely adopted for energy-efficient processing on UAVs [8].

The publically released dataset contains a large quantity of manually annotated training images, while the testing dataset is withheld for the evaluation purpose. There are a total of 150k images provided by a UAV company DJI [4]. Participating teams trained their models/algorithms with the training dataset, and sent the trained models/algorithms to the organizers to get the final testing results including throughput, energy, and detection accuracy. Such evaluation was performed at the end of each month and the detailed rank was released then. The final rank was released at the end of the competition and the top-3 entries from both GPU and FPGA categories were invited to present their work at a technical session at DAC.

This paper describes the LPODC in detail including: the task, the evaluation method and the dataset. Furthermore, a comprehensive discussion of the methods and results of the top-3 entries from both GPU and FPGA categories is presented to provide insights and rich lessons for future development of object detection algorithms especially for UAV applications. Particularly, we will elaborate hardware-software co-design for efficient processing on embedded platforms.

The training dataset, the source codes of the top-3 entries of both GPU and FPGA categories, and additional information about this challenge can be found at www.github.com/xyzxinyizhang/2018-DAC-System-Design-Contest.

1.1 Related Work

In this section we briefly discuss the related work about benchmark image datasets for object detection. As segmentation

- *X. Xu, X. Sharon Hu, and Y. Shi are with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556 USA. E-mail: {xxu8, shu, yshi4}@nd.edu.*
- *B. Yu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong. E-mail: byu@cse.cuhk.edu.hk.*
- *X. Zhang and J. Hu are with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA 15261 USA. E-mail: {xinyizhang, jthu}@pitt.edu.*
- *C. Rowen is with the Cognite Venture, 210 Mission Street, Santa Cruz, CA 95060 USA. E-mail: rowen@cogniteventures.com.*

Manuscript received 31 Aug. 2018; revised 10 July 2019; accepted 24 July 2019. Date of publication 5 Aug. 2019; date of current version 7 Jan. 2021. (Corresponding author: Xiaowei Xu.)
Recommended for acceptance by B. Morse.
Digital Object Identifier no. 10.1109/TPAMI.2019.2932429

datasets can also be used for object detection, some widely-used segmentation datasets are also included.

Most of the datasets for object detection contain common photographs. LabelMe [23] has 187k images each of which contains multiple objects annotated with bounding polygon. It also provides a web-based online annotation tool for easy contribution by the public. Like LabelMe, PASCAL VOC dataset [9] further increases the number of images to 500k. ImageNet [22] is one of the most popular datasets in computer vision community, and it contains more than 14 million images. It was primarily for classification in 2010 and extended to support object detection and scene recognition in 2013. Compared with ImageNet, SUN database [28] mainly focuses on scene recognition and contains about 131k images. Microsoft Common Objects in Context (COCO) [18] contains complex everyday scenes of common objects in their natural context and has 2.5 million images. Open Images [6] contains more than 9 million real-life images within 6,000 categories which is much larger than that of ImageNet (about 1,000).

There are also some datasets for specific applications, and the images are taken from particular views. KITTI vision benchmark dataset [12] is specific for autonomous driving, and the images are taken from an autonomous driving platform in a mid-size city. FieldSAFE [16] is for agriculture application and has approximately 2 hours of raw sensor data from a tractor-mounted sensor system in a grass mowing scenario. DOTA [27] focuses on aerial applications, and all the images are captured from cameras on aircrafts.

The dataset in this paper is specific for UAV applications. The images in the dataset are taken from UAVs which operates on a much lower height than general aircrafts in DOTA. The associated environment in the images is also much more complex than that in DOTA.

The most related work to our challenge is the Low-Power Image Recognition Challenge (LPIRC) [10], [11], [20], which considers accuracy, speed, and power consumption for efficient object detection and classification on different platforms. LPIRC adopts a popular dataset ImageNet for general classification and detection, while LPODC is specific for UAV applications which is a more realistic scenario of computer vision on embedded systems. In addition, the tasks of LPODC and LPIRC are different. In LPIRC, the detection task is to detect all the related objects in the image, and usually the number of objects can be more than five. The task of LPODC, on the other hand, is to detect only one object in the image which is also specific for UAV scenarios. Compared with LPIRC, our UAV-specific challenge of one object detection is even more challenging. Fox example, our task need to detect the object from many similar objects, and some object is with very small size (less then 20 pixels) or within very dim environment. More details of the dataset is discussed in Section 3. Furthermore, our challenge has more specific hardware constraint. For example, LPIRC in 2018 has two tracks: a GPU platform and platforms without hardware and software restrictions, while in other years LPIRC has no hardware or software restrictions. LPODC constrains to two specific GPU and FPGA platforms. To accommodate the differences between the compute capabilities of the two platforms, we have designed different scoring methods for GPU and FPGA platforms, and more weight is given to accuracy on the GPU platform as it is more powerful.

1.2 Paper Layout

We have given an overview of the LPODC at DAC'18. The rest of the paper is organized as follows. In Section 2, the challenge task and its evaluation method, as well as the provided two hardware platforms are described. The details of the dataset and its analysis are given in Section 3. The analysis and discussion of the methods for object detection of GPU and FPGA entries are presented in Section 4. Section 5 details the results of the GPU and FPGA entries. We conclude the paper with some discussions of the challenge and possible improvements.

2 LPODC Task and Evaluation Criteria

In this section, the details of the challenge task are presented, followed with an introduction of the hardware platforms and the evaluation method.

2.1 Object Detection

The LPODC task is to perform single object detection in each image with an axis-aligned bounding box indicating the object's position and scale. As the challenge is targeted at UAV applications, there are several aspects that need to be emphasized. First, the object detection task is to locate a specific object from the training dataset, rather than objects from a training category. For example, if images containing person A are in the training dataset, then the task is to detect person A rather than other persons. More details about the detection objects are discussed in Section 3. Second, the object detection task needs to be executed with high throughput and high accuracy which are required by UAV applications. This requirement is achieved through the weighting of throughput in the scoring system, discussed in Section 2.3.

2.2 Hardware Platforms

In this challenge, two hardware platforms, either FPGA or GPU, were provided to the participating teams from the challenge sponsors Xilinx [7] and Nvidia [5], respectively. Particularly, the *FPGA platform* is Xilinx PYNQ Z-1 board which is an embedded system based platform combining Zynq system and Python [15]. Participants are allowed to use Cortex-A9 processor and ZYNQ XC7Z020-1CLG400C on the platform to realize their solutions to the challenge. The embedded FPGA chip contains 53K 6-input look-up-tables, 220 DSPs, and 630 KB fast block RAM. A 512 MB DDR3 memory with 16-bit bus at 1,050 Mbps is also embedded on the platform which can be accessed by both the processor and the FPGA. The power consumption of the FPGA platform is about 1-4 watts.

The *GPU platform* is Nvidia Jetson TX2, which is an embedded AI computing device. This GPU is very powerful with a 6-core CPU (Dual-core NVIDIA Denver2 + quad-core ARM Cortex-A57), a 256-core Pascal GPU, and 8 GB LPDDR4 DRAM. It can provide more than 1TFLOPS of FP16 compute performance in less than 7.5 watts of power. Note that both hardware platforms target low-power embedded computation and are suitable for UAV applications.

2.3 Evaluation Method

The evaluation for the challenge is based on detection accuracy, throughput, and energy consumption. For simplicity, test images are stored in the on-board memory of the GPU platform or a Secure Digital (SD) card of the FPGA platform and fed to the object detection algorithm.

The metric for object detection accuracy is Intersection over Union (IoU). Suppose there are two bounding boxes (BB): a predicted BB and the ground-truth BB. Then the accuracy or IoU of the predicted BB is the ratio between the area of the union of the predicted BB and the ground-truth BB and the area of the overlap encompassed by both the predicted BB and the ground-truth BB, i.e.,

$$IoU = \frac{BB_p \cap BB_g}{BB_p \cup BB_g}, \tag{1}$$

where BB_p and BB_g are the areas of the predicted and ground-truth BBs, respectively. Note that the challenge only cares the IoU results, but does not care the object categories.

