2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Object Detection Using a Single Extended Feature Map

Young-Chul Lim and Minsung Kang

Abstract—**Fully convolutional neural network-based object detectors have achieved considerable detection accuracy in recent years. It is a recent trend to establish complex and deep network architectures for improvement of the detection accuracy. However, object detectors for intelligent vehicle applications require fast inference speed, lightweight network architecture, and less memory usage as well as high detection accuracy to implement the algorithm in an embedded hardware. In this paper, we propose a fast object detection method based on a single stage and a single extended feature map. A lightweight network based on an extended paththrough layer is proposed to improve both the accuracy and speed. The extended paththrough layer enlarges the resolution of the last feature map by concatenating later feature maps with lower resolution to earlier feature map maps with higher resolution. The layer helps to search and detect smaller objects more densely on the extended last feature map. Our experimental results show that the proposed detection model outperforms the previous state-of-the-art methods in both detection accuracy and inference speed.**

I. INTRODUCTION

In a safe autonomous driving system (ADS), the most important task is to perceive the external environment on the road. Autonomous vehicles must accurately and reliably detect objects on the road with fast inference speed: cars, pedestrians, cyclists, road signs, and other objects. The ADS obtains and combines reliable information from external sensors, such as radar [1], lidar [2], and cameras [3], and automatically manipulates the controls, such as the steering and brake systems. In these systems, the camera sensors are essential to recognize categories of various objects in dynamic road environments. The camera-based perception systems are required to accurately and reliably classify and localize different kinds of objects in the ADS.

Until a few years ago, the object detection task was considered a highly challenging problem due to complex environmental conditions, such as occlusion, blurring, appearance change, different viewpoints, bad illumination, etc. Conventional object detection methods, such as Viola and Jones (VJ) [4], the histogram of oriented gradients (HOG) [5], and aggregate channel features (ACF) [6], have shown promising accuracy in a relatively less complex environment. The detection performance deteriorates in highly dynamic and complex external environments because they learn only low-level features in the training samples for detecting objects. The advent of deep learning technology has brought breakthrough improvement in object classification and detection tasks. Deep learning-based object classifiers learn high-level features through deeply stacked convolutional and pooling networks [7]. Recently, deep learning-based detectors have already achieved leading performance in most detection benchmarks [8-10] based on an object classification backbone architecture. However, there are still many challenges to applying such systems to a dynamic road environment while ensuring both real-time inference and high detection confidence.

For accuracy, convolutional neural networks (CNN)-based object detectors have been outperforming previous hand-craft feature-based detectors since three or four years ago. Recently, a fully convolutional network (FCN)-based object detector was proposed to reduce the number of parameters and computation power [11-13]. Recent FCN-based object detectors have two structures: a two-stage detector and a single-stage detector. The early deep learning-based object detector Regional CNN (R-CNN) [14] was the beginning of the two-stage architecture, which is divided into a region proposal stage and an object classification stage. The candidate regions were proposed in the region proposal stage using selective search [15], and then final classifications and bounding box regressions were executed in the second classification stage. Over a few years, many descendants [12, 16, 17] of the R-CNN have achieved improvement in accuracy and learning/inference speed. The two-stage architecture has become a more and more popular approach in general object detection benchmarks [8, 9].

On the other hand, a single-stage detector directly predicts the bounding boxes and classes of target objects without a region proposal network. The single-shot multi-box detector (SSD) [11] and you only look once (YOLO) [13, 21] are representative methods in single-stage detectors. These methods classify and localize objects on regularly sampled anchor boxes that densely cover spatial positions, scales, and aspect ratios. These methods are less accurate than the latest two-stage detectors. Most of the low quality bounding boxes come from failure localization of either small objects or overlapping objects. However, with the single-stage detectors it is usually easier to train the network and it is more computationally efficient in inference. The single-stage detectors are a more suitable structure in intelligent vehicle applications where the algorithm needs to be implemented on an embedded platform.

