Journal of Theoretical and Applied Information Technology
15th November 2020. Vol.98. No 21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

REAL-TIME VEHICLE AND PEDESTRIAN DETECTION ON EMBEDDED PLATFORMS

HOANH NGUYEN
Faculty of Electrical Engineering Technology, Industrial University of Ho Chi Minh City, Ho Chi Minh City, Vietnam
E-mail: nguyenhoanh@iuh.edu.vn

ABSTRACT

Real-time pedestrian and vehicle detection on embedded devices play crucial role in many intelligent transport systems because of the limited hardware in autonomous driving devices. This paper presents a lightweight two-stage detector for real-time pedestrian and vehicle detection. The proposed detector includes a lightweight backbone at first stage and a lightweight detection network at second stage. The proposed lightweight backbone is designed based on the ShuffleNetv2 network, which achieves the best accuracy in very limited computational budgets. The proposed lightweight detection network consists of an improved R-CNN to improve the computational cost and a separable convolution module to increase the receptive field. In addition, a lightweight region proposal network is used to improve both accuracy and inference speed of proposals generation stage. The lightweight region proposal network includes pointwise convolution to reduce the number of channels of input features and dilated convolution to enlarge the receptive field. The KITTI dataset is adopted to evaluate the effectiveness of the proposed detector. Experimental results on recent embedded devices, including Raspberry Pi 4 and NVIDIA Jetson TX2, and GPU-based computer show that the proposed method achieves a much better trade-off between accuracy and efficiency compared with recent methods and meets the requirement for real-time object detection on embedded platforms.

Keywords: *Vehicle Detection, Pedestrian Detection, Convolutional Neural Network, Embedded Platforms, Real-time Detection, Lightweight Network*

1. INTRODUCTION

Real-time object detection on embedded devices is a crucial but challenging task in computer vision. Compared with GPU-based computer, embedded devices are computation-constrained and raise more strict restrictions on the computational cost of detectors. However, recent deep CNN-based object detection frameworks are resource-hungry and require massive computation to achieve good detection accuracy, which hinders them from real-time inference in embedded platforms. A deep CNN-based object detection framework usually includes two parts: a backbone network and a detection network. The backbone network first extracts feature from input image, and the detection network locates every object in image. Recent state-of-the-art backbone networks such as ResNet, VGG, Inception, Inception-ResNets usually consist of many convolution layers and adopt large input images, which requires massive computational cost. Recently, researchers have developed lightweight backbone network such as MobileNetv1, MobileNetv2, ShuffleNetv1, ShuffleNetv2 to facilitate real-time object detection. In the detection part, recent state-of-the-art detection frameworks can be divided into two groups: one-stage framework and two-stage framework. Two-stage frameworks usually include a region proposal network (RPN) at first stage for generating object proposals and a detection network at second stage for localizing and classifying objects. Recent state-of-the-art two-stage frameworks such as Faster R-CNN adopted a heavy detection part for better accuracy, but it is too expensive for embedded devices. On the other hand, one-stage frameworks directly predict bounding boxes and class probabilities without generating object proposals. The detection network of one-stage frameworks is usually based on additional layers to generate predictions, which usually requires a small computational cost. For this reason, one-stage frameworks are usually faster than two-stage frameworks. However, as one-stage frameworks do

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

not conduct RoI-wise feature extraction and recognition, their results are coarser than two-stage detectors.

With above research ideals, this paper presents a real-time framework for pedestrian and vehicle detection on embedded devices. The proposed framework is based on two-stage architecture. In the proposed framework, a lightweight backbone network based on ShuffleNetv2 is designed to increase the speed of feature extraction stage. A lightweight region proposal network and a lightweight detection network are designed to improve both detection accuracy and inference speed. Experimental results on Raspberry Pi 4 and NVIDIA Jetson TX2 show that the proposed framework meets the requirement for real-time object detection on embedded platforms.

The remaining of this paper is organized as follows. Section 2 introduces the related work. Section 3 details the proposed framework. Section 4 provides the experimental results and comparison between the proposed method and other methods on public datasets. Finally, the conclusions and future works is drawn in Section 5.

