Multi-task ADAS system on FPGA

Jinzhang Peng[1], Lu Tian[2,1], Xijie Jia[1], Haotian Guo[1], Yongsheng Xu[1]
Dongliang Xie[1], Hong Luo[1], Yi Shan[1], Yu Wang[2]
[1]Xilinx,Inc.
[2]Department of Electronic Engineering, Tsinghua University
jinzhang@xilinx.com

***Abstract*— Advanced Driver-Assistance Systems (ADAS) can help drivers in the driving process and increase the driving safety by automatically detecting objects, doing basic classification, implementing safeguards, etc. ADAS integrate multiple subsystems including object detection, scene segmentation, lane detection, and so on. Most algorithms are now designed for one specific task, while such separate approaches will be inefficient in ADAS which consists of many modules. In this paper, we establish a multi-task learning framework for lane detection, semantic segmentation, 2D object detection, and orientation prediction on FPGA. The performance on FPGA is optimized by software and hardware co-design. The system deployed on Xilinx zu9 board achieves 55 FPS, which meets real-time processing requirement.**

Keywords: ADAS; Multitask framework; Software-hardware co-optimization; FPGA

I. INTRODUCTION

Advanced Driver Assistance Systems, or ADAS, are systems to help the driver in the driving process and aim at improving traffic safety. ADAS has been becoming one of the most active areas in recent years and deep learning plays an important role in automotive fields because of its high accuracy and reliability. ADAS are complex systems consisting of FCW (Forward Collision Warning), PCW (Pedestrian Collision Warning), LDW (Lane Departure Warning), etc., which integrate multiple modules with different computer vision tasks such as object detection, semantic segmentation, and lane detection.

Deep learning brings incredible improvement in both accuracy and reliability compared with traditional algorithms in various computer vision tasks. For example, ResNet[1], GoogLeNet[2] and VGG-Net[3] have been proposed for image classification. Object detection approaches such as Faster R-CNN [4], YOLO[5] and SSD[6] are widely used in ADAS for pedestrian detection and vehicle detection. DeconvNet[7], SegNet[8] and U-Net[9] are popular networks for semantic segmentation. For Lane detection, LaneNet[10] and VPGNet[11] have also been developed and outperform traditional algorithms.

While deep neural networks are usually accompanied by high computational complexity, the ADAS, served as assistants for car driving, should feedback real-time perception of the road environment to the drivers. Thus, high processing frame rate should be guaranteed in ADAS. However, whether for power, area or financial considerations, the computing resources embedded on cars are always limited. Therefore, it is necessary and challenging to integrate multiple tasks on limited hardware.

Multi-task learning provides a promising and efficient solution for ADAS. Multi-task learning is an increasingly active field of machine learning, in which multiple learning tasks are implemented by sharing the same model while exploiting commonalities and differences across tasks. It brings two benefits compared with the task-specific designs. On one hand, by sharing the backbone encoder computations, the whole computational complexity will be substantially reduced. On the other hand, multi-task learning may result in improved learning efficiency and better prediction accuracy, since there are mutual and inherent associations and constraints among different tasks [12], [13].

In addition, to further accelerate the processing speed, the ADAS should also be optimized from both software and hardware on FPGA. To fully utilize the resources on the FPGA board and build a highly energy efficient system, software-hardware co-design should be considered [14], [15], [16].

This paper proposes an efficient solution for ADAS on FPGA. Specifically, the main contributions of this work are listed as follows. (1) We propose a multi-task learning framework for ADAS, which integrate the tasks of object detection, orientation prediction, lane detection, and semantic segmentation, and achieve comparable precision with task-specific models. (2) We optimize the system performance through the joint optimization of software and hardware. The system deployed on Xilinx zu9 [17] board achieves 55 FPS for one channel.

The rest of the paper is organized as follows. Section II shows the ADAS algorithms of the proposed system. Section III and IV illustrates the detail of the software and hardware optimization in ADAS system. The section V shows the experimental results of the system. Finally, the section VI concludes the paper.