Embedded platforms for automotive applications need to consume much smaller power and have smaller energy dissipation than desktop and rack systems, which have powerful GPUs. Significantly smaller and simpler networks are necessary for fast inference speed, production cost, and more feasible embedded system deployment. SqueezeDet [18] simultaneously guaranteed real-time inference speed as well as a small network size and low energy consumption for

* This work was supported by the DGIST R&D Program of the Ministry of Science, ICT.

Y. C. Lim, and Ms. Kang are all with Convergence Research Center for Future Automotive Technology, Daegu Gyeongbuk Institute of Science & Technology, Room 309, 3nd floor, 4nd research center, 333, Techno Jungang Daero, Hyeonpung-Myeon, Dalseong-gun, Daegu, 711-873, Republic of Korea (e-mail : ninolyc@dgist.ac.kr).

embedded system deployment. This model only contains a single forward pass that extracts feature maps and also regresses multiple bounding boxes and computes their class probabilities. In [19], a low-complexity object detector (LCDet) using an 8-bit quantization model performed as good as a detection network using floating point weights for object detection tasks. The quantization model is advantageous for implementation in DSPs or dedicated hardware accelerators.

In this paper, we propose a single extended feature map-based object detection method for improving both speed and accuracy. We shrink the detection network drastically for memory reduction and computational efficiency. In order to handle an inaccurate localization problem for small objects, an extended paththrough (EPT) layer is added to our extended network (ExtendNet) architecture. The rest of the paper is organized as follows. Some related studies are introduced in Section 2. In Section 3, our detection method based on the EPT layer is described in detail. Experimental results and analyses on public datasets are presented in Section 4. Finally, Section 5 provides the conclusion and future work.

II. RELATED WORKS

A. Two-stage detector

Recently, many state-of-the-art detectors have used a two-stage architecture since the success of region proposal methods and region-based CNN classification methods. The R-CNN [14] developed in the early days was computationally expensive and very slow because it repeatedly extracted features from all object proposal regions without computational sharing. The Fast R-CNN [17] accelerated the R-CNN by 10 to 100 times at test time and three times faster at training time by using spatial pyramid pooling networks (SPPnets) [20]. However, the selective search algorithm [15] for region proposals became a bottleneck in the Fast R-CNN. In the Faster R-CNN [16], the region proposal network (RPN) shares full-image convolutional features with the detection network to enable cost-free computation for the region proposals. The R-FCN [12] has region-based and fully convolutional networks for two-stage object detection. The method first generates candidate regions based on objectness scores for pre-defined anchor boxes and then ROI pooling is executed to aggregate position-sensitivity scores to classify boxes and regression maps to refine the box coordinates.

B. Single-stage detector

Single-stage detectors simultaneously classify and regress anchor boxes that are densely sampled with multiple scales and different aspect ratios in each grid on feature maps. Recently, SSD [11] and YOLOv2 [13] demonstrated competitive accuracy and fast inference speed with a single forward stage. SSD has a single-stage model based on multi-scale feature maps. The feed-forward process was executed to localize objects on the multi-scale feature maps that cover objects within a specified scale range. SSD repeatedly predicts objects at hierarchical multi-scale feature maps, which improves both detection accuracy and inference speed. YOLO [21] is known as the first CNN-based general object detector with real-time inference. However, YOLO experienced relatively low recall and precision performance. YOLOv2 improved the original YOLO detector by adopting several kinds of techniques, such as batch normalization, high resolution classifiers, convolution with anchor boxes, dimension clusters, direct location prediction, fine-grained features, and multi-scale training. YOLOv2 achieved reasonable performance with a much faster inference speed compared with SSD and faster R-CNN.

