Proceedings of the 8th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'22)
Prague, Czech Republic - July 28- 30, 2022
Paper No. MVML 105
DOI: 10.11159/mvml22.105

Robust Inference of Multi-Task Convolutional Neural Network for Advanced Driving Assistance by Embedding Coordinates

Masayuki Miyama[1]
[1]Advanced Mobility Research Institute, Kanazawa University
Kanazawa, Ishikawa, Japan
miyama@se.kanazawa-u.ac.jp

Abstract - In this study, we develop a multitasking CNN (Convolutional Neural Network) for advanced driving assistance. The network simultaneously performs three tasks: object detection, semantic segmentation, and disparity estimation. Edge computing requires low computation and low storage capacity, so the three tasks share not only one encoder, but also one decoder that employs a combination of depth-wise point-wise convolution and bilinear interpolation instead of the usual transpose convolution. This reduces the number of multiply-accumulate operations to 44.0% and the number of convolution weight parameters to 38.2%. In multitasking CNN training, the loss weights for each task were automatically adjusted by backpropagation, and the three tasks were learned in a balanced manner. Reducing the complexity of the decoder did not degrade the recognition accuracy, but rather improved it. Moreover, we found that entering pixel coordinates in this CNN significantly reduced misestimations for images that differed significantly from those during training.

Keywords**:** object detection, semantic segmentation, disparity estimation, multi-task, CNN, advanced driving assistance

1. Introduction

Image recognition is a process of inferring the characteristics of an entity projected onto an image by analysis of the image and is one of the pattern recognition techniques. Category classification is a task to find the category of the object shown in the image. Object detection is a task to detect objects in an image. Semantic segmentation performs category classification pixel by pixel. Disparity estimation by stereo vision finds the disparity from the correspondence of pixels shown in two left right images simultaneously captured from cameras installed parallel. Using the disparity, the distance from the camera to the object corresponding to the pixel is determined by the principle of trigonometry. Studies on disparity estimation using a monocular image are also being conducted.

DNN (Deep Neural Network) is a neural network with more than three layers, and rapidly developed. CNN (Convolutional Neural Network) is a type of DNN, and its application to image recognition has been dramatically improved the recognition accuracy. Its applications are evolving to (category identified) object detection, semantic segmentation, and motion / disparity estimation. In general, CNNs are computationally intensive and require a lot of power for training and inference. Also, many misestimations occur when the characteristics of the inferred dataset (target domain) and the characteristics of the dataset used for training (source domain) are significantly different. Furthermore, it has been reported that inputting pixel position information into the CNN along with the image does not significantly improve accuracy [1].

Advanced driving assistance is to support safe driving by using advanced technology such as image recognition. Forward collision warning, emergency braking system, and lane departure warning are examples of this assistance. Development of the multi-task CNN that simultaneously performs multiple AI (Artificial Intelligence) tasks with one CNN is progressing, and it is being applied to advanced driving assistance [2].

Contribution: This study develops a multi-task CNN for advanced driving assistance that simultaneously performs the three tasks of object detection, semantic segmentation, and disparity estimation. The three tasks fully share one decoder that uses a combination of depth-wise point-wise convolution and bilinear interpolation instead of usual transposed convolution. This results in reducing the number of multiply-accumulate operations to 44.0%, the number of weight parameters to 38.2%. At the training of the multi-task CNN, the weight of the loss for each task was automatically adjusted by the error backpropagation method, and three tasks were learned in a well-balanced manner. Though the complexity of the decoder was reduced, the recognition accuracy did not decrease, but rather improved. Furthermore, the proposed CNN was applied to images taken in a country different from learning. The results show that entering pixel position information into the CNN does not significantly improve the accuracy of the source domain, but it can significantly reduce misestimation of the target domain. This result overturns the conventional knowledge.

The structure of this article is as follows. Section 2 describes related works. Section 3 describes the proposed multi-task CNN structure, and weighting method for loss function. Section 4 describes the experimental setup and shows the results. Section 5 concludes this paper.