II. ALGORITHM

The Deep Neural Networks (DNN) are widely utilized in ADAS and we focus on the improvement and acceleration of deep learning-based algorithms.

The authors represent the combined effort of many talented engineers inside Xilinx from the algorithm, software framework, compiler, FPGA, and SoC design teams. Yu Wang's work is supported by supported by National Key RD Program of China (2018YFB0105005) and National Natural Science Foundation of China (No. 61622403, 61621091).

A. Single task algorithm

1) Semantic Segmentation : Most semantic segmentation algorithms use end-to-end networks, consisting of encoder and decoder parts. As Fig 1shows, the result of image segmentation is a set of segments that collectively cover the entire image. While deploying segmentation models on FPGA is challenged by the following factors.

Large computation cost: State-of-the-art semantic segmentation algorithms usually require several TFLOPs or hundreds of GFLOPs per image, which cannot guarantee the frame rate for real-time applications. Therefore, we propose a light-weight semantic segmentation network based on Feature Pyramid Network (FPN) structure [13], which obtain higher accuracy and reduce lots of workloads compared with heavy networks. Our network consumes only 9 GOPs per image (with 512x256 input) and achieves 57.59% mIOU on cityscapes 500 validation dataset [18].

Extra upsample computations: Besides conventional operations in CNN. (e.g. convolution, activation ,and pooling), the decoder part requires some special operations like deconvolution or nearest/bilinear interpolation to upsample the feature map to the original size of the input image.

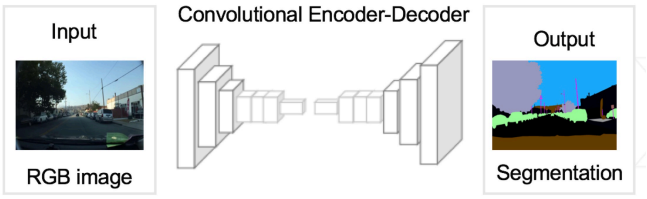


Fig. 1. Segmentation Architecture Design.

2) Object Detection: The processing flow of object detection is more complicated than segmentation. As shown in Fig 2, there are two prediction branches following the feature extraction backbone network. One branch is for bounding box regression, which obtains the coordination of bounding boxes based on pre-defined anchors and regressed localization offsets. The other branch is recognizing the corresponding category of the object in each bounding box.

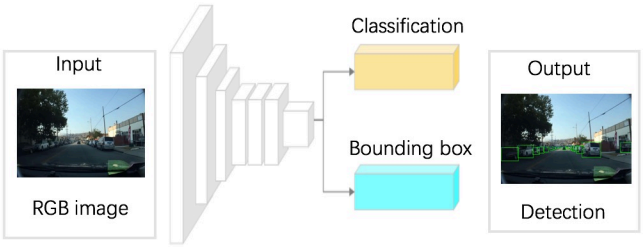


Fig. 2. Object Detection Architecture Design.

Different from the segmentation network design, we obtain a light-weight SSD through quantizing and pruning the original SSD models. The convolutional layers of light-weight SSD are iteratively pruned thinner, and the weights and activations are quantized into 8-bit numbers. The light-weight SSD object detection model is deployed on FPGA with 480*360 input, with the computational complexity of 4.9G operations per image. Other than CNN part, the post-processing, such as bounding box transform, NMS, etc., will be run on embedded ARM processor.

3) Lane Detection: lane detection is tackled using segmentation algorithms to predict the label of each pixel, followed by some post-processing like clustering and regression to get the final polynomial function of lanes. By means of quantization and pruning, the VPGNet model is compressed to a lighter network with around 10G computational operations per image(480x360 pixels). The post processing like image transform, clustering, fitting. etc., will also be run on the ARM processor.

4) Orientation Prediction: To further improve the safety of car driving, predicting the moving orientation of objects around the driving car is essential. Orientation is predicted in object detection using angle regression together with bounding box regression and only adds a bit computation cost. During training, the smooth L1 [19] loss is used.