2. RELATED WORK

2.1 Light Weight Deep CNN Architecture

Recently, researchers have developed lightweight backbone network to facilitate real-time processing systems. Most state-of-the-art efficient backbone networks [2, 3, 5] use depth-wise separable convolutions [2]. Depth-wise separable convolutions factor a convolution into two stages to reduce computational complexity: depth-wise convolution and pointwise convolution. Depth-wise convolution performs light-weight filtering by applying a single convolutional kernel per input channel, and pointwise convolution usually expands the feature map along channels by learning linear combinations of the input channels. Another group of lightweight backbone networks [16, 17] adopts group convolution [18], where input channels and convolutional kernels are factored into groups and each group is convolved independently. In addition to convolutional factorization, a network's efficiency and accuracy can be further improved using methods such as channel shuffle and channel split [5]. Another approach to improve inference of a pre-trained network is low-bit representation of network weights using quantization [19, 20]. These approaches use fewer bits to represent weights of a pre-trained network instead of 32-bit high-precision floating points. For lightweight detection network, it is common that one-stage detectors are regarded as the key to real-time detection. For instance, YOLO/YOLOv2 [14] and SSD [13] run in real time on GPU. When coupled with small backbone networks, lightweight one-stage detectors, such as MobileNet-SSD [2], MobileNetV2-SSDLite [3], Pelee [21] and Tiny-DSOD [22], achieve inference on mobile devices at low frame rates. For two-stage detectors, Light-Head R-CNN [6] utilizes a light detection head and runs at over 100 fps on GPU. Light-Head R-CNN proposed a light-head design to build an efficient yet accurate two-stage detector. Specifically, a large-kernel separable convolution was applied to produce "thin" feature maps with small channel number. This design greatly reduces the computation of following RoI-wise subnetwork and makes the detection system memory-friendly. A cheap single fully connected layer is attached to the pooling layer, which well exploits the feature representation for classification and regression.

2.2 Pedestrian and Vehicle Detection

The huge success of deep learning and CNN technologies significantly boost research and development of autonomous driving. The popular models are applied and enhanced for object detection in driving environments, including pedestrian and vehicle. However, the popular models including Faster-RCNN, SSD, YOLO, YOLOv2 did not produce good detection accuracy results over the KITTI test dataset. But with certain modifications and adaptations, the variants of Faster-RCNN and SSD models are taking the top entries in the KITTI object detection leader board. For example, [23] improved the region proposal quality with resource to subcategory information. [24] presented an improved framework for vehicle detection based on deconvolutional modules and multi-layer region proposal network. As it is hard for Faster-RCNN to handle the large object size variation, which is designed to detect all the objects on a single layer, MS-CNN [15] extends the detection over multiple scales of feature layers, which produces good detection performance improvement. Scale dependent pooling and cascaded rejection classifiers are used in [25]. In [26], authors propose a recurrent rolling convolution architecture on top of SSD model, which produces top detection performance for pedestrian detection. However, the RRC model is very complex and significantly increases computation time.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. PROPOSED FRAMEWORK

Table 1: The Architecture of The Original ShuffleNetv2 [5].

Layer	Type	Kernel Size	Stride	Repeat	Output Size	Output Channel
0	Conv1	3×3	2	1	112×112	24
1	MaxPool	3×3	2	1	56×56	24
2	Shuffle Unit	3×3 depthwise convolution layers	2 1	1 3	28×28	176
3	Shuffle Unit	3×3 depthwise convolution layers	2 1	1 7	14×14	352
4	Shuffle Unit	3×3 depthwise convolution layers	2 1	1 3	7×7	704
5	Conv5	1×1	1	1	7×7	1024
6	GlobalPool	7×7			1×1	
7	FC					1000

Table 2: The Final Backbone Network Architecture Used in This Paper.