2. Related Work

Category classification is an AI task that identifies the type of the object in an image. MobileNet v1 is a category classification CNN designed to be aware of the limited resources of embedded applications and to maximize accuracy efficiently [3]. The feature of MobileNet v1 is a two-stage convolution called depth-wise separable convolution. We call it depth-wise point-wise (DP) convolution here. As a result, the amount of calculation can be reduced to about 1/8 to 1/9 compared to ordinary convolution with 3 by 3 kernel.

Object detection is the detection of position, size and category for each object in an image. SSD (Single Shot Multi-Box Detector) is a CNN for object detection [4]. The image features are extracted by the base network (VGG-16, the well-known CNN for category classification) and the extra network, and multiple feature maps with different resolutions are generated. Default boxes of different sizes and shapes are arranged on each pixel of each feature map. In each default box, the inference of the difference between the bounding box surrounding the detected object and the default box (regression), and the inference of the class probability distribution of the object (classification), are conducted.

Semantic segmentation is an AI task that divides an image by pixel-wise category classification. U-Net is one of the semantic segmentation CNNs [5]. The U-Net consists of an encoder, a decoder, and skip connections. The encoder performs a step-by-step feature extraction on the multiple convolution layers and the max pooling layers and outputs a feature map. The decoder upsamples the feature map gradually with the multiple transposed convolution layers to generate a segmentation image with the original image size. The skip connection connects the output of the encoder to the input of the decoder with the same resolution level and sharpens the boundaries of the output image.

FlowNet is a fully convolutional network with a U-Net like encoder-decoder architecture for optical flow estimation [6]. It can also be applied to disparity estimation using stereo vision. The encoder extracts the feature map from a pair of two images while gradually lowering the resolution, and the decoder gradually increases the resolution to generate the motion flow for each pixel from the encoded feature map. Recently, studies have been conducted on CNNs for monocular motion estimation [7].

Multi-task learning is learning multiple different tasks simultaneously. In machine learning, it has been studied before the development of deep learning. Multi-task learning implicitly augments data, suppresses overfitting by regularization, and potentially speeds up learning [8,9]. There are two methods for multi-tasking DNN, one is to share the parameters hard, and the other is to share them softly (regularize the parameters of each task as a whole). Kendall et al. propose a multitasking network that performs semantic segmentation, instance segmentation, and disparity estimation at the same time [2]. In the instance segmentation, a vector to the center coordinate of the instance to which each pixel belongs is estimated for each pixel, clustering using the Hough transform is performed, and the pixels are assigned to the instance. Then, they propose a method to dynamically assign weights to the loss of each task during learning.

CNNs are originally position-invariant. Studies are being conducted to improve accuracy by inputting position information into the CNN. Islam et al. showed that zero padding in the convolutional operation implicitly gives the CNN position information [10]. Miura et al. applied a CNN with explicit position information to medical images and reported that the accuracy did not improve much [1]. Ren et al. reported that CNNs with explicit position information were applied to traffic sign detection to improve accuracy [11], but their datasets used for learning and inference were similar.

3. Methods

3.1. Network Structure

We have developed the multitasking CNN for advanced driving assistance. This network performs object detection, semantic segmentation and disparity estimation at the same time. Assuming use at the edge, the following techniques are