C. Shortcut connections

The depth of CNNs has been getting deeper and deeper since the network depth was reported to be crucial for accuracy improvement of visual recognition tasks [22]. However, it is more difficult to train such networks as depth increases due to gradient vanishing/exploding problems. Recently, Highway networks [23] and Residual networks (ResNets) [24] bypassed feature information from an earlier layer to the next higher layer via shortcut connections to handle these problems. In Highway networks, adaptive gating units inspired by the Long Short-Term memory (LSTM) [25] network control information flow across each layer much easier [23]. However, the gating units have parameters and non-residual functions. In [24], the residual network was formulated so that some stacked layers directly connected desired underlying mapping, in which it was easier to optimize the residual mapping than to optimize the original, unreferenced mapping. Instead of stacking many more layers to get rich and representative feature information, the dense convolutional network (DenseNet) exploited a condensed network architecture for parameter efficiency and information flow improvement [26]. The DenseNet connects all layers in a dense block with every other layer to maximize information flow and establish a shorter path between the layers [26]. Unlike element-wise adding in [23], the DenseNet concatenated each layer with same size, which increased the diversity of the input feature maps.

D. Paththrough layers

Recent fast object detectors [13, 18, 19, 21] predict the coordinates of bounding boxes and class probabilities directly on a single feature map. This is not a problem with large objects, but may causes problems when there are localization errors for smaller objects. In YOLOv2 [13], they simply added a paththrough (PT) layer to a plain network architecture that consists of only convolutional and pooling layers. The PT layer brings fine grained features from an earlier layer with higher resolution. The fine grained feature channels are resized to the same size as the current feature map by resampling and reorganizing with four adjacent feature channels, and then two feature maps are concatenated into the size of the current feature map (Figure. 2(a)). A final feature map is established through several convolutional operations. In this approach, there is a limit to the amount of improvement in the detection accuracy for smaller objects, because the resolution of the final feature map is unchanged.

Feature pyramid networks (FPN) exploited a top-down architecture with lateral connections for establishing high-level feature maps at each scale. The networks combine higher-level features with lower resolution into lower-level features with higher resolution through a top-down pathway and lateral connections. This approach is conceptually similar to the PT layer used in [13]. The top-down pathway resizes higher resolution features by upsampling spatial resolution by a factor of 2. The features become spatially coarser but semantically stronger. The feature information is then

enriched when it is merged with the corresponding bottom-up feature map by element-wise adding via lateral connections. This approach predicts bounding boxes and class probabilities on several multi-scale feature maps. However, this architecture increases the inference time and complexity.

III. EXTEND NETWORK

A. Lightweight network architecture

Our detector has a single-stage network model that is based on YOLOv2 architecture [13]. Our network shrinks the YOLOv2 network architecture by reducing the number of channels in each convolutional layer. To compensate for the degradation of the detection performance, the pooling layer is delayed to have a larger activation map on the each convolution layer. Our network consists of 24 convolutional layers, five pooling layers, and one EPT layer, as shown in Figure 1. In recent studies [13, 18, 19, 21], 3×3 and 1×1 convolution filters were used together to reduce the number of parameters. YOLO architecture [13, 21] used a 1×1 convolution filter to reduce the number of channels of the feature map, which established a feature map with implicit representation. In our network model, only 3×3 convolution filters were used to preserve a reduced network capacity caused by a decrease in the number of channels in our network model. In [13], the size of the last feature map became 1/32 of the size of the input image due to five pooling layers with stride 2. However, our network generates a feature map with a size 1/16 of the input image with the help of the EPT layer. A larger feature map improves detection accuracy for smaller objects [11, 12, 13, 18].

B. Extended paththrough layer

The paththrough layer was proposed in the YOLOv2 model [13] to handle an inaccurate localization problem for smaller objects and overlapped objects while combining finer grained features from lower-level features with higher resolution (Figure 2(a)). However, the size of the combined feature map is same as that of the previous feature map, and it does not help enough to improve the localization accuracy. In this work, we propose an EPT layer to lead to better localization accuracy for smaller objects. In contrast to the network with the PT layer, our approach extends the resolution of a higher-level feature map and concatenates the extended feature map with an earlier feature map with higher resolution. This enlarges the resolution of the previous feature map and creates more fluent feature information. Extended processing enlarges the current feature maps by resampling and reorganizing adjacent feature maps (Figure 2(b)). The resolution of the extended feature maps is twice the size of the original feature maps, but the number of channels is reduced (see Figure 2(b)) by four times. This channel reduction decreases the number of parameters, whereas the number of both channels and parameters increase in the PT layer.