Layer	Type	Kernel Size	Stride	Repeat	Output Size	Output Channel
0	Conv1	3×3	2	1	112×112	24
1	MaxPool	3×3	2	1	56×56	24
2	Shuffle Unit	5×5 depthwise convolution layers	2 1	1 3	28×28	60
3	Shuffle Unit	5×5 depthwise convolution layers	2 1	1 7	14×14	120
4	Shuffle Unit	5×5 depthwise convolution layers	2 1	1 3	7×7	240
5	Conv5	1×1	1	1	7×7	512

3.1 Backbone Network

The backbone network in deep CNN frameworks extracts features from the input image, thus having a great influence on both accuracy and efficiency of the whole framework. Specifically, a lightweight deep CNN backbone will improve the inference speed of the whole framework. Recent deep CNN-based object detectors adopt pre-trained classification network on ImageNet [1] as the backbone network. However, as classification and object detection conduct different tasks on the backbone network, simply using pre-trained classification backbone for object detection task does not achieve good results. In the backbone network, low-level feature maps at shallow layers have higher resolution. However, high-level feature maps at deep layers contain more discriminative information. The high-resolution feature maps facilitate the localization subtask, and the high-level feature maps enhance the classification subtask. For the localization subtask, the receptive field in deep CNN layers plays a crucial role. Each layer of a deep CNN backbone can capture only information inside receptive field. Thus, a large receptive field can leverage more context information, which enhances the performance of the localization subtask, especially for large objects. Recent lightweight deep CNN backbones such as MobileNetv1 [2], MobileNetv2 [3], ShuffleNetv1 [4], and ShuffleNetv2 [5] contain a fixed receptive field in each layer, thus decreasing the performance of localization subtask in object detection framework. With above insights, this paper designs a lightweight deep CNN backbone network based on ShuffleNetv2 [5] for real-time vehicle and pedestrian detection on mobile and embedded devices. The architecture of the original ShuffleNetv2 is shown in Table 1. ShuffleNetv2 is a lightweight deep CNN network which achieves the best accuracy in very limited computational budgets. By shuffling the channels, ShuffleNetv2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

outperformed MobileNetV1, MobileNetv2, and ShuffleNetv1. Based on the ShuffleNetv2, this paper first replaces all 3×3 depthwise convolution layers in ShuffleNetv2 by 5×5 depthwise convolution layers.

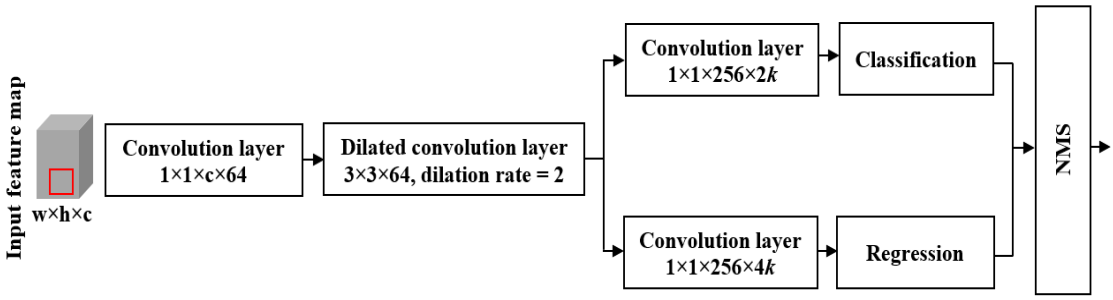

Figure 1: The Structure of Lightweight RPN for Generating Object Proposals Used in This Paper.

By using 5×5 depthwise convolution layers, the receptive field is enlarged to capture more semantic information, while providing similar computational budget to 3×3 convolution layers. Next, to balance information between low-level feature maps and high-level feature maps, this paper uses a 1×1 convolution layer to compress the number of channels of Conv5 layer in ShuffleNetv2 to 512 channels and increases the number of channels in shallow layers at the same time. This method can effectively balance the feature information between low-level and high-level feature, thus improving both classification subtask and localization subtask. In addition, by modifying the number of channels between convolution layers, there is no additional computational cost adding in new network. The final backbone network architecture used in this paper is presented in Table 2.