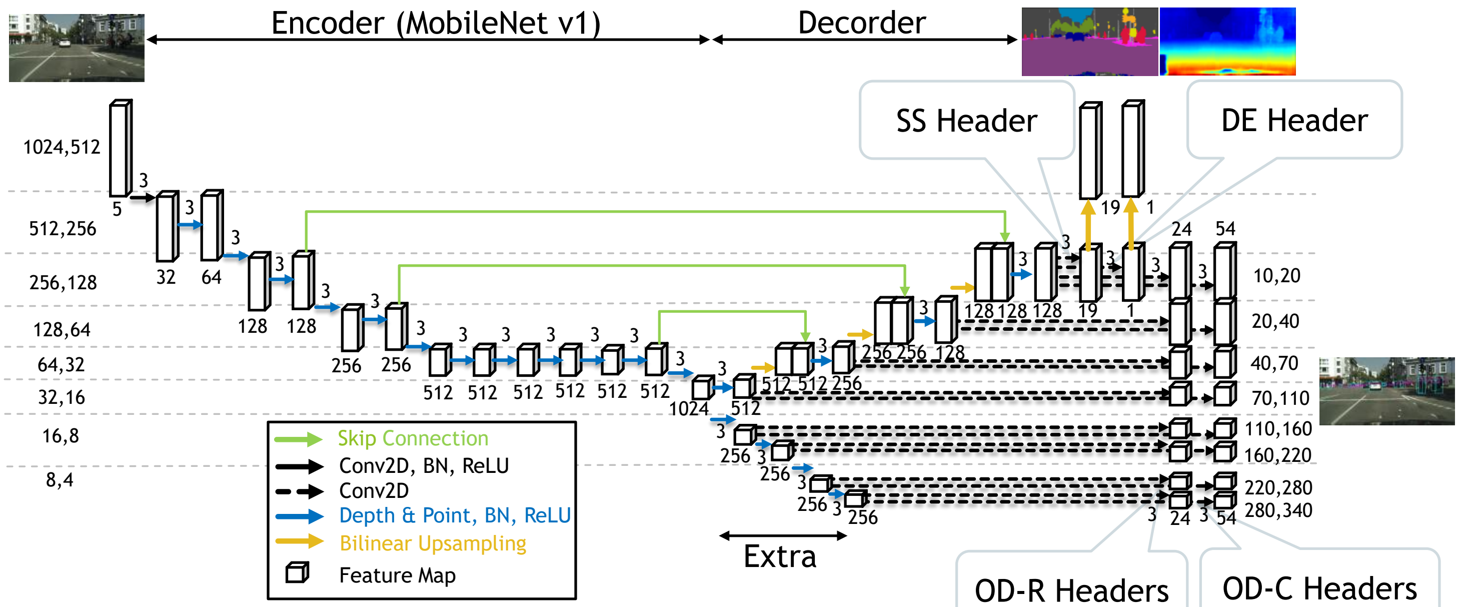

Fig. 1**:** The proposed multitasking CNN structure. This network performs object detection, semantic segmentation, and disparity estimation at the same time. Not only the encoder, but also the decoder is fully shared by the three tasks. Replacing the normal transposed convolution of the decoder with a combination of DP convolution and bilinear interpolation reduces the number of operations and parameters while maintaining accuracy.

used to reduce the amount of calculation while maintaining accuracy.

- Based on SSD for all tasks.
- High-precision, low-calculation MobileNet V1 for feature extraction.
- Not only the encoder but also the decoder is fully shared by three tasks.
- Replacing a usual transposed convolution in a decoder with a combination of DP convolution and bilinear interpolation.

Fig. 1 shows the structure of the multitasking CNN. As shown in the legend in the figure, the arrow represents the process, and the cube represents the feature map. The number above the arrow represents the kernel size. The numbers below the cube represent the number of channels. The width and height of the feature map are shown on the left side of the figure. The minimum and maximum bounding box sizes of the object detection are shown on the right side of the figure.

The base network is the SSD. The inputs have three image channels and two x, y coordinates, five channels in total. MobileNet v1 is used for feature extraction. The extra network has changed the resolution to 2 levels and the standard convolution to DP convolution. The decoder is fully shared by the three tasks. The decoder consists of a combination of DP convolution and bilinear interpolation rather than the usual transpose convolution. The encoder and decoder are connected by skip connections like U-Net. The headers of each task (Semantic Segmentation Header, Disparity Estimation Header, Object Detection Regression Header, Object Detection Classification Header) are connected after the shared decoder. The network performs a 4x bilinear interpolation to get the output of semantic segmentation and disparity estimation with the same resolution as the input.