The metric for throughput is frames per second (FPS). For real-time processing in UAV applications, the minimum throughput requirement in this challenge was set to 20 FPS on the GPU platform and 5 FPS on the FPGA platform. If the FPS is lower than the requirement, then a penalty to IoU is added, i.e.,

$$IoU_r = IoU_m \times (min(FPS_m, FPS_r))/FPS_r, \tag{2}$$

where IoU_r and IoU_m are the actual and measured IoUs, respectively, and FPS_r and FPS_m are the required and measured FPSs, respectively. The min function outputs the minimum one of its inputs.

Energy consumption is the energy consumed in the whole evaluation. For the GPU platform, the realtime power consumption is measured using the on-board power monitors [1] and recorded using a bash script. For the FPGA platform, a power meter is used, and power consumption is sampled with a frequency of 0.01 Hz. Finally the energy consumption is obtained by multiplying the power consumption with the runtime.

The final score is a combination of accuracy, throughput, and energy consumption. Suppose there are I registered entries and the dataset contains K evaluation images. Let $IoU_{i,k}$ be the IoU score of image k ($k \leq K$) for entry i ($i \leq I$). Then the IoU score R_{IoU_i} for entry i is computed as

$$R_{IoU_i} = \frac{\sum_{k=1}^{K} IoU_{i,k}}{K}. \tag{3}$$

Let E_i be the energy consumption of processing all K images for entry i. Let $\bar{E}_I$ be the average energy consumption of the I entries. $\bar{E}_I$ is computed as

$$\bar{E}_I = \frac{\sum_{i=1}^{I} E_i}{I}. \tag{4}$$

Then the energy consumption score ES_i for entry i is

$$ES_i = max\{0,\ 1 + 0.2 \times log_x \frac{\bar{E}_I}{E_i}\}, \tag{5}$$

where x is set to 2 and 10 for FPGA category and GPU category, respectively. Through profiling, we find that the energy-performance Pareto frontier of the GPU platform exhibits much smaller gradient than that of the FPGA platform, reflecting the fact that the FPGA is more resource constrained than the GPU and thus its performance is much more sensitive to energy consumption. As such, we stress more on the accuracy for the GPU, while more on the energy for the FPGA. The final total score TS_i for entry i is

$$TS_i = R_{IoU_i} \times (1 + ES_i). \tag{6}$$

The factor 0.2 in Equation (5) is set based on the estimated range of energy consumption variation in participating teams. Through our profiling of the platform, we anticipate that a team will normally have an energy consumption from 1/4x to 4x the average of all teams for both GPU and FPGA categories. As such, $log_x \bar{E}_I/E_i$ will take a range of -2 to 2, and a factor of 0.2 will make ES_i in the range of 0.6 to 1.4. Thus, the final (1+ES_i) in Equation (6) is in the range of 1.6 to 2.4 which is appropriate to act as a reward factor for energy efficiency. The max function and the addition of 1 to ES_i are for teams with extremely low performance on energy efficiency. In such condition, $log_x \bar{E}_I/E_i$ is a large negative number, and the max function ensures that ES_i is not a negative number but zero. Then the addition of 1 to $ES_i = 0$ further ensures that TS_i can still be graded based on accuracy with no multiplying rewarding factor for energy efficiency rather than being frozen to zero even if very high IoUs are obtained.

3 Dataset

As shown in Fig. 1, the adopted dataset from DJI [4] contains 12 categories of images and 95 sub-categories. For each sub-category, 70 percent of the images are provided for training and 30 percent are reserved for evaluation. It should be highlighted that compared with existing general purpose datasets such as ImageNet [22] and PASCAL VOC dataset [9], the object is captured in a UAV view and with different points of view.

The distributions of the training and testing datasets with respect to category, object size ratio, image brightness and amount of information are shown in Figs. 2, 3, 4, and 5. Here, object size ratio is the ratio of the object size to the image size. The brightness of a pixel is defined as in Equation (7) [24] (r, g, b are the three channels of images), where the image brightness is the average brightness of all its pixels.The amount of information is the amount of textures or edges in the area of objects in the image, and it is defined as the average pixel entropy of the object where the pixel entropy is calculated in a moderate 5×5 region [26].

$$brightness = \sqrt{0.241 \times r^2 + 0.691 \times g^2 + 0.068 \times b^2}. \tag{7}$$

Fig. 2 shows that the ratio of quantity of training and testing images in different categories are almost the same, which is manually segmented to achieve a good balance of training and testing dataset. The categories *person*, *car*, and *rider* contain much more images than others as they contain more subcategories than others. It can be noticed that there is also a good balance between the training dataset and the testing dataset for different object sizes except for some large object size ratio as shown in Fig. 3. As the object size ratio increases,

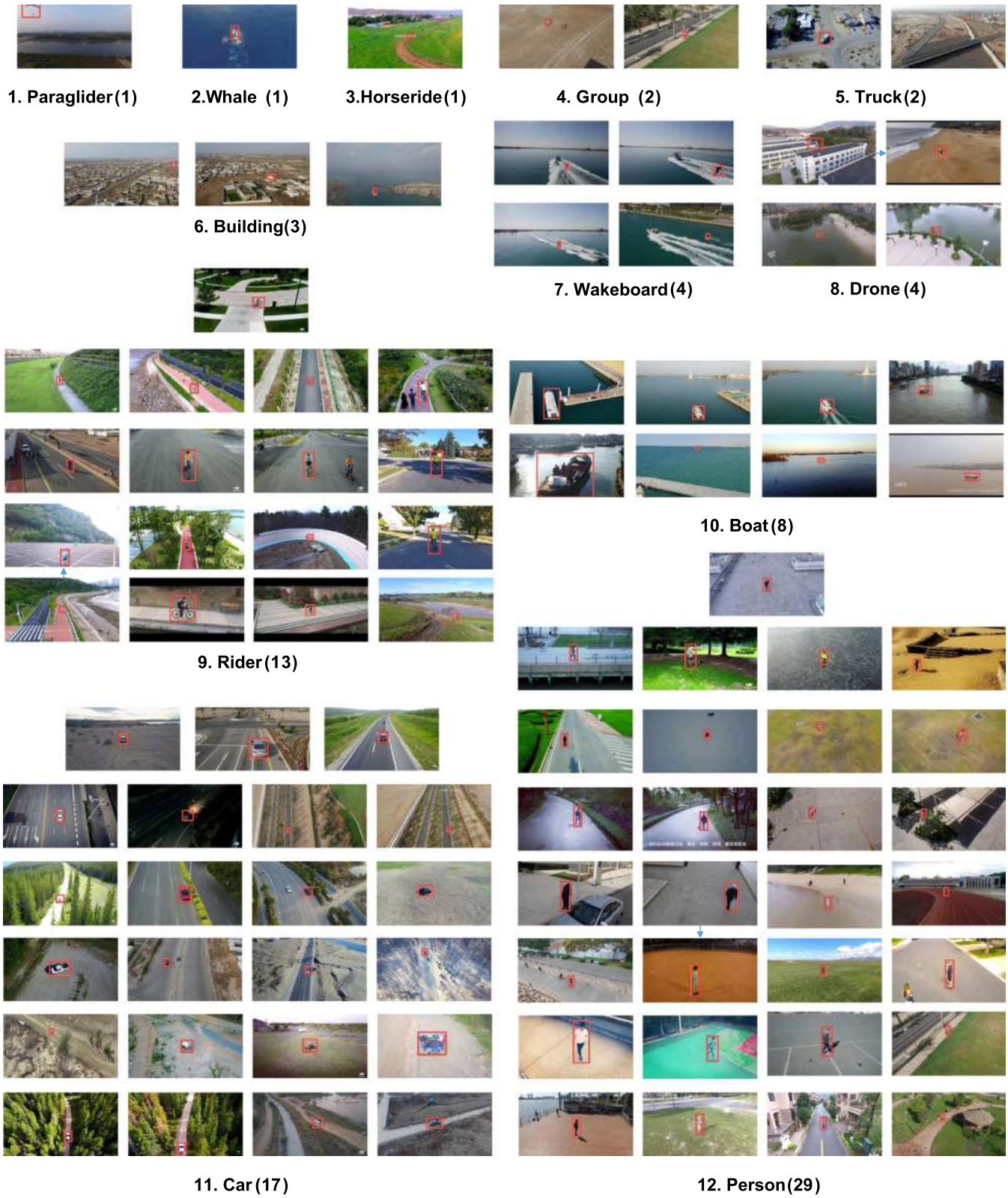


Fig. 1. Overview of the dataset provided by DJI. There are 12 categories, each of which includes several sub-categories as indicated in the bracket, and there are totally 95 sub-categories. Note that there is only one object in each image.