3.2 Lightweight Region Proposal Network

Regular two-stage object detectors usually adopt region proposal network (RPN) at first stage for generating object proposals. At second stage, a heavy detection head including full connected layers is used to further classify and regress object proposals. Light-Head R-CNN [6] adopted a lightweight detection head to improve the computational cost of the network. However, the proposed detection network in Light-Head R-CNN is still too complicated when coupled with lightweight backbone networks. In addition, Light-Head R-CNN adopted imbalance features between the backbone and the detection part. This imbalance not only leads to redundant computation but increases the risk of overfitting. To address this issue, this paper designs a lightweight RPN for generating object proposals. Figure 1 illustrates the structure of the proposed lightweight RPN. First, a 1×1 convolution layer is used after input feature maps to reduce the number of channels of input feature maps. Reducing the number of channels of input features can further reduce the number of parameters in the subsequent convolutional layers, thus improving the inference speed. Then, 3×3 dilated convolution with dilation rate at 2 is adopted to replace standard 3×3 convolution in the original RPN. Dilated convolution is used to enlarge the receptive field, thus including more information from other areas to help recognize the boundaries of objects. By using dilated convolution, the number of parameters does not increase while including more context information. For scales and aspect ratios, this paper sets three aspect ratios [1:2, 1:1, 2:1] and five scales [32×32; 64×64; 128×128; 256×256; 512×512] to cover vehicle and pedestrian of different shapes. Since there are many proposals heavily overlapping with each other, non-maximum suppression (NMS) is used to reduce the number of proposals. This paper sets the intersection over-union (IoU) threshold of 0.7 and 0.5 for vehicle and pedestrian respectively. Training labels of anchor boxes are designed based on their IoU ratios with ground-truth bounding boxes. If the anchor has IoU over 0.7/0.5 with any ground-truth box, it will be set a positive label. Anchors which have highest IoU for ground-truth box will also be assigned a positive label. Meanwhile, if extra anchors have IoU less than 0.3 with all ground-truth box, their labels will be negative.

3.3 Light Detection Network

Faster R-CNN [7] adopts R-CNN [8], which utilized two large fully connected layers, as a second stage classifier which is beneficial to the detection performance. The R-CNN adopted at second stage enhances the detection performance of Faster R-CNN and its extensions. However, the computational cost of the R-CNN is intensive,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

especially when the number of object proposals generated by the RPN is large. Follow Light-Head R-CNN [6], this paper adopts the Light-Head R-CNN as shown in Figure 2 to improve the

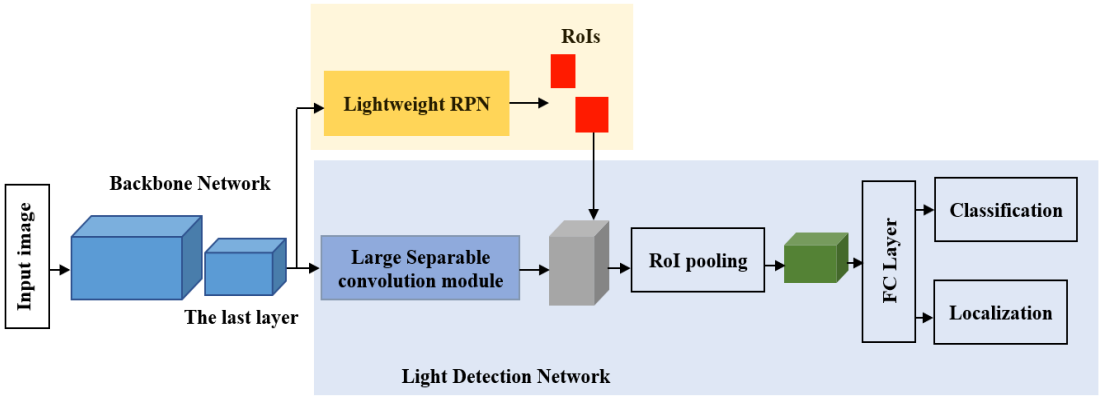

Figure 2: The Detection Network Used in This Paper.