The number of output channels for semantic segmentation is 19, which is the same as the number of Cityscapes classes [12,13]. The number of output channels for disparity estimation is 1. Eight feature maps are used for object detection. Object detection headers use the outputs from the decoder for the high-resolution feature maps, from the extra for the low-resolution feature maps. Since the number of default box types is 6 and the number of detection classes is 9, the number of output channels for regression is 6 × 4 (center x, y coordinates, width, height) = 24, and the number of output channels for classification is 6 × 9 = 54.

Table 1: Relationship between the decoder configuration, the number of operations, and the number of parameters.

	# Operations				**# Parameters**			
Subnet	***TC* [M**	**%]**	***C&I* [M**	**%]**	***TC* [K**	**%]**	***C&I* [K**	**%]**
Encoder	5,390	18.1	5,390	41.1	2,127	16.3	2,127	42.8
Extra	48	0.2	48	0.4	475	3.6	475	9.5
Decoder	12,080	40.5	2,582	19.7	7,668	58.9	910	18.3
SS Head	1,474	4.9	757	5.8	44	0.3	22	0.4
DE Head	78	0.3	40	0.3	2	0.0	1	0.0
OD-R Head	3,302	11.1	1,320	10.1	829	6.4	442	8.9
OD-C Head	7,429	24.9	2,970	22.7	1,866	14.3	995	20.0
total	29,801	-	13,107	44.0	13,012	-	4,973	38.2

Table 1 shows the relationship between the decoder configuration, the number of multiply-accumulate operations, and the number of convolution weight parameters. The combination of DP convolution and bilinear interpolation in the proposed network (C & I) replaces the kernel size 3 transpose convolution in the conventional network (TC). The headers of the conventional network input the feature map created by concatenating the feature map from the encoder and the feature map from the decoder (lower layer). Compared to the conventional network, the number of multiply-accumulate operations in the proposed network is reduced to 44.0%, and the number of weight parameters is reduced to 38.2%. The decoder ratio to the whole has also decreased significantly. Note that, in fact, by fully sharing the decoder, both the complexity and the number of parameters have already been significantly reduced in the conventional network. As detailed in the experiments in the next section, the proposed network is more accurate than the conventional network, despite the significant simplification of the decoder.

3.2. Weight Assignment for Loss of Each Task

When learning the multitasking CNN, Kendall et al. gave the weights expressed by the following equations to the loss of each task [2].

$$loss = \frac{1}{2{\sigma_A}^2} loss_A + \frac{1}{2{\sigma_B}^2} loss_B + log\sigma_A + log\sigma_B \tag{1}$$

$$p(y_A) = \mathcal{N}(f(x), \sigma_A^2),\ \ p(y_B) = \mathcal{N}(f(x), \sigma_B^2) \tag{2}$$

Here, y is the correct answer for the input $x, p(y)$ is the probability of y, $\mathcal{N}$ is the normal distribution, $f(x)$ is the output of the CNN for the input x, σ^2 is the variance, and the subscripts represent the two tasks A and B. During training, the variance is calculated, and the loss is weighted according to this equation. A method has also been proposed in which this equation is used only to calculate the loss and the weight value is obtained by error back propagation [14]. We conducted preliminary experiments on this method, but the weight for each loss did not change significantly during training. Instead, we decided to use the method of reference [15], that uses the following equation weighted by the exponential function to calculate the loss followed by backpropagation to automatically calculate the of the parameter w.

$$loss = e^{-w_A}\, loss_A + e^{-w_B}\, loss_B + w_A + w_B \tag{3}$$

4. Experiments

4.1. Setup

We used the Cityscape [12,13] for our experiments. Cityscapes is a dataset of driver-viewpoint images taken by a dashcam in multiple cities in Germany, including correct data for object detection and semantic segmentation. The correct data for disparity estimation is pseudo data created by the SGM algorithm. It consists of 2975 images for training and 500 images for verification, with an original resolution of 2048 by 1024, but we used images resized to 1024 by 512. PyTorch was used for network and learning / inference programming. The number of learnings was 200 epochs, the optimization method was Adam, the learning rate was 0.001, and the learning rate was reduced to 1/10 every 80 epochs.

Table 2: Accuracy comparison between combinations of network, coordinate addition, and loss weighting method.