C. Region layer

Our detection model simultaneously performs both region proposals and classifications on a single feature map called the region layer. The region layer has uniformly distributed spatial grids which are the pixels on the last feature map with $W_f \times H_f$ resolution. Several anchor boxes are initially set at each grid. The total number of anchor boxes N_t becomes $W_f \times H_f \times N_b$ where N_b indicates the number of anchor boxes per grid. In the region layer, the channels per anchor box consist of the objectness score, the coordinates of the bounding box, and the classification probabilities, as shown in Figure 3. The total number of channels N becomes $(1+4+N_C+1) \times N_b$ where N_C is the number of object categories. The objectness score channel represents the class-agnostic probability that measures the membership to all the object classes against the background. The channels of bounding box coordinates regress the multi-scale anchor boxes at each grid. Unlike in [13], our approach computes the background probability as well as the class probabilities of each object through a softmax activation function. In [13], an erroneous objectness score may cause a problem in which a false positive with the highest class probability is selected because the softmax function must determine one of the candidate objects. Our approach handles

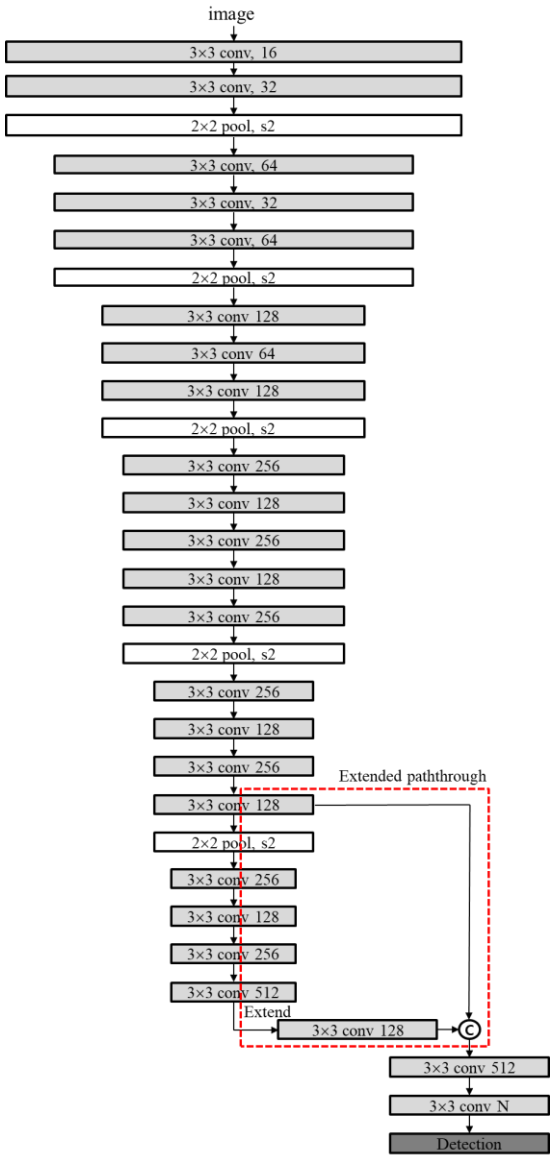

Figure 1. Our netwrok architecture.