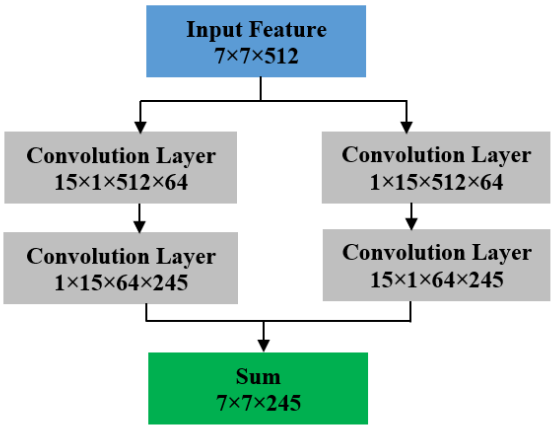

Figure 3: The Structure of Large Separable Convolution Module Used in This Paper.

computational cost. As shown, a large separable convolution module [9] whose structure is shown in Figure 3 is used after the final layer of the backbone network to generate a thin feature map with $\alpha \times \beta \times \beta$ channels before RoI pooling layer, where $\beta = 7$ is the pooling size, and α is a reduced factor. In this paper, α is set to 5 instead of 10 in Light-Head R-CNN to eliminate redundant computation. The separable convolution module significantly increases the receptive field. For the R-CNN subnet, a single fully connected layer with 1024 channels is used in R-CNN subnet to further reduce the computational cost of R-CNN subnet without sacrificing accuracy, followed by two sibling fully connected layers to predict RoI classification and regression.

4. EXPERIMENTAL RESULTS

4.1 Embedded Platforms for Real Time Pedestrian and Vehicle Detection

This paper uses recent embedded devices for evaluating the performance of the proposed method on real-time pedestrian and vehicle detection, including Raspberry Pi 4 [10] and Jetson TX2 [11]. In addition, this paper conducts experiments on NVIDIA GTX 1080 GPU to compare the detection performance of the proposed method with recent methods on pedestrian and vehicle detection.

Raspberry Pi 4

Raspberry Pi 4 (Figure 4) is the latest, cheapest, and most flexible tiny product in the popular Raspberry Pi range of computers. Raspberry Pi 4

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

provides a great improvement in processor speed, multimedia performance, memory, and connectivity compared with recent Raspberry Pi 3 Model B+. The key features of the Raspberry Pi 4 module include a

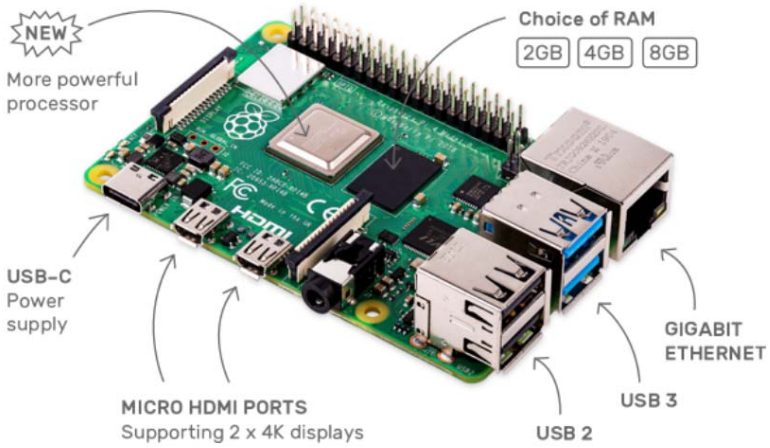

Figure 4: Raspberry Pi 4 [10].

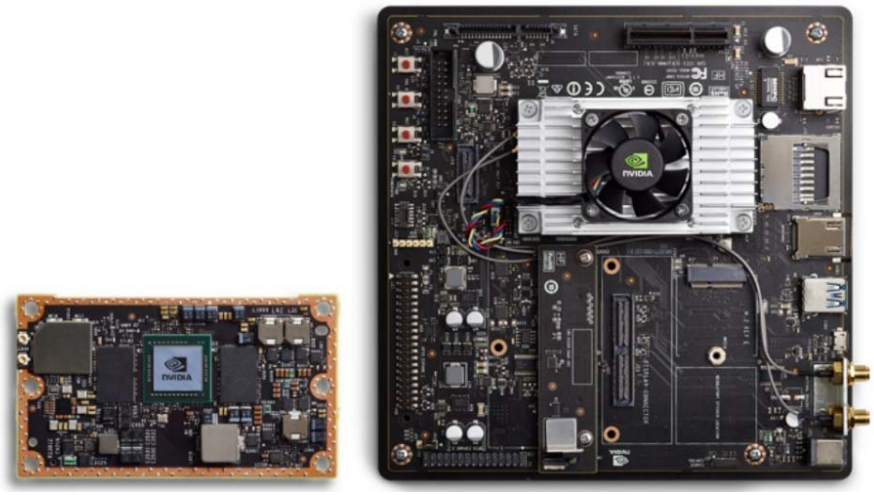

Figure 5: Jetson TX2 [11].