the image quantity decreases. Note that the average object size ratio in ILSVRC is 17 percent and in PASCAL VOC is 20 percent. However, in this dataset, most of the images have an object size of 1-2 percent of the captured images (640 x 360), which is the main character of UAV-view images. This good balance still holds for image brightness and amount of information as shown in Figs. 4 and 5. Both the two distributions in the two figures have the same shape: most of the images have a moderate brightness/amount of information, while many fewer images contain too large or too small brightness/amount of information, which are like a Gaussian distribution.

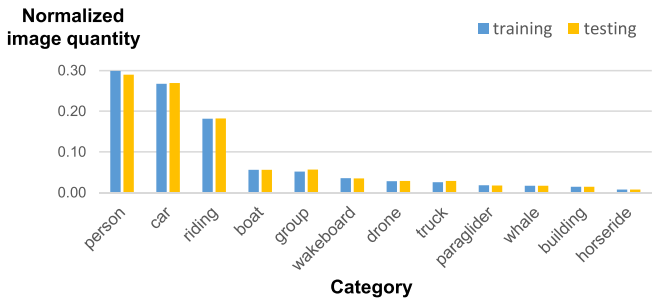


Fig. 2. Distributions of the training and testing datasets with respect to image categories.

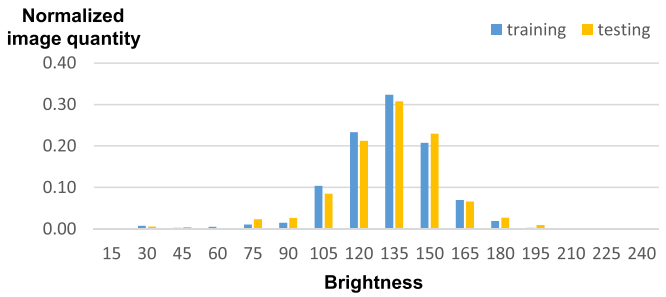


Fig. 4. Distributions of the training and testing datasets with respect to image brightness.

4 Methods for Object Detection

In this section, the analysis and discussion of the methods for object detection reported by the representative GPU and FPGA entries are presented. The details of the top-3 entries of GPU and FPGA categories are discussed, and statistical significance analysis is also presented.

4.1 GPU

A total of 24 out of the 53 participating teams successfully implemented their designs in the GPU category, and all of them adopted deep learning based approaches. The distribution of used neural network models and deep learning frameworks are shown in Fig. 6. The popular light-weight detection framework, Tiny YOLO [21] is the most widely used model in the challenge, and a majority of the entries achieve high performance by adding some revision to the network structure. Darknet is the most popular deep learning framework as tiny YOLO is originally implemented in Darknet. The top three entries*ICT-CAS*,*DeepZ*, and*SDU-Legend* all adopted the YOLO model as the base design and improved it with structure and computation optimization. Note that the image size is 640 $\times$ 360 in the challenge, and ICT-CAS and SDU-Legend resized it to improve accuracy and throughput.

ICT-CAS adopted the original tiny YOLO as their network structure as shown in Fig. 7a, and deployed the Tucker decomposition and hard example mining for accuracy enhancement, and low-bits computation for fast processing. In Tucker decomposition, they tested different decomposition parameters for optimal precision and throughput. By extracting the hard examples, re-training was performed to increase accuracy. In order to speed up the inferencing stage, they deployed half precision float point computation to reduce computation complexity and power consumption. In the TX2 GPU platform, this entry also adopted TensorRT [2] as the inference optimizer to speed up the inference.

DeepZ implemented their own network structure as shown in Fig. 7b. It combined Feature Pyramid Network [17] to fuse fine-grained features with strong semantic features to enhance the ability in detecting small objects. Meanwhile, DeepZ utilized focal loss function to mitigate the imbalance between the single ground truth box and the candidate boxes in the training phase, thereby partially resolving occlusions and distractions.

SDU-Legend focused on both neural network and architectural level optimization to achieve better balance between system performance and detection accuracy. SDU-Legend chose YOLO v2 as the starting design point, and performed design space exploration to choose the key training parameters including anchors, coordinates scale in the loss function, batch size, and learning rate policy. Moreover, SDU-Legend reduced YOLO v2 network architecture from 32 layers to 27 layer (as shown in Fig. 7c), and decreased the downsampling rate to strengthen the performance on small targets. At the architectural level, SDU-Legend aimed at balancing the workload between the GPU and the CPU by executing the last 2 layers on the CPU. SDU-Legend utilized the half data type (16-bit float) instead of 32-bit float to improve the memory throughput and reduce computation cost with minimum loss in accuracy.

4.2 FPGA

There are a total of seven out of the 61 participating teams that successfully implemented their designs on the FPGA platform provided. Among these entries, only entry *TGIIF* adopted Verilog hardware description language for compact FPGA implementation, while the rest utilized Xilinx high-level synthesis for fast FPGA implementation.

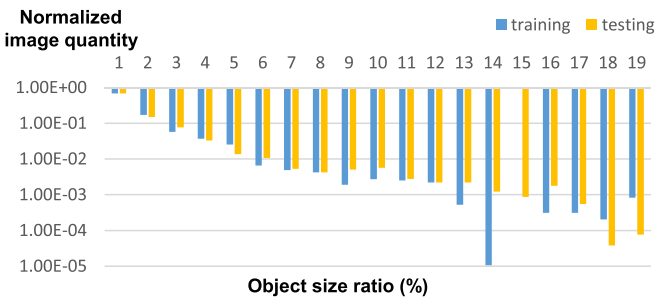


Fig. 3. Distributions of the training and testing datasets with respect to object size ratio.

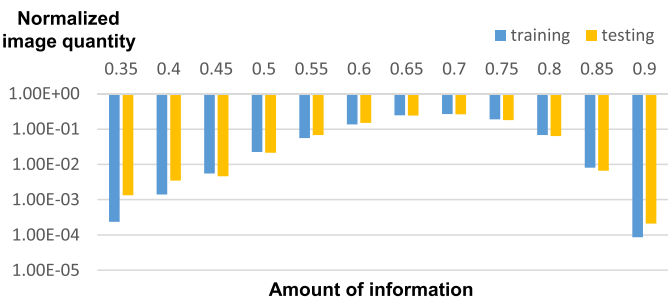


Fig. 5. Distributions of the training and testing datasets with respect to amount of information.

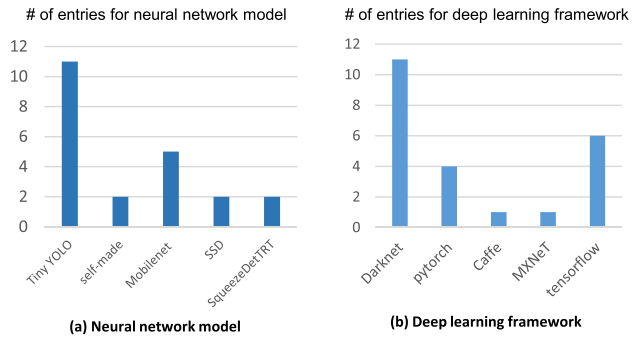


Fig. 6. Number of entries using (a) neural network models and (b) deep learning frameworks.