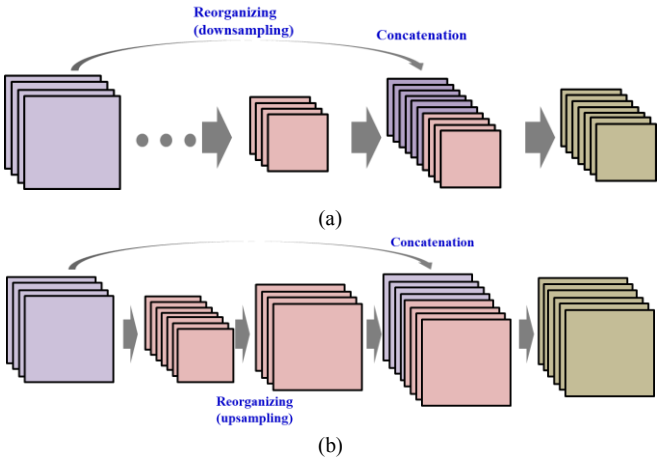

Figure 2. Paththrough layer (a) and extended paththrough layer (b).

this problem by including an additional background class to object classes in the softmax function.

D. Training and inference

Loss function An objective function with multi-task loss should be minimized through a backpropagation algorithm to train a deep network. Our objective function is represented as a weighted sum of objectness loss (L_{ob}), localization loss (L_{loc}), and classification loss (L_{cls}).

$$L\left(\hat{o}_i, \hat{b}_i, \hat{c}_i\right) = \lambda_1 \sum_i L_{ob}\left(\hat{o}_i, o_i\right) + \lambda_2 \sum_i L_{loc}\left(\hat{b}_i, b_i\right) + \lambda_3 \sum_i L_{cls}\left(\hat{c}_i, c_i\right), \quad (1)$$

where i is the index of an anchor box in the region layer. The objectness scores $\hat{o}_i$ and o_i denote the predicted and ground truth scores, respectively, and b_i and c_i indicate the bounding box coordinate and class probability. The three loss functions are normalized with the balancing weights λ_1, λ_2, and λ_3. In this study, all the balancing weights are set to 1. The cross entropy loss function with sigmoid activation function $\sigma(\hat{o}_i)$ is used to calculate objectness loss.

$$L_{ob}\left(\hat{o}_i, o_i\right) = o_i \log \sigma\left(\hat{o}_i\right) - \left(1 - o_i\right) \log\left(1 - \sigma\left(\hat{o}_i\right)\right), \quad (2)$$

The localization loss calculates the differences between coordinates of the predicted and ground truth bounding boxes. The center coordinates loss is calculated by the cross entropy function with the sigmoid activation function, and the loss of width and height is calculated by the smooth L1 used in [14]. The classification loss based on the softmax activation function is computed by the cross entropy function.

Data augmentation Our network can be easily trained end-to-end with hard negative mining and data augmentation schemes through stochastic gradient decent (SGD) optimization. Each SGD mini-batch is established by eight sample images that are chosen uniformly and randomly from the training dataset. Random scaling, translation, flip, and color distortion are processed for data augmentation. First, the input image is randomly added or cropped by up to 20% of the original image size, and then the images are randomly resized to arbitrary size

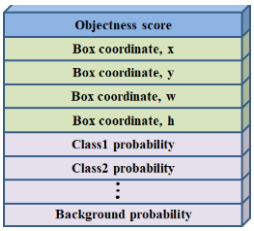

Figure 3. Channel configuration per a anchor box in the last featuer map.

between a half and two times. After that, the image is flipped by a 0.5 probability. The severe geometric distortion is minimized while the aspect ratio of the image is maintained. The distorted image is randomly adjusted in the HSV color space to be robust against illumination changes as well as to augment the training images.

Training In the last feature map, the $W_f \times H_f$ non-overlapping grid cells are established through multiple convolutional and pooling layers. The objectness score, box regression, and classification probabilities for multiple anchor boxes are calculated at each grid. During training, the negative examples are defined as the boxes that overlap a predicted box and with ground truth boxes less than 0.5. Only one box, the one that best matches a ground truth box, is selected as a positive sample. Thus, the number of positive samples equals that of the ground truth boxes. The desired target value is defined as the intersection-of-union (IOU) between the predicted box and the ground truth box in the objectness score loss if the predicted box is positive, otherwise, the desired value becomes 0. If a predicted box corresponds to a positive sample, the regression loss of the bounding box is calculated by the difference between the predicted box and the ground truth box. L1 and the smooth L1 loss are calculated for center coordinates and the width/height of the anchor box, respectively. Most predicted boxes correspond to negative examples, which leads to a significant imbalance between positive and negative samples during training. A hard negative example mining scheme is adopted to handle this problem [11-12]. The predicted boxes corresponding to negative samples are sorted and ones with the highest object confidence are selected as hard negative examples so that the ratio between positives and negatives is 1:4. The objectness score loss and classification loss are computed using the hard negative samples and positive samples.