high-performance Broadcom BCM2711, a quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz, a pair of micro-HDMI ports which are used to connect dual displays with 4k resolution, H.265 and H.264 hardware video decoding (maximally supporting up to 4Kp60), 8GB RAM, dual-band 2.4/5.0 GHz IEEE 802.11ac wireless (wireless LAN), OpenGL ES, 3.0 graphics, Bluetooth 5.0, Bluetooth Low Energy (BLE), standard 40-pin GPIO, Gigabit Ethernet (2 × USB 2.0 ports and 3.0 ports), a micro SD card slot for loading OS and data storage, operating temperature 0^0C-50^0C, and power over Ethernet (PoE) enabled (requires separate PoE HAT).

NVIDIA Jetson TX2

NVIDIA Jetson TX2 (Figure 5) is one of the fastest and most power efficient embedded devices. Jetson TX2 provides an easy environment to deploy hardware and software for real-time object detection. Jetson TX2 supports NVIDIA Jetpack on a software development kit (SDK) which includes a board support package (BSP), deep learning libraries, computer vision applications, GPU computational power, and image and video processing. Jetson TX2 features include dual-core NVIDIA Denver2 + quad-core ARM Cortex-A57, an integrated 256-core NVIDIA Pascal GPU, 8GB with 128-bit interface DDR4 internal memory, 32 GB external memory card, 4kp60 H.264/ H.265 encoder, and a decoder. Jetson TX2 supports 10/100/1000 BASE-T Ethernet, HDMI 2.0, M.2 Key E, SD, GPIOs, I2C, I2S, SPI, and Dual CAN bus, and provides on-chip TTL UART. Jetson TX2 provides USB 3.0, USB 2.0, and micro USB. Jetson TX2 is useful for developing applications in the field of computer vision and deep

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

learning, since it runs with open-source Linux OS and performs calculations over one teraflop.

Figure 6: Images in The KITTI Dataset [12].

4.2 Dataset and Evaluation Metrics

In order to evaluate the effectiveness of the proposed approach on real-time pedestrian and vehicle detection on embedded devices, this paper conducts experiments on widely used public dataset: KITTI dataset [12]. KITTI dataset is a widely used dataset for evaluating vehicle and pedestrian detection algorithms. This dataset consists of 7481 images for training with available ground-truth and 7518 images for testing with no available ground-truth. Images in this dataset include various scales of vehicle and pedestrian in different scenes and conditions and were divided into three difficulty-level groups: easy, moderate, and hard as shown in Figure 6. If the bounding boxes size was larger than 40 pixels, a completely unshielded vehicle/pedestrian was considered to be an easy object, if the bounding boxes size was larger than 25 pixels but smaller than 40 pixels, a partially shielded vehicle/pedestrian was considered as a moderate object, and an vehicle/pedestrian with the bounding boxes size smaller than 25 pixels and an invisible vehicle/pedestrian that was difficult to see with the naked eye were considered as hard objects.

For evaluation metrics, this paper uses the average precision (AP) and intersection over union (IoU) metrics [12] to evaluate the performance of the proposed method in all three difficulty level groups of the KITTI dataset. These criteria have been used to assess various object detection algorithms. As in [12], the IoU is set to 0.7 for vehicle and 0.5 for pedestrian in this paper, which means only the overlap between the detected bounding box and the ground truth bounding box greater than or equal to 70% and 50% is considered as a correct detection.