The top three entries *TGIIF*, *SystemsETHZ*, and *iSmart2* adopted Convolution Neural Network and used variations on the general approaches such as Single Shot MultiBox Detector (SSD) [19], SqueezeNet [14], MobileNet [13], and Yolo [21]. From the original models, the proposed models used 1) fewer layers, 2) dynamic precision, 3) pruned topology, and 4) layer-shared Intellectual Property (IP). To further speed up inferencing, these entries downsized the images in the ARM processor before FPGA processing. In order to compensate the limited on-chip BRAM, these entries utilized the off-chip 512 MB DRAM to store the intermediate data between network layers.

The entry *TGIIF* proposed an optimized SSD network. It downsized the SSD network topology by removing the last two convolutional layers and quantized the network parameters to eight-bit fixed point. After modifying the network depth, it also pruned and fine-tuned the resultant network, making it with 14.2x less parameters and 10x less operations. The network topology proposed by *TGIIF* is shown in Fig. 8a.

The entry *SystemsETHZ* proposed a variation of SqueezeNet+Yolo networks. It reduced the number of parallel convolution operations in fire layer of SqueezeNet [14] to half and binaried the fire layer, and introduced a deep network consisting of 18 convolutional layers where the halfFire layers are binary. The network topology proposed by *SystemsETHZ* is shown in Fig. 8b. It also adopted dynamic precision weights among different layers: five-bit fixed point parameters in all activation layers, eight-bit fixed point

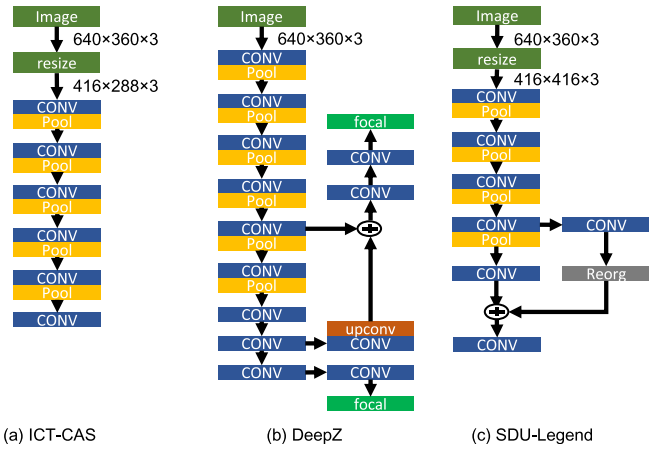


Fig. 7. Neural network structure of the top-3 GPU entries: (a) ICT-CAS, (b) DeepZ, and (c) SDU-Legend.

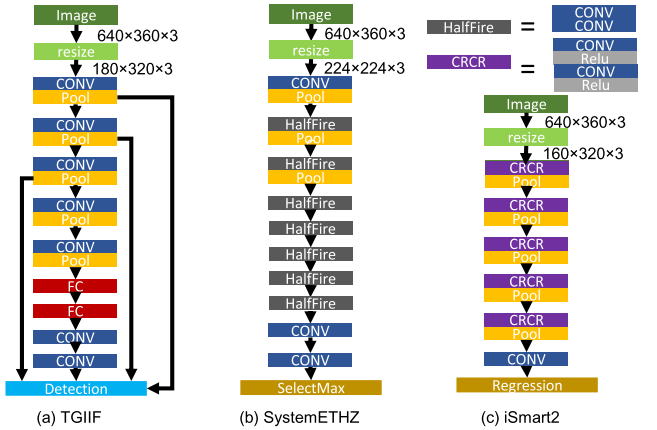


Fig. 8. Neural network structure of the top-3 FPGA entries: (a) TGIIF, (b) SystemETHZ, and (c) iSmart2.

parameters in the first convolutional layer, and binary weights in all fire layers. With dynamic precision, it reduced the weight size to 64 KB and the number of multiplication operations to 154 millions.

The entry *iSmart2* proposed a variation on MobileNet +Yolo networks. It introduced a hardware-friendly network, consisting of multiple depth-wise separable convolution kernels. In each kernel, it adopted a convolution-Relu-Convolution-Relu topology. The network topology proposed by *iSmart2* is shown in the Fig. 8c. By calling one depth-wise separable convolution IP when processing different layers, the proposed network achieved a relatively low resource utilization with minimum look-up-table and Flip-Flop usage.

The entry *traix* proposed a varied SSD network with sixteen-bit fixed-point parameters. The entry *hwac_object_tracker* proposed a varied Tiny YOLO network topology with half-precision floating point parameters. The entry *Lilou* proposed a binaried VGG16 network topology with less pooling layers, thus retaining the inference data flow on the FPGA. The entry *Qiu's Team* proposed a varied PYNQ-BNN [3], adopting 2-bit precision for all layers' parameters.

The FPGA resource utilization of all the entries is shown in Table 1. For entries with sixteen or eight bit-width (*TGIIF*, *SystemsETHZ*, *iSmart2*, *traix*, *hwac_object_tracker*), the DSP utilization is close to 90 percent. The top entry *TGIIF* achieved a 100 percent DSP utilization. For entries with one or two bit-width (*Lilou*, *Qiu's Team*), the DSP utilization is as low as 12 percent (*LiLou*), while the LUT utilization can be as high as 100 percent (*Qiu's Team*).

TABLE 1
Resource Utilization of FPGA Entries

Entries	LUTs (53K)	Flip-Flop (106K)	BRAM (630KB)	DSP (220)
TGIIF	83.89%	54.24%	78.93%	100%
SystemsETHZ	88%	62%	77%	78%
iSmart2	63%	22%	95%	86%
traix	90%	-	90%	90%
hwac_object_tracker	85.02%	41.51%	42.14%	87.73%
Lilou	75%	38%	98%	12%
Qiu's Team	100%	61%	96%	23%

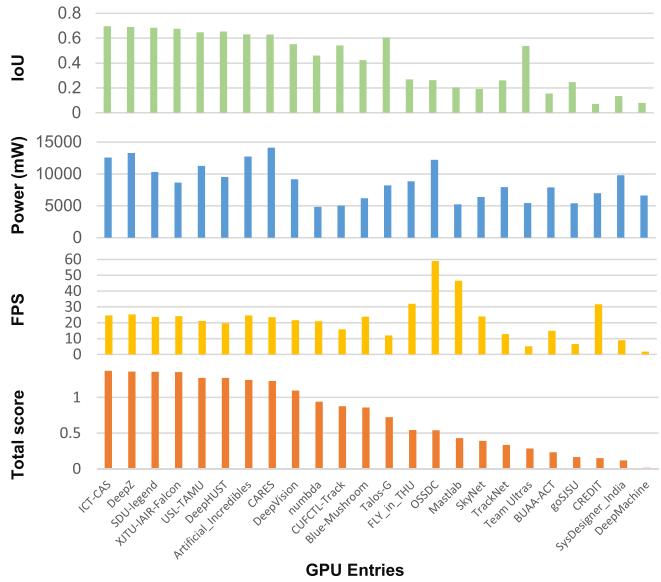


Fig. 9. Overall results of GPU entries. Note that the entries are ranked from high to low in the horizontal axis. The details of the scores can be found on the website of the challenge.

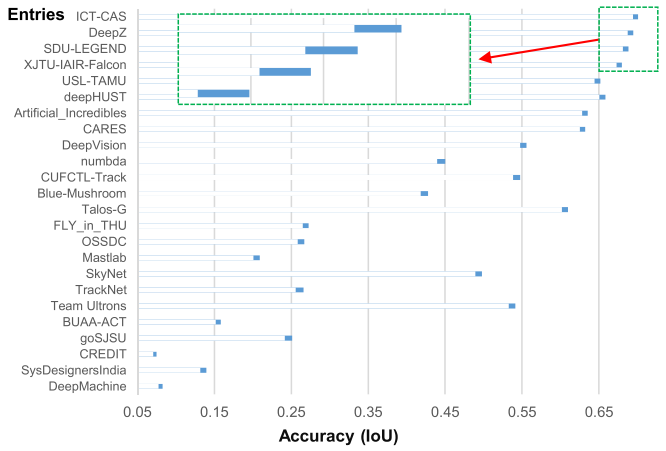


Fig. 10. Statistical significance analysis using bootstrapping with 99.9 percent confidence intervals for GPU entries. Note that the entries are ranked from high to low in the vertical axis.