Network	*Coordinate*	*Weight*	*mAP[%]*	*mIoU[%]*	*RDE[%]*
C&I	No	1:0:0	**50.3**	-	-
		0:1:0	-	**67.7**	-
		0:0:1	-	-	3.50
		1:1:1	42.0	50.6	**3.39**
		auto	48.5	65.2	3.84
TC	No	auto	48.4	64.4	4.14
C&I	Yes	auto	48.0	65.1	3.78

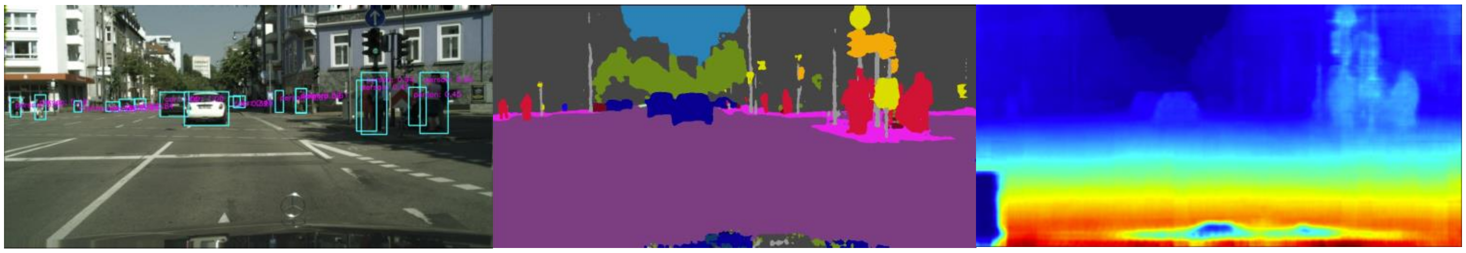

Fig. 2**:** Results of proposed network. Left: object detection, center: semantic segmentation, right: disparity estimation.

4.2. Results

Table 2 except the bottom two rows shows the comparison results of the proposed network (without pixel coordinates) by changing the weighting method for the loss of each task. Here, X:Y:Z is a fixed weight given to the loss of object detection (X), semantic segmentation (Y), and disparity estimation (Z), respectively. 'auto' represents that the weights are automatically assigned by the method of section 3.2. As accuracy indexes, mAP (mean Average Precision) at IoU (Intersection over Union) =0.5 was used for object detection, mIoU (mean IoU) was used for semantic segmentation, and RDE (Relative Disparity Error) defined by the following equation was used for disparity estimation.

$$RDE = \frac{\frac{1}{n}\sum_i^n |GroundTruth_i - Prediction_i|}{\frac{1}{n}\sum_i^n |GroundTruth_i|} \tag{4}$$

Here, $GroundTruth_i$ is the disparity of the correct answer at point i, and $Prediction_i$ is the estimated value. The larger the mAP and mIoU and the smaller the RDE, the more accurate.

Compared to 1:1:1, it can be seen that 'auto' is learning three tasks in a well-balanced manner. 1:0:0, 0:1:0 and 0:0:1 are the results of learning the corresponding tasks individually. It can be seen that each task accuracy of 'auto' is almost equivalent to the accuracy of individual learning. Fig. 2 shows the result of inference by the CNN learned by 'auto', and it can be seen that each task is performed well. Since the learning is performed using the estimation by the SGM algorithm as a pseudo correct answer, obvious mistakes can be seen in the disparity estimation.

The second row from the bottom of Table 2 is the result of the conventional network using transposed convolution in the decoder. The loss weighting method is 'auto'. It can be seen that the accuracy is low for all tasks compared to the proposed network using DP convolution and bilinear interpolation in the decoder. As shown in Table 1, the number of multiply-accumulation operations and the number of convolution weight parameters are less than half, and the inference accuracy is high, so it can be said that the proposed method is much better than the conventional method.