4.3 Experimental Results

This section presents the detection results of the proposed method and recent methods on the KITTI dataset. First, this paper conducts experiments on the KITTI test set by using the proposed model and recent models to compare the detection performance. The reference models include SSD [13], Faster R-CNN [7], YOLOv2 [14], and MS-CNN [15]. All models are implemented on NVIDIA GTX 1080 GPU. Table 3 presents the detection results of the proposed model and reference models on all three difficulty-level groups of the KITTI test set. As shown in Table 3, the proposed model obtains 88.46%, 84.31%, and 73.12% of the AP on easy, moderate, and hard group respectively for vehicle detection. For pedestrian detection, the proposed model obtains 84.12%, 74.98%, and 63.48% of the AP on easy, moderate, and hard group respectively. It can be observed that the proposed model achieves superior results to state-of-the-art object detectors, including both one-stage and two-stage detectors such as Faster R-CNN, SSD, and YOLOv2. Compared with MS-CNN, the proposed method achieves competitive results. However, MS-CNN adopted multi-scale features at different layers for object detection, thus increasing the computational cost. Specially, the proposed method outperforms all reference methods on the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

inference speed. More specific, the proposed method takes 0.04-0.06 second for processing an image. This result demonstrates that the proposed method

Table 3: Detection Results on All Three Difficulty-Level Groups of The KITTI Test Set.

Method	AP (%)						Time (s)
	Vehicle			Pedestrian			
	Easy	Moderate	Hard	Easy	Moderate	Hard	
Faster R-CNN [5]	86.71	81.84	71.12	76.21	62.14	60.33	2.5-3.6
SSD [11]	77.71	64.06	56.17	25.12	18.20	16.21	0.12-0.33
YOLOv2 [12]	76.79	61.31	50.25	22.16	16.16	15.82	0.08-0.16
MS-CNN [13]	90.03	89.02	76.11	84.12	74.98	63.48	0.6-0.8
Proposed Model	88.46	84.31	73.12	80.63	72.18	64.02	0.04-0.06

Table 4: The Performance Results in Terms of Model Complexity, Computational Complexity, and Inference Speed.

Method	Backbone	Model Size (MB)	GFLOPs	FPS	
				Raspberry Pi 4	Jetson TX2
SSD	MobileNetv1	35	1.2	3-5	7-9
Faster R-CNN	VGG-16	520	150	0.1	0.8
Proposed Model	ShuffleNetv2	30	1.12	4-6	10-12

achieves a much better trade-off between accuracy and efficiency.

Next, to evaluate the effectiveness of the proposed method on real-time detection based on embedded devices, this paper conducts experiments on Raspberry Pi 4 and NVIDIA Jetson TX2. For the model complexity, this paper analyzes from the total learnable parameter that each model had. The model complexity itself has two values, which comes from the total number of parameter and the model size in terms of MB. For the computational complexity, total floating-point operations (GFLOPs) are used as the computational complexity value. In addition, the FPS of each model is calculated on both Raspberry Pi 4 and NVIDIA Jetson TX2 to compare the inference speed. Table 4 shows the performance results of the proposed model and reference models in terms of model complexity, computational complexity, and inference speed. As shown, the proposed model for vehicle and pedestrian detection running on NVIDIA Jetson TX2 and Raspberry Pi 4 achieves inference speed of 10-12 fps and 4-6 fps respectively. This result shows that the proposed method meets the requirement for real-time object detection on embedded platforms. It can be observed that the proposed model surpasses both SSD and Faster R-CNN framework in terms of model complexity, computational complexity, and inference speed. Figure 7 visualizes several examples of detection results of the proposed method on the KITTI test set.

5. CONCLUSIONS

This paper presents a real-time framework for pedestrian and vehicle detection on embedded devices. The proposed framework is based on two-stage architecture. In the proposed framework, a lightweight backbone network based on ShuffleNetv2 is designed to increase the speed of feature extraction stage. A lightweight region proposal network including pointwise convolution and dilated convolution is designed to enlarge the receptive field. Furthermore, a lightweight detection network including an improved R-CNN to improve the computational cost and a separable convolution module to increase the receptive field is presented to improve both detection accuracy and inference speed. Experimental results on Raspberry Pi 4, NVIDIA Jetson TX2, and GPU-based computer show that the proposed framework meets the requirement for real-time object detection on embedded platforms and achieves a much better

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

trade-off between accuracy and efficiency compared with recent methods.