5 Results and Analysis

In this section, we will discuss and analyze the result with respect to method, category, object size, image brightness, and amount of information for both GPU and FPGA entries. As power and throughput are determined once the method is chosen, we focus on the detection accuracy in this section.

5.1 Results of GPU Entries

5.1.1 Overall Results

As shown in Fig. 9, the optimal IoU, power and FPS are 0.6975, 4,834 mW, and 58.91 FPS, respectively. Note that all the top-3 entries (ICT-CAS, DeepZ, SDU-Legend) have high IoUs and FPS. Only high IoU or FPS is not enough to achieve a high total score/good ranking. For example, the entry OSSDC obtains a rather high FPS and a low IoU, which only ranks the 15th. While another entry Talos-G achieved a high IoU and a low FPS (lower than 20), which triggers the penalty as discussed in Equation (2) and only ranks the 14th.

Actually all the top-8 entries get IoUs higher than 0.60 and throughput higher than 20 FPS. Moreover, their values are both very close to each other which shows rather fierce competition among the top entries. Compared with IoUs, the power has a less influence on the final ranking. The large variation in power consumption is due to fact that the GPU platform has several power modes and these entries choose different modes.

In order to analyze whether results of different entries are statistically significantly different from each other, statistical significance analysis is performed. The bootstrap method is adopted here which is also employed by PASCAL VOC [9] and ImageNet [22]. In each bootstrap round, M images are sampled with replacement from the available M testing images and the average IoU for the sampled images is obtained for one entry. The above process iterates for each entry until reaching the pre-defined bootstrapping round. With the results obtained from all the bootstrapping rounds, the lower and upper a fraction are discarded, and the range of the remaining results are the 1-2a confidence interval. We set the number of bootstrapping rounds to 20,000 and a to 0.0005 (99.9 percent confidential interval), and the final results of the entries are shown in Fig. 10. It can be observed that almost all the entries are statistically significantly different from each other, and the difference of the top-4 entries are also very obvious even with minor differences.

5.1.2 Detection Results by Category

As shown in Fig. 11a, the category *boat* is with the highest detection accuracy as it contains moderate quantity of images and its object size is relatively larger than other categories as shown in Fig. 1. The category *person*, *rider*, and *car* are with a high accuracy as their image quantities are large which can provide a large variety of the object for training as shown in Fig. 2. The category *drone* and *paraglider* also get high accuracy (though their image quantities are small) which is due to the fact that their backgrounds are usually simple such as the sky and their structures are also very special compared with others.

The rest six categories are with relatively lower accuracy as their image quantities are small as shown in Fig. 2. Furthermore, the category *whale* has a very low contrast between the object and the background (the sea) as shown in Fig. 1. The category *building* is rather challenging as there exists many similar objects which results in a very low accuracy. The category *group* gets the lowest accuracy as there also exists multiple similar objects which makes it very hard to detect the right one.

5.1.3 Detection Results by Object Size

In Fig. 3 the quantity of images with 1 percent object size ratio is larger than that of images with 2 percent object size ratio. However, as shown in Fig. 11b, the accuracy of images with 1 percent object size ratio is much lower than that of images with 2 percent object size ratio. The main reason is that too small object is much harder to be detected with high accuracy. The accuracy of the images with 2-5 percent object size ratios is almost the same and relatively higher than others as their

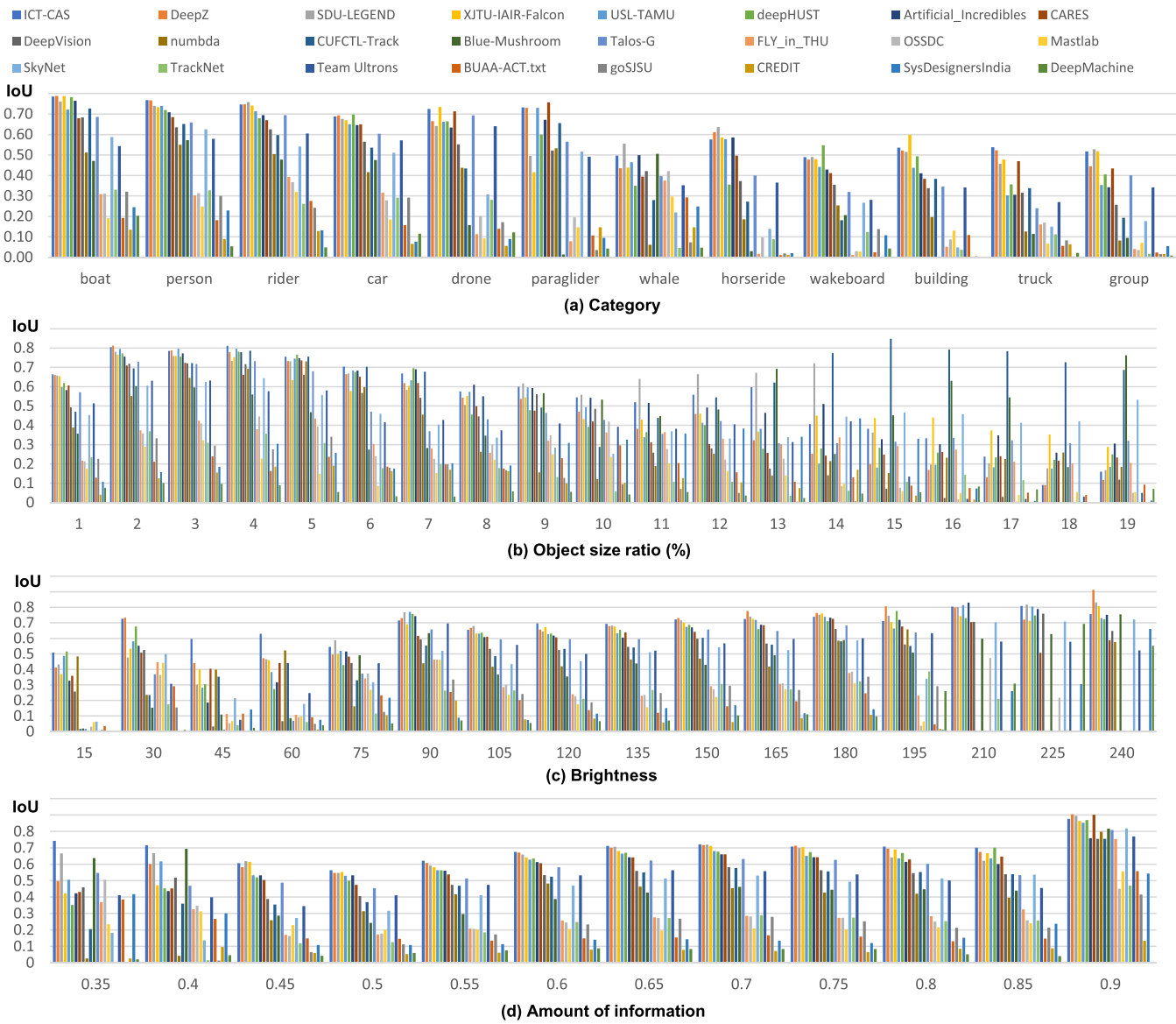


Fig. 11. Detection accuracy of GPU entries with respect to (a) category, (b) object size, (c) brightness and (d) amount of information. Note that in (a) the category is sorted in a ranked order and the left-most one is the one with the highest average total score.

corresponding image quantity is large and the object size is moderate. However, when the object size ratio increases from 6 percent, the accuracy decreases as the corresponding training image quantity is much lower than others.

5.1.4 Detection Results by Image Brightness

As shown in Fig. 11c, there is a almost linear trend between the accuracy and the image brightness. As the image brightness increases, the accuracy also increases. However, if the accuracy is directly correlated with the image brightness, the accuracy with a brightness of about 135 should be the highest as the image quantity with this brightness is the largest. Thus, we tend to believe that higher brightness will make the object more clear to show its features which will make it relatively easier to be detected with high accuracy.