The last row in Table 2 is the result of the proposed network using pixel coordinates. It is a little less accurate than if no coordinates were entered. The cities of training and inference are different, but the accuracy does not change much when we enter the coordinates. This result is the same as the conventional research result that the estimation accuracy does not change even if the coordinates are input [1]. The reason why the accuracy does not change is probably because both images are collected in the same way in the same Germany and the difference is not very large.

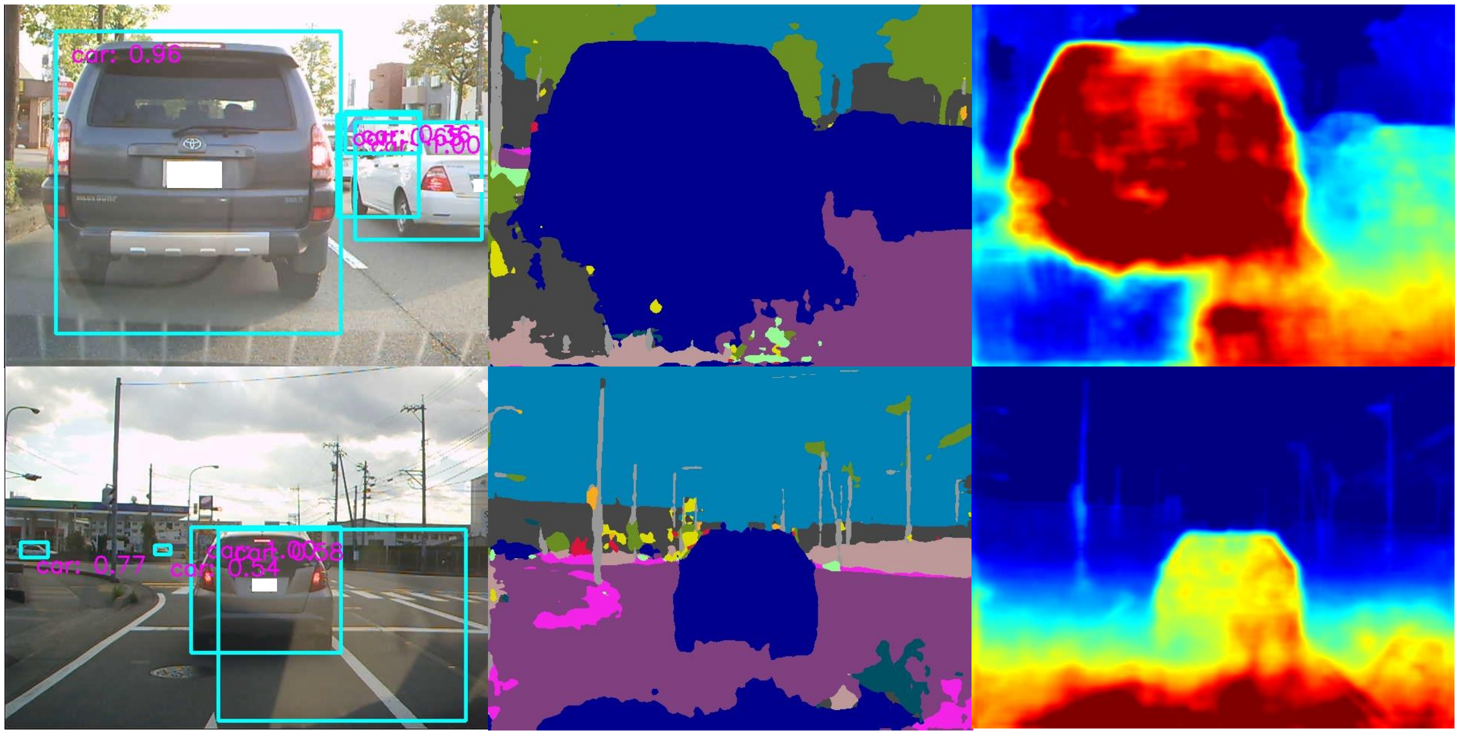

Fig. 3**:** Estimated results when the coordinates of the image are not entered. Top: Scene 1, Bottom: Scene 2. Left: Object detection, Center: Semantic segmentation, Right: Disparity estimation. There is noticeable noise on the road in both scenes. In scene 1, the building is misestimated in front of the car. In scene 2, the vehicle is erroneously detected by noise.