Figure 7: Examples of Detection Results of The Proposed Method on The KITTI Test Set.

REFERENCES:

[1] Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang et al. "Imagenet large scale visual recognition challenge." *International journal of computer vision* 115, no. 3 (2015): 211-252.

[2] Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[3] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. "Mobilenetv2: Inverted residuals and linear bottlenecks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510-4520. 2018.

[4] Zhang, Xiangyu, Xinyu Zhou, Mengxiao Lin, and Jian Sun. "Shufflenet: An extremely efficient convolutional neural network for mobile devices." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848-6856. 2018.

[5] Ma, Ningning, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. "Shufflenet v2: Practical guidelines for efficient cnn architecture design." In *Proceedings of the European conference on computer vision (ECCV)*, pp. 116-131. 2018.

[6] Li, Zeming, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. "Light-head r-cnn: In defense of two-stage object detector." *arXiv preprint arXiv:1711.07264* (2017).

[7] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91-99. 2015.

[8] Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580-587. 2014.

[9] Peng, Chao, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. "Large kernel matters--improve semantic segmentation by global convolutional network." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4353-4361. 2017.

[10] RaspberryPI. Available online: https://www.raspberrypi.org/ (accessed on 31 December 2019).

[11] JetsonTX2. Available online: https://elinux.org/Jetson_TX2 (accessed on 31 December 2019).

[12] Geiger, Andreas, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite." In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354-3361. IEEE, 2012.

[13] Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "Ssd: Single shot multibox detector." In *European conference on computer vision*, pp. 21-37. Springer, Cham, 2016.

[14] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263-7271. 2017.

[15] Cai, Zhaowei, Quanfu Fan, Rogerio S. Feris, and Nuno Vasconcelos. "A unified multi-scale deep convolutional neural network for fast object detection." In *European conference on computer vision*, pp. 354-370. Springer, Cham, 2016.

[16] Huang, Gao, Shichen Liu, Laurens Van der Maaten, and Kilian Q. Weinberger. "Condensenet: An efficient densenet using learned group convolutions." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2752-2761. 2018.

[17] Zhao, Hengshuang, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. "Icnet for real-time semantic segmentation on high-resolution images." In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 405-420. 2018.

[18] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." In *Advances in neural information processing systems*, pp. 1097-1105. 2012.

[19] Andri, Renzo, Lukas Cavigelli, Davide Rossi, and Luca Benini. "YodaNN: An architecture for ultralow power binary-weight CNN acceleration." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37, no. 1 (2017): 48-60.

[20] Courbariaux, Matthieu, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." *arXiv preprint arXiv:1602.02830* (2016).

[21] Wang, Robert J., Xiang Li, and Charles X. Ling. "Pelee: A real-time object detection system on mobile devices." In *Advances in*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Neural Information Processing Systems, pp. 1963-1972. 2018.

[22] Li, Yuxi, Jiuwei Li, Weiyao Lin, and Jianguo Li. "Tiny-DSOD: Lightweight object detection for resource-restricted usages." *arXiv preprint arXiv:1807.11013* (2018).

[23] Xiang, Yu, Wongun Choi, Yuanqing Lin, and Silvio Savarese. "Subcategory-aware convolutional neural networks for object proposals and detection." In *2017 IEEE winter conference on applications of computer vision (WACV)*, pp. 924-933. IEEE, 2017.

[24] NGUYEN, HOANH. "ENHANCED VEHICLE DETECTION APPROACH USING DEEP CONVOLUTIONAL NEURAL NETWORKS." *Journal of Theoretical and Applied Information Technology* 97, no. 23 (2019).

[25] Yang, Fan, Wongun Choi, and Yuanqing Lin. "Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2129-2137. 2016.

[26] Ren, Jimmy, Xiaohao Chen, Jianbo Liu, Wenxiu Sun, Jiahao Pang, Qiong Yan, Yu-Wing Tai, and Li Xu. "Accurate single stage detector using recurrent rolling convolution." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5420-5428. 2017.