5.1.5 Detection Results by Amount of Information

As shown in Fig. 11d, images with larger amount of information tend to have higher accuracy as larger amount of information contains more features which are relatively easier to be detected. The images with amount of information of 0.35-0.40 get a slightly higher accuracy than that with amount of information of 0.45-0.50 which may caused by that these images are with unique characteristics and a small quantity. We can also notice that the entry ICT-CAS can get a much higher accuracy for images with amount of information of 0.35-0.40 than others, and get relatively lower accuracy for images with amount of information of 0.90 than other top-3 entries.

5.2 Results of FPGA Entries

5.2.1 Overall Results

Fig. 12 summarizes the performance of all FPGA entries. Entry TGIIF stands out as the most successful method, obtaining an IoU score of 0.6238 and a throughput of 11.96 FPS. The adopted deep network leads to a high IoU score, and the network pruning ensures the inference throughput. Entry SystemsETHZ comes second while it achieves a

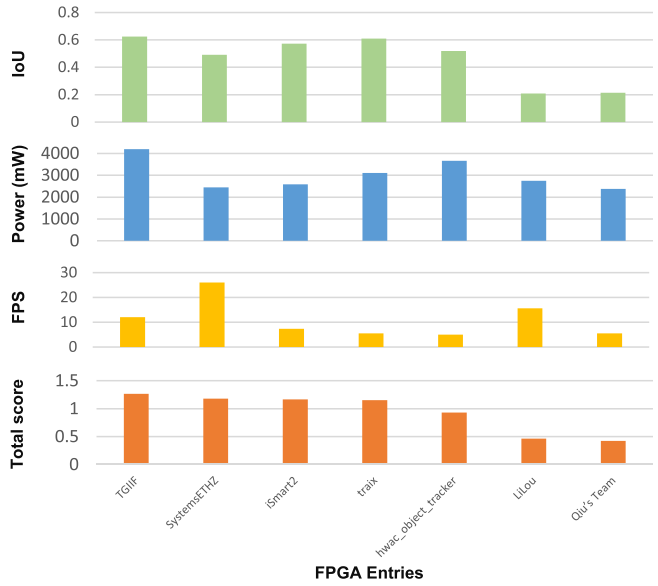


Fig. 12. Overall results of FPGA entries. The details of the scores can be found on the website of the challenge.

throughput of 25.97 FPS and an IoU score of 0.4919. Its dynamic precision architecture and binarized layers accelerate the inference throughput at the expense of IoU loss. iSmart2 comes third with a throughput of 7.35 FPS and an IoU score of 0.5733. Its uniform-layer deep network design gives rise to IP reuse and fine-grained memory allocation, keeping a good balance between throughput and accuracy. The top four entries have similar total scores, but each entry has different accuracy, throughput, and power values. All four entries except SystemETHZ achieve high IoU, while SystemETHZ puts more effort on FPS and obtains a much higher FPS than others. Compared with TGIIF, iSmart2 and traix achieve lower FPS with lower power.

Many entries achieved IoUs of around 0.5-0.6, such as entry *traix* which achieves an IoU of 0.6100 with sixteen-bit fixed-point network parameters. Despite the success in accuracy, there is a quite range in inference throughput. As mentioned in Sections 2.2 and 4.2, the FPGA platform has 630 KB on-chip BRAM, which is not sufficient to retain all parameters and output feature maps of layers. Thus, a frequent data transfer between DRAM and FPGA are observed in all designs, and then keeping a proper balance between accuracy and network size is very important in this challenge. The standout top-3 entries adopted network pruning, parameters binarization and IP-reuse to ensure the inference throughput.

The statistical significance analysis is shown in Fig. 13. It can be easily noticed that all the entries are statistically significantly different from each other, and the difference of the top-3 entries are even larger and there is also moderate difference between their accuracies.

5.2.2 Detection Results by Category

Fig. 14a summaries the results obtained according to object category. The categories are listed in sequence of average IoU. There is drastic variation among categories and in different entries, in which, *boat* keeps the most promising average IoU of 0.5734 and *group* with an average IoU of 0.2060 is

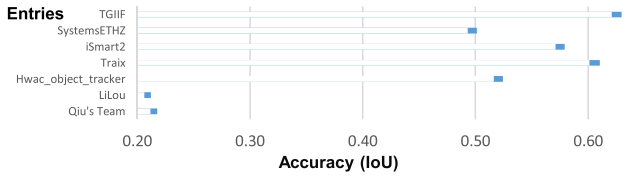


Fig. 13. Statistical significance analysis using bootstrapping with 99.9 confidence intervals for FPGA entries.

the most challenging category to be detected. On average IoU, the most promising categories are *boat*, *person*, *rider*, and *car*, while those four categories have moderate quantities of images and their object sizes are relatively larger than other categories. Among those four categories, *boat* stands out as the most promising category while it contains 5k (the least) training images compared with *person* which contains 28k (the most) training images. This is due to the fact that the background in the *boat* category is simple (the sea) and the target is distinct, while the *person* category contains numerous noise such as trees, grass, and even other non-objected persons. The *rider* stands ahead of *car* while the *car* category contains 25k training images and *rider* contains only 1.6k training images. This is due to the fact that the *rider* category is captured from a relatively close view while *car* category is captured from a distant view.

For these challenging categories such as *whale*, *truck*, *building*, and *wakeboard*, there is an approximately 20 percent accuracy gap between these categories and the promising categories for both maximum IoU and average IoU. Not only the low image quantity, but also the complex object features worsen the detection accuracy. The features of these four challenging categories can be summarized as follows: the category *whale* contains blurry object and background, while categories *truck*, *wakeboard*, and *building* have tiny objects or similar objects to the target objects such as the same color and shadow.

We notice an interesting phenomenon where entry traix achieves the highest accuracy except the category car but does not have the highest IoU as shown in Fig. 12. This is due to the fact that the category car occupies a large part (more than a quarter) of all the test images as shown in Fig. 2. Though traix can achieve slightly higher accuracy on all categories except car than TGIIF, it obtains much lower accuracy (0.44) than TGIIF (0.65) on the category car. Considering the large number of test images in the category car, TGIIF finally achieves a slightly higher overall accuracy than traix.

5.2.3 Detection Results by Object Size, Image Brightness, and Amount of Information

Considering the object size, image brightness, and amount of information, the detection results of FPGA entries show the same trend as that of GPU entries. A moderate object size can usually achieve the highest detection accuracy as shown in Fig. 14b. Higher brightness usually leads to higher detection accuracy as shown in Fig. 14c. In Fig. 14d, we can notice that larger amount of information tends to achieve higher accuracy.

5.3 Hard Examples and Lessons

The results for GPU and FPGA entries show almost the same phenomena and trends as they adopted the same