B. Multi-task Network

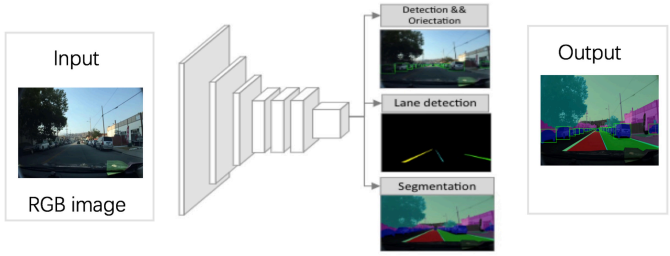


Fig. 3. Multitask Architecture Design.

Our multi-task learning merges multiple individual modules in the ADAS system into a single network, as shown in Fig 3. In this way, a backbone encoder is shared to extract the feature of input images, and then different decoder branches are designed respectively for different tasks.

In this paper, a single stage multi-task network is proposed in our ADAS system for object detection, orientation prediction, segmentation, and lane detection. Naively merging multiple losses from four tasks will cause the problem of convergence in the training. Therefore, we make some optimizations in our multi-task network design. First, lane detection and semantic segmentation are combined into one task and their annotation labels are merged. Thus the supervision of land markings could enrich the semantic information of the road. And the semantic segmentation can reduce the false positive rate of the lane line predictor, vice versa. Second, we propose to extend 2D object detection with orientation prediction as discussed previously. Finally, the optimized loss function is shown in equation 1. The λ represents a hyper-parameter to weight losses from different branches, which will benefit the convergence of training.

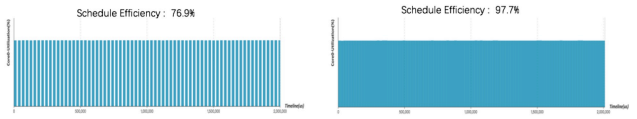
$$L_{loss} = L_{det}(x, c_1, 1, \theta) + \lambda L_{seg}(x, c_2). \quad (1)$$

We design a single stage multi-task model with a computation cost of only 13.6 GOPs, by using the ResNet18 network as the backbone to extract shared features.

III. Software Optimization

Our ADAS system run on DPU (Deep learning Processing Unit) and CPU (ARM) together. DPU handles most computation of CNN, while pre- and post-processing are deployed on CPU. In this section, we will detail the optimization of our software design.

In order to reduce the scheduling overheads of DPU runtime and reach better performance results, we perform some optimizations to the execution mode of DPU instruction stream. The original scheduling granularity for DPU kernel is in a unit of the layer, i.e, DPU kernel executes on DPU in the mode of the layer by layer. Then we reduce most interrupt overhead through combining several layers into one single super-layer, which brings significant improvement to runtime scheduling efficiency. As shown in figure 4, schedule efficiency increase from 76.9% to 97.7%.



(a) (b)

Fig. 4. Indicates DPU scheduling optimization comparison.(a) Express before optimization,(b) Express optimized

For different networks, the memory of pre- and post-processing data is carefully allocated to avoid repeated reading. Moreover, multi-threading and pipeline technology are introduced to fully exploit the processing capacity of ZU9 with its dual-core DPUs and quad-core CPUs. Our software optimization delivers low latency and high-throughput for ADAS applications.

IV. Hardware optimization

Deep learning networks vary and it's unpractical to generate an all-around hardware parameter to meet all networks, especially under resource limits. Generally, less hardware resource design leads to better FPGA timing, higher clock-frequency, lower power, and room for additional functions. We provide tunable parameters to reach a compromise and make it suitable for specified applications. In this section, our hardware architecture is introduced together with the resource optimization for ADAS requirements on high-efficiency performance.