Inference Each grid on the last feature map predicts N_b of the bounding boxes and their class-specific probabilities $P_r(O, C_i)$ which are calculated using objectness scores $P_r(O)$ and class probabilities $P_r(C_i)$.

$$P_r^i\left(O, C_i\right) = P_r\left(O\right) P_r\left(C_i\right), \quad (3)$$

where the class-specific probability $P_r(O, C_i)$ denotes the joint probabilities for the i^{th} target class, where $i \in [0, N_C]$ and N_C class corresponds to the background. Each bounding box represents a category with the highest probability. The multiple bounding boxes are generated around a target object due to scale and spatial correlation. Greedy non-maximum suppression (NMS) using an IOU threshold removes less confident bounding boxes. The NMS is processed independently for each class, and final detection boxes are established.

TABLE I. COMPARATIVE RESULT OF DETECTORS WITH/WITHOUT PRE-TRAINED WEIGHTS

Pre-train	easy			moderate			hard		
	car	pedestrian	cyclist	car	pedestrian	cyclist	car	pedestrian	cyclist
No	**96.67**	65.14	69.65	88.75	56.28	46.28	79.27	47.57	44.13
Yes	96.62	**79.62**	**88.09**	**91.11**	**70.55**	**62.77**	**79.81**	**61.47**	**60.18**

TABLE II. EVALUATION RESULTS OF DETECTON MODELS BASED ON EXTENDED PATHTHROUGH LAYER (EPT), PATHTHROUGH LAYER (PT), 4-POOLING AND 5-POOLING WITH RANDOM WEIGHTS.

method	easy			moderate			hard		
	car	pedestrian	cyclist	car	pedestrian	cyclist	car	pedestrian	cyclist
EPT	**95.05**	**60.94**	**59.46**	**85.25**	**51.01**	**39.01**	**74.26**	**44.34**	**37.39**
PT	90.87	57.01	47.77	79.39	46.71	30.88	68.37	40.03	29.52
4-pooling	90.68	58.09	48.77	80.76	49.63	32.07	70.18	43.01	30.54
5-pooling	91.06	53.10	39.72	76.30	43.42	26.63	65.60	38.63	25.13

TABLE III. RUN TIME, MEMORY USAGE, AND COMPLEXITY ANALYSES. RUNTIMES ARE MEASURED ON A NVIDIA TITAN XP GPU, EXCEPT NMS PROCESSING.

	EPT	PT	4-pooling	5-pooling
Inference speed	20.0ms	20.5ms	24.5ms	17.5ms
# of parameters	49.1M	65.5M	47.1M	47.1M
FLOPS	96.01B	86.25B	179.72B	70.47B

TABLE IV. EVALUATION RESULTS COMPARED WITH STATE-OF-THE-ART METHODS ON THE KITTI VALIDATIN SET.

method	easy			moderate			hard		
	car	pedestrian	cyclist	car	pedestrian	cyclist	car	pedestrian	cyclist
YOLOv2 [13]	93.94	65.10	57.65	83.56	54.24	35.22	72.18	47.35	33.53
MS-CNN [27]	94.08	77.74	-	89.12	**72.49**	-	75.54	**64.43**	-
3DOP [28]	93.08	71.40	83.82	88.07	64.46	63.47	79.39	60.39	60.93
MONO3D [30]	93.89	72.20	84.26	88.67	65.10	**64.25**	79.68	64.25	**61.94**
ExtendNet (ours)	**96.62**	**79.62**	**88.09**	**91.11**	70.55	62.77	**79.81**	61.47	60.18

TABLE V. RUN TIME ANALYSES OF STATE-OF-THE-ART METHODS.