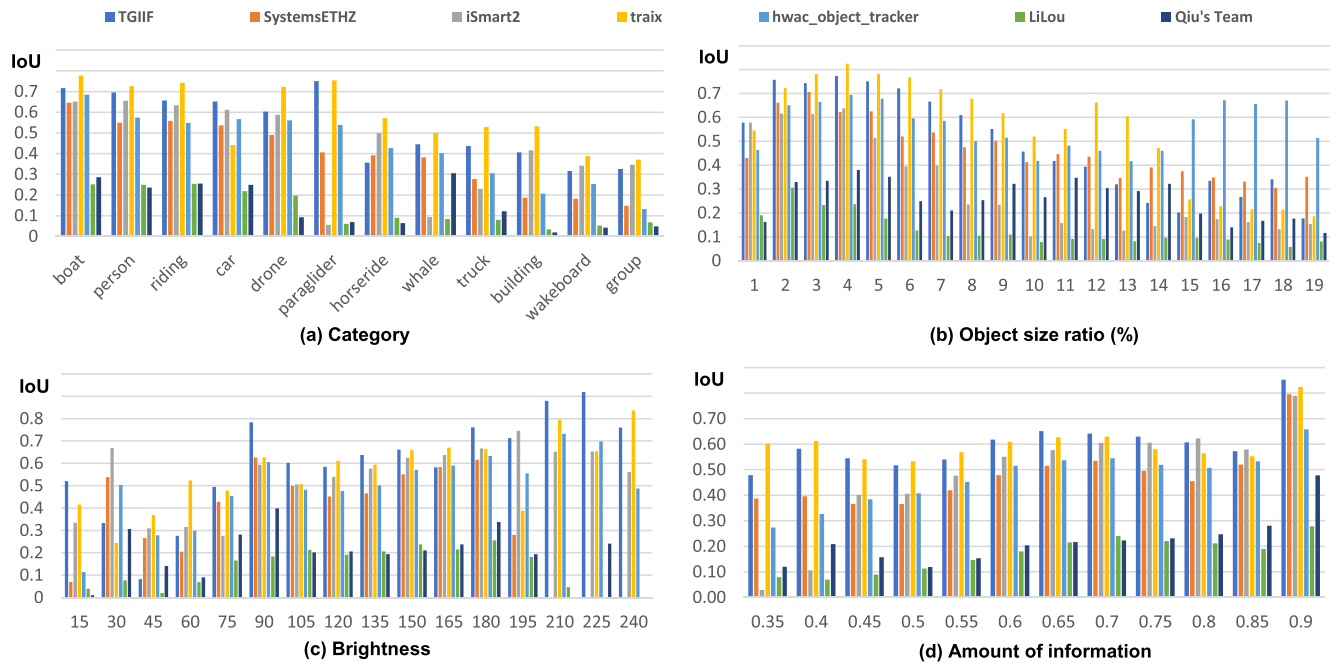


Fig. 14. Detection accuracy of FPGA entries with respect to (a) category, (b) object size, (c) brightness and (d) amount of information. Note that in (a) the category is sorted in a ranked order and the left-most one is with the highest average total score.

method (deep neural networks), and their hard-to-detect images are also almost the same. Fig. 15 shows some hard examples with hard categories, small sizes, low brightness, and low amount of information for both GPU and FPGA entries. The most challenge category is *group* as discussed in Sections 5.1.2 and 5.2.2. As shown in Figs. 15a and 15b, the objects in the two images are very small, and there are several similar objects around, which makes accurate detection rather hard. The two images (c) and (d) contain very small objects (*rider* in image (c) and *truck* in image (d)), both of which are hard to be recognized even by humans. Images (e) and (f) are two images with very low brightness, and the objects are very blurry without clear outlines. With the context in image (e), humans can infer that the object is a car. However, the object in image (f) is very hard for humans to recognize as the light is too dim. The objects in image (h) and (i) are with very low average amount of information, which is also hard to recognize. Image (h) contains a very small object which is almost invisible and has very limited information, while image (i) has a object which is large however with smooth surfaces and unclear boundaries with the sea. Within all the hard examples, it can be observed that almost all the images are within very small objects. In fact small objects are common for UAV applications which is the major challenge for accurate object detection. Furthermore, similar objects (*persons*, *rider*, *buildings*, *boats*, *cars*, etc.) and special scenarios (*wake board*, operation at night) add more difficulties.

With the above hard examples and previous results discussed, we present the learned lessons as follows. First, FPGA is much more energy efficient than GPU. Though FPGA achieves a relatively lower FPS than GPU, it can achieve almost the same accuracy but with only 1/3-1/2 energy consumption as that of GPU, which is promising for long-term UAV applications. Second, object detection from UAV views in real world is complicated. In the challenge, there are many images that can not be accurately detected by all the entries. Dividing the task into well-defined sub-

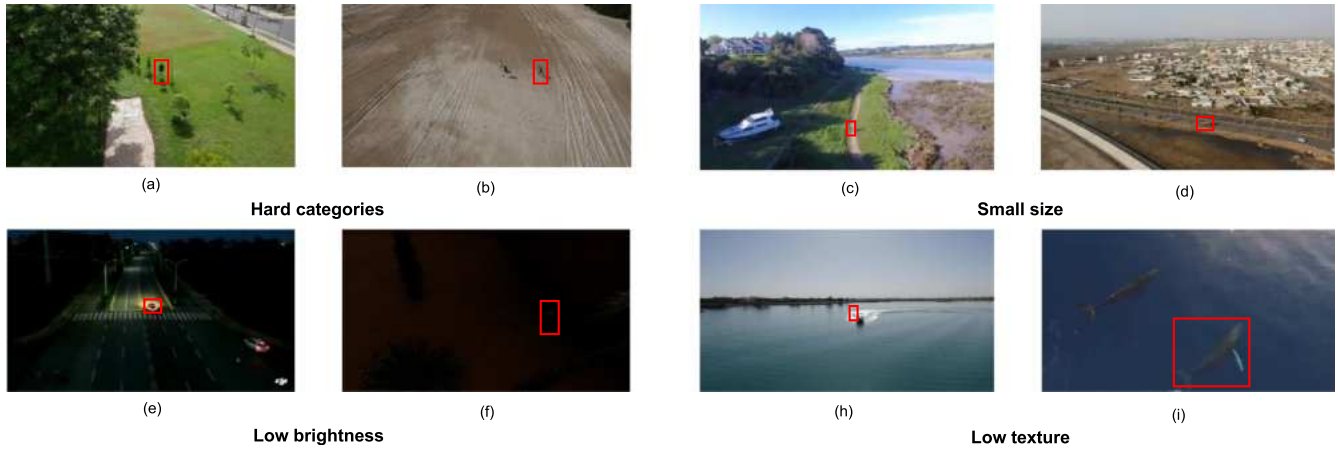


Fig. 15. Hard examples with hard categories, small size, low brightness, and low amount of information in the challenge.

tasks for specific scenarios may improve the performance. Third, more data is preferred for more accurate detection. In the challenge we find that many objects get a low detection accuracy when their brightness or view-angle changes. Training images with more diversity (scale, view-angel, etc.) of the object will further improve the overall accuracy.

6 Conclusion

In this paper we present the DAC-SDC low power object detection challenge for UAV applications. Specifically, we describe the unique dataset from UAV views, and give a detailed discussion and analysis of the participating entries and their adopted methods. We also discuss the details of methods proposed by the LPODC entries for efficient processing for UAV applications. The result analysis provides a good practical lesson for researchers and engineers to further improve energy-efficient object detection in UAV applications.

Acknowledgments

The authors would like to thank DJI for providing the dataset, Xilinx and Nvidia for providing free GPU and FPGA platforms, and DAC organizers for their support to the DAC SDC challenge. Xiaowei Xu and Xinyi Zhang contribute equally to this work.