A. Hardware Implementation on FPGA

Fig. 5 illustrates our Aristotle architecture which is a heterogeneous architecture with host CPU, scheduler, on-chip buffer, and computing engine. The CPU running DNNDK (Deep Learning Development Toolkit) is responsible for preparing off-chip memory, scheduling multi-tasks. The instr-scheduler fetches and decodes instructions from off-chip memory, and schedules DPU running. The on-chip

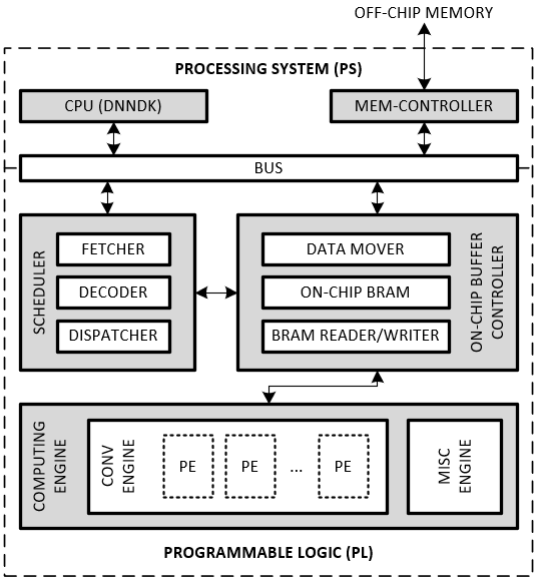


Fig. 5. The hardware architecture

buffer controller manages the transferring and rearrangement of data, such as activation, weights, and bias, among off-chip memory, on-chip BRAM, and computing engines. The parallel computing engine executes heavy tasks like Conv and FC on the conv engine, and other various kinds of workload like element-wise, pooling, and softmax on the misc engine. The Aristotle architecture can support most popular deep learning networks in a flexible and efficient way. For Deconv and Dilated Conv in segmentation networks as an example, only some additional data rearrangement like Split and Concat operations are required, compared with traditional Conv.

B. SERDES-like Optimization

The workloads of network layers are imbalanced. We achieve a SERDES-like structure by reducing the parallelism P of the low-workload layers into Q and reusing them serially. The SERDES ratio R represents P/Q where $P\%Q = 0$.

For example, the PE of Conv operates multiply-accumulate at each cycle for 100% workload. For every M cycles, one group of accumulation is done and followed by one cycle non-linear layers. Thus, the workload of non-linear is only $1/M$, but with the SERDES, increased to R/M, along with that the resource is decreased to $1/R$. The trade-off is that when $R > M$, the bandwidth becomes the bottle-neck and the performance is limited. While the effect on the resource is weaker for larger R. We adjust R as 8 for ADAS. Fig. 6 shows the example of the impact of SERDES on non-linear.

C. Utilization Optimization

The FPGAs integrate rich kinds of powerful resources for designers to combine properly. One characteristic is that the resource of ADD3 is equal to ADD2. This inspires us to further reduce the resource of non-linear by migrating the bias step into the accumulation. Thus two ADD2s are

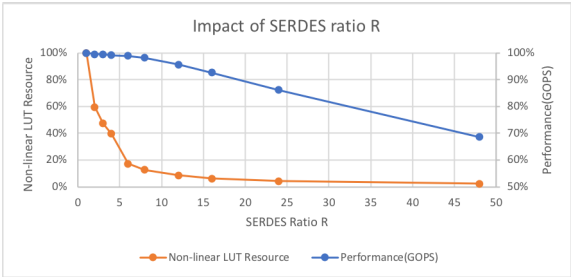


Fig. 6. Impact of SERDES RATIO R

combined into one ADD3. The bias input of ADD3 is non-zero only at the last cycle of the accumulation.

For our ADAS networks built on Xilinx ZU9 FPGA, high-performance DSP slices can be used to replace ADD3 in the accumulate-bias operation. Thus the LUT resources are further reduced.

V. EXPERIMENTAL RESULTS

In this section, two ADAS systems are built and evaluated on Xilinx ZU9 FPGA (peak performance 2.7 TOPs) with our software and hardware optimization. The result is shown in I.