	MS-CNN	3DOP	MONO3D	ExtendNet (ours)
Inference speed	0.4s	3.0s	4.2s	0.019s

IV. EXPERIMENTAL RESULTS

Our detection model is trained on the MS COCO datasets [9] for pre-training and KITTI training images [10] for fine-tuning. The MS COCO dataset has 80 object categories on the 80k train set, 40k validation set, and 20k test set. The dataset is used for pre-training our detection model because the dataset contains much richer images than the KITTI training images. In the pre-training, the learning rate is set as 10^{-2} for 100 iteration, 10^{-3} for the next 60k, and 10^{-4} for the next 120k with a batch size of 64.

The KITTI dataset is captured from a camera mounted on a driving vehicle. The dataset consists of 7481 training images and 7518 test images which includes several different sequences. The training set contains 51,865 labeled objects including 28,742 cars, 4,487 pedestrians, and 1,627 cyclists. The other labeled objects are truck, van, tram, misc, dontcare and sitting person, which are not evaluated in the KITTI evaluation kit. In these experiments, we evaluated our detection model on the validation set split from the training set as in the previous work [27, 28, 30]. This split ensures that sequences used in training sets are excluded in validation sets. Our detection model is fine-tuned with the 3712 training images while the training images are augmented by color and geometric distortions in order to artificially increase the number of training images. In the fine-tuning, the learning rate is set as 10^{-3} for 100 iteration, 10^{-4} for the next 20k, and 10^{-5} for the next 60k with a batch size of 64. Detection performance was evaluated using the average precision (AP) in a precision-recall graph. The detection IOU was 0.7 for cars and 0.5 for both pedestrians and cyclists. The KITTI evaluation is defined as easy, moderate, and hard according to the difficulty of the conditions, namely, the minimum height, occlusion level, and maximum truncation.

First, we examined the performance of our detection models (ExtendNet) with or without pre-trained weights. In the model with pre-trained weights, the weights were initially learned using the MS COCO dataset with five categories, namely, car, bus, bicycle, motorbike, and person, and then the pre-trained weights were used as the initial weights and the network was fine-tuned using the KITTI training dataset. In the model without pre-trained weights, the model was directly learned by the KITTI training set. Our experimental results showed that the method with pre-trained weights outperformed the method without pre-trained weights by a large margin (see Table I). The accuracy of car detection was slightly improved over the other categories because the number of cars was much higher than in the other classes. The cyclist category was not labeled in the MS COCO dataset, but similar objects, such as bicycle and motorbike, were labeled in the dataset instead. This way sufficiently improved the detection accuracy when the number of objects was insufficient in the training set.

We investigated the superiority of the proposed EPT layer. YOLOv2 network architecture was used as a baseline network

in the experiments for comparison under the same conditions. A method using an EPT layer was compared with a method using a PT layer, a method using four pooling layers without a PT layer, and a method using five pooling layers without a PT layer. The size of the last feature map in the methods with an EPT layer and four pooling layers became twice the size of that in the other methods. The method using five pooling layers had the fastest inference speed with the smallest memory usage and complexity, but it achieved far worse performance than the other methods (see Table Ⅱ and Table Ⅲ). The proposed architecture based on EPT achieved the best performance with reasonable complexity and inference speed compared to the other architectures (see Table Ⅱ and Table Ⅲ).

Performance of our ExtendNet based on the EPT layer was compared with other state-of-the-art methods whose performances were evaluated in the same validation set. The experimental results showed that our detector achieves a superior performance compared to the previous state-of-the-art methods under various conditions, especially easy conditions (see Table Ⅳ). Our detector ran much faster in inference speed and consumed much less memory than the other methods (see Table Ⅴ). Our model requires approximately 6.2M of parameters, which is about ten times less than those of YOLOv2.