References

[1] [Online]. Available: https://developer.nvidia.com/embedded/dlc/jetson-tx2-thermal-design-guide. Accessed on: Aug. 13, 2019.
[2] [Online]. Available: https://developer.nvidia.com/tensorrt. Accessed on: Aug. 13, 2019.
[3] [Online]. Available: https://github.com/xilinx/bnn-pynq. Accessed on: Aug. 13, 2019.
[4] [Online]. Available: https://www.dji.com/. Accessed on: Aug. 13, 2019.
[5] Nvidia, 2018. [Online]. Available: http://www.nvidia.com/page/home.html. Online; Accessed on: Jul. 5, 2018
[6] Open images, 2018. [Online]. Available: https://www.kaggle.com/bigquery/open-images. Online; Accessed on: Jul. 4, 2018
[7] Xilinx, 2018. [Online]. Available: https://www.xilinx.com/. Online; Accessed on: Jul. 5, 2018
[8] H. Chao, Y. Gu, and M. Napolitano, "A survey of optical flow techniques for UAV navigation applications," in *Proc. Int. Conf. Unmanned Aircraft Syst.*, 2013, pp. 710–716.
[9] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.
[10] K. Gauen, R. Dailey, Y.-H. Lu, E. Park, W. Liu, A. C. Berg, and Y. Chen, "Three years of low-power image recognition challenge: Introduction to special session," in *Proc. Des. Autom. Test Europe Conf. Exhib.*, 2018, pp. 700–703.
[11] K. Gauen, R. Rangan, A. Mohan, Y.-H. Lu, W. Liu, and A. C. Berg, "Low-power image recognition challenge," in *Proc. 22nd Asia South Pacific Des. Autom. Conf.*, 2017, pp. 99–104.
[12] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.
[13] 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," *CoRR abs/1704.04861*, 2017.
[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5MB model size," *CoRR abs/1602.07360*, 2016.
[15] B. Janßen, P. Zimprich, and M. Hübner, "A dynamic partial reconfigurable overlay concept for PYNQ," in *Proc. 27th Int. Conf. Field Programmable Logic Appl.*, 2017, pp. 1–4.
[16] M. F. Kragh, P. Christiansen, M. S. Laursen, M. Larsen, K. A. Steen, O. Green, H. Karstoft, and R. N. Jørgensen, "FieldSAFE: Dataset for obstacle detection in agriculture," *Sensors*, vol. 17, no. 11, 2017, Art. no. 2579.
[17] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, vol. 1, pp. 936–944.
[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[20] Y.-H. Lu, A. M. Kadin, A. C. Berg, T. M. Conte, E. P. DeBenedictis, R. Garg, G. Gingade, B. Hoang, Y. Huang, B. Li, et al., "Rebooting computing and low-power image recognition challenge," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2015, pp. 927–932.
[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.
[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[23] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A database and web-based tool for image annotation," *Int. J. Comput. Vis.*, vol. 77, no. 1/3, pp. 157–173, 2008.
[24] A. Sanmorino, "Clustering batik images using fuzzy C-means algorithm based on log-average luminance," *Comput. Eng. Appl. J.*, vol. 1, no. 1, pp. 25–31, 2012.
[25] J. Torres-Sánchez, F. López-Granados, and J. M. Peña, "An automatic object-based method for optimal thresholding in UAV images: Application for vegetation detection in herbaceous crops," *Comput. Electron. Agriculture*, vol. 114, pp. 43–52, 2015.
[26] D.-Y. Tsai, Y. Lee, and E. Matsuyama, "Information entropy measure for evaluation of image quality," *J. Digital Imag.*, vol. 21, no. 3, pp. 338–347, 2008.
[27] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang, "DOTA: A large-scale dataset for object detection in aerial images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3974–3983.
[28] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 3485–3492.
[29] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, and Y. Shi, "Scaling for edge inference of deep neural networks," *Nature Electron.*, vol. 1, no. 4, p. 216, 2018.

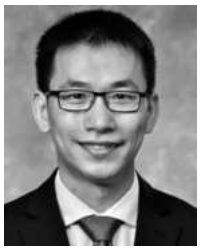
Xiaowei Xu (S'14-M'17) received the BS and PhD degrees in electronic science and technology from the Huazhong University of Science and Technology, Wuhan, China, in 2011 and 2016 respectively. He is currently a post-doc researcher at University of Notre Dame, IN. His research interests include deep learning, medical image segmentation and embedded computing. He was a recipient of DAC system design contest special service recognition reward, in 2018 and Outstanding contribution in reviewing, Integration, the VLSI journal, in 2017. He has served as TPC members in ICCD, ICCAD, ISVLSI and ISQED. He is a member of the IEEE.

Xinyi Zhang received the BS degree in electrical engineering from the Southwest Jiaotong University, Chengdu, China, in 2014, and the MS degree in electrical engineering from the University of New Mexico, Albuquerque, NM, in 2016. He is currently working toward the PhD degree at the University of Pittsburgh, Pittsburgh, PA under the advisement of professor Jingtong Hu. His current research interests include non-volatile FPGA architecture, FPGA place and routing, and deep learning algorithm hardware accelerator design. He is a student member of the IEEE.

Bei Yu (S'11–M'14) received the PhD degree from The University of Texas at Austin, in 2014. He is currently an assistant professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. He has served as TPC chair of ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He is editor-of-chief of IEEE TCCPS Newsletter. He received five Best Paper Awards from Integration, the VLSI Journal, in 2018, International Symposium on Physical Design 2017, SPIE Advanced Lithography Conference 2016, International Conference on Computer Aided Design 2013, and Asia and South Pacific Design Automation Conference 2012, and five ICCAD/ISPD contest awards. He is a senior member of the IEEE.

Xiaobo Sharon Hu (S'85-M'89-SM'02-F'16) received the BS degree from Tianjin University, China, the MS degree from Polytechnic Institute of New York, and the PhD degree from Purdue University, West Lafayette, Indiana. She is professor with the Department of Computer Science and Engineering, University of Notre Dame. Her research interests include computing with beyond-CMOS technologies, low-power system design and cyber-physical systems. She has published more than 300 papers in the related areas. She served as associate editor for the *IEEE Transactions on Very Large Scale Integration*, the *ACM Transactions on Design Automation of Electronic Systems*, and the *ACM Transactions on Embedded Computing*, and is currently an associate editor for the *ACM Transactions on Cyber-physical Systems* She is the general chair of 2018 Design Automation Conference (DAC), and program chair and TPC chair of 2016 and 2015 DAC. She received the NSF CAREER Award in 1997, and the Best Paper Award from Design Automation Conference in 2001 and ACM/IEEE International Symposium on Low Power Electronics and Design in 2018, etc. She is a fellow of the IEEE.

Christopher Rowen is an American entrepreneur and technologist. He is one of the founders of MIPS Computer Systems, Inc, in 1984, of Tensilica Inc., in 1997 and of Babblelabs, Inc, in 2017. He was named fellow of the Institute of Electrical and Electronics Engineers (IEEE), in 2016 for leadership in the development of microprocessors and reduced instruction set computers.



Jingtong Hu is currently an assistant professor with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA. His current research interests include embedded systems, FPGA, and nonvolatile memory. He has published more than 30 ACM/IEEE Transactions and 70 conference papers in these areas. His papers have been nominated for Best Paper Award by ISQED 2018, DAC 2017 and 2019. His research has been supported by NSF, AFRL, ARL, Altera, etc. He is a recipient of Outstanding new Faculty Award from Oklahoma State University, Summer Faculty Fellowship from AFOSR. He has served as the chair for SIGDA System Design Contest at DAC, SIGDA CADathlon Competition at ICCAD, Student Research Forum at ASP-DAC, track chair for of Design Automation Conference Technical Program Committee and ACM Symposium on Applied Computer Technical Program Committee, and guest editor for the *IEEE Transactions on Computers*. He is a member of the IEEE.

Yiyu Shi (S'06-M'10-SM'15) received the BS degree in electronic engineering from Tsinghua University, Beijing, China, in 2005, the MS and PhD degrees in electrical engineering from the University of California, Los Angeles, in 2007 and 2009 respectively. He is currently an associate professor with the Department of Computer Science and Engineering, University of Notre Dame, the site director of NSF I/UCRC Alternative and Sustainable Intelligent Computing, and the director of the Sustainable Computing Lab (SCL). His current research interests focus on hardware intelligence and biomedical applications. In recognition of his research, many of his papers have been nominated for the Best Paper Awards in top conferences. He was also the recipient of IBM Invention Achievement Award, Japan Society for the Promotion of Science (JSPS) Faculty Invitation Fellowship, Humboldt Research Fellowship, IEEE St. Louis Section Outstanding Educator Award, Academy of Science (St. Louis) Innovation Award, Missouri S&T Faculty Excellence Award, NSF CAREER Award, IEEE Region 5 Outstanding Individual Achievement Award, and the Air Force Summer Faculty Fellowship. He has served on the technical program committee of many international conferences including DAC, ICCAD, DATE, ISPD, ASPDAC and ICCD. He is on the executive committee of ACM SIGDA, a member of IEEE CEDA Publicity Committee, deputy editor-in-chief of IEEE VLSI CAS Newsletter, and an associate editor of the *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, the *IEEE Access*, the *ACM Journal on Emerging Technologies in Computing Systems*, the *VLSI Integration*, and the *IEEE TCCCPS Newsletter*. He is a senior member of the IEEE.