TABLE I
PERFORMANCE OF MULTI-TASK AND MULTIPLE MODEL ON FPGA

Application	Multiple model			Multi-task
	ssd	VPG	FPN	
backbone	VGG	VGG	resnet	resnet18
Computation(GOPS)	117	100	8.75	13.6
Compression Ratio	0.85	0.9	-	-
platform	Xilinx ZU9 FPGA			
FPS	36(each channel)			55
Power(W)	13.7			12.1

The multi-model system consists of several single task algorithms which are designed and tuned respectively: a light-weight SSD detection network with 85% pruned rate, a VPGNet lane detection network with 90% pruned rate, and an FPN-based segmentation network. The average FPS is 36.

The multi-task system uses ResNet-18 as the backbone encoder to extract shared features. The entire computational load is only 13.6G and its FPS reach up to 55.

VI. CONCLUSIONS

A multi-task system is proposed for ADAS, along with its network design, software optimization, and hardware optimization. The system can reach up to 55 FPS on Xilinx ZU9, which meets real-time requirements in ADAS. The work carried out in this paper has revealed many promising areas of further research in the ADAS system optimization field. Further improvements can be achieved by deep model compression, software-hardware collaborative optimization, and so on.

REFERENCES

[1] K. He, X. Zhang, S. Ren and J. Sun, Deep Residual Learning for Image Recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.
[2] C. Szegedy et al., Going deeper with convolutions, 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 1-9.
[3] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-scale Image Recognition, Proc. ICLR, 2015.
[4] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017.
[5] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779-788.
[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A. C. Berg, SSD: Single shot multibox detector, European Conference on Computer Vision, 2016, pp. 21-37.
[7] H. Noh, S. Hong and B. Han, Learning Deconvolution Network for Semantic Segmentation, 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1520-1528.
[8] V. Badrinarayanan, A. Kendall and R. Cipolla, SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation, in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 12, pp. 2481-2495, 1 Dec. 2017.
[9] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation in MICCAI, Springer, 2015, pp. 234-241.
[10] Neven D, Brabandere B D, Georgoulis S, et al. Towards End-to-End Lane Detection: an Instance Segmentation Approach[J]. 2018.
[11] Lee S, Kweon I S, Kim J, et al. VPGNet: Vanishing Point Guided Network for Lane and Road Marking Detection and Recognition[J]. 2017:1965-1973.
[12] Ruder S. An Overview of Multi-Task Learning in Deep Neural Networks[J]. 2017.
[13] Doersch C, Zisserman A. Multi-task Self-Supervised Visual Learning[C]// IEEE International Conference on Computer Vision. IEEE Computer Society, 2017:2070-2079.
[14] J. Qiu, J. Wang, S. Yao, K. Guo, B. Li, E. Zhou, J. Yu, T. Tang, N. Xu, S. Song, Y. Wang, H. Yang, Going Deeper with Embedded FPGA Platform for Convolutional Neural Network , in ACM International Symposium on FPGA, 2016, pp.26-35.
[15] K.Guo,S.Han,S.Yao,Y.Wang,Y.Xie,H.Yang,SoftwareHardware Codesign for Efficient Neural Network Acceleration , in IEEE Micro, vol.37, No.2, 2017, pp.18-25.
[16] K. Guo, L. Sui, J. Qiu, J. Yu, J. Wang, S. Yao, S. Han, Y. Wang, H. Yang, Angel-Eye: A Complete Design Flow for Mapping CNN onto Embedded FPGA , in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), vol.37, No.1, 2018, pp.35-47.
[17] Nunez-Yanez, Jose, et al. "Parallelizing Workload Execution in Embedded and High-Performance Heterogeneous Systems." arXiv preprint arXiv:1802.03316 (2018).
[18] Cordts, Marius, et al. "The cityscapes dataset for semantic urban scene understanding." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[19] Girshick R. Fast R-CNN[J]. Computer Science, 2015.