V. Conclusion

In this paper, we proposed a fast and lightweight detection model based on an extended single feature map. Our network is based on the previous YOLOv2 architecture, but both the performance and inference speed are much more improved due to several kinds of techniques, such as hard negative mining, data augmentation, loss function, and extended path through layer. Experimental results demonstrated that our method outperforms previous state-of-the art methods with an overwhelmingly fast inference speed.

In the future, we will carry out ablation tests on the effects of various hyper-parameters. We will also study the integrated research on a multiple target tracking framework based on a fully convolutional network.

Acknowledgment

This work was supported by the DGIST R&D Program of the Ministry of Science, ICT.

References

[1] H. Zhou, P. Cao, and S. Chen, "A novel waveform design for multi-target detection in automotive FMCW radar," *IEEE Radar Conference*, pp. 1-5, 2016.

[2] T. Ogawa and G. Wanielik, "Track-Before-Detect approach on LIDAR signal processing for low SNR target detection", in *Proc. IEEE Intelligent Vehicles Symposium,* pp. 676-682, 2016.

[3] S. Mueller, D. Hospach, J. Gerlach, O. Bringmann, and W. Rosenstiel, "Robustness Evaluation and Improvement for Vision-based Advanced Driver Assistance Systems," in *Proc. IEEE Conference on Intelligent Transportation Systems*, pp. 2659-2664, 2015.

[4] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE CVPR*, vol. 1, pp. 511-518, 2001.

[5] N. Dalal, N and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE CVPR*, vol. 1, pp. 886-893, 2005.

[6] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Trans. PAMI*, vol. 36, no. 8, pp. 1532-1545, 2014.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, vol. 1, pp. 1097-1105, 2012.

[8] M. Everingham, S. M. A. Eslami, L. Van Gool, C. Williams, J. Winn and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *IJCV*, vol. 111, no. 1 pp. 98-136, 2015.

[9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. ECCV*. pp. 740-755, 2014.

[10] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite, in *Proc. IEEE CVPR*, pp. 3354–3361, 2012.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy and S. Reed, "SSD: Single shot multibox detector," in *Proc. ECCV,* pp. 21-37, 2016.

[12] J. Dai, Y. Li, K. He and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. NIPS*, pp. 379-387, 2016.

[13] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *Proc. CVPR,* pp.6517-6525, 2017.

[14] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR,* pp.580-587, 2014.

[15] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, "Selective search for object recognition," IJCV, vol. 104, no. 2, pp 154-171, 2013.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. NIPS*, vol. 1, pp. 91-99, 2015.

[17] R. Girshick, "Fast R-CNN", in *Proc. ICCV*, pp. 1440-1448, 2015.

[18] W. Bichen, I. Forrest, H. J. Peter, and K. Kurt, "SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving," in *Proc. CVPRW,* pp. 446-454*,* 2017.

[19] T. Subarna, D. Gokce, K. Byeongkeun, B. Vasudev, and N. Truong, "LCDet: Low-complexity fully-convolutional neural networks for object detection in embedded systems," in *Proc. CVPRW*, pp. 411-420*,* 2017.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. ECCV*, pp. 346-361, 2014.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE CVPR*, vol. 1, pp. 779-788, 2016.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015.

[23] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. NIPS*, pp. 2377-2385, 2015.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, pp.770-778, 2016.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] G. Huang, Z. Liu, and K. Q.Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, pp.2261-2269, 2017.

[27] X. Chen, K. Kunku, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler,and R. Urtasun, "3d object proposals for accurate object class detection," in *Proc. NIPS*, pp.424-432, 2015.

[28] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Proc. ECCV*, pp.424-432, 2016.

[29] K. Ashraf, B. Wu, F. Iandola, M. Moskewicz, and K. Keutzer, "Shallow networks for high-accuracy road object-detection," in *Proc. VEHITS*, pp. 33-40, 2017.

[30] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *Proc. CVPRW*, pp. 411-420*,* 2016.