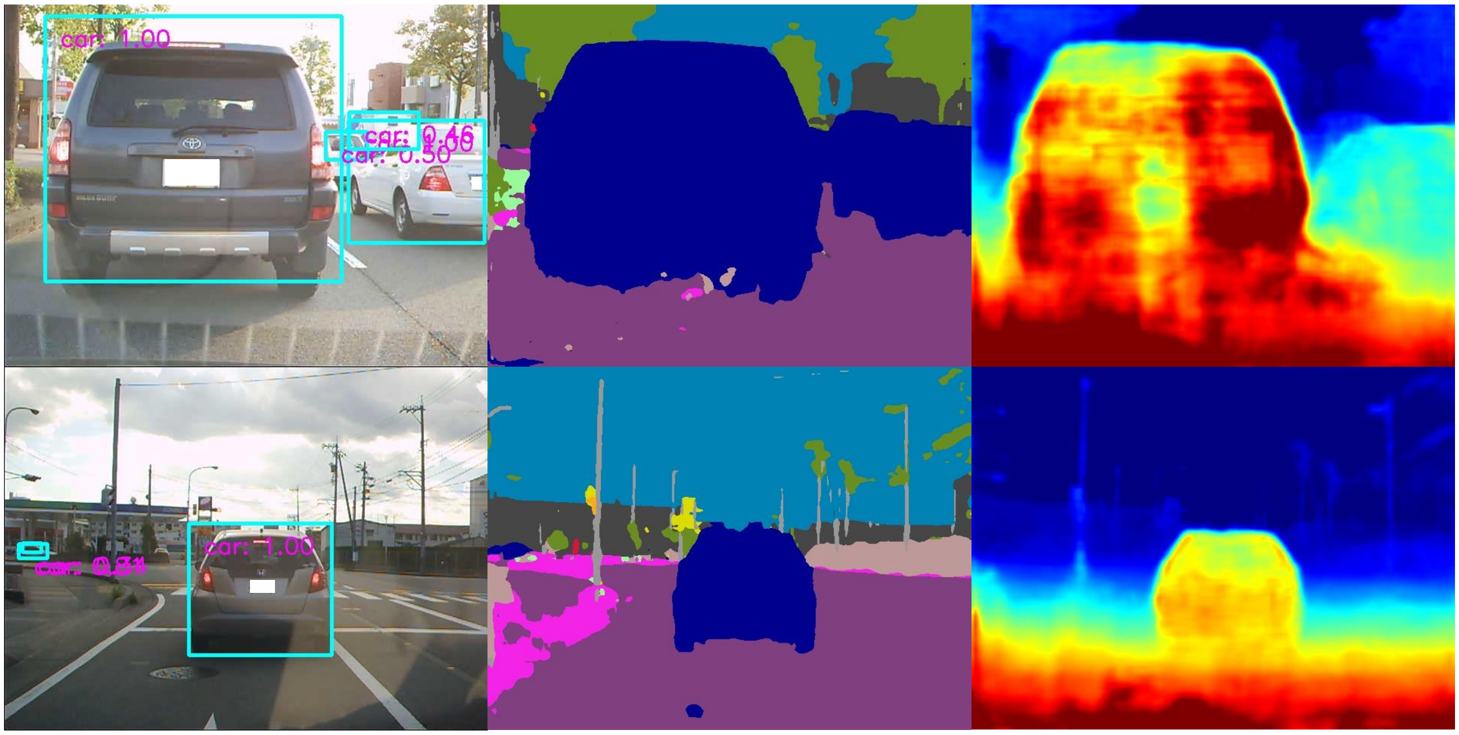

Fig.**4:** Estimated results when the coordinates of the image are entered. Top: Scene 1, Bottom: Scene 2. Left: Object detection, Center: Semantic segmentation, Right: Disparity estimation. There is no noticeable noise on the road in both scenes.

In general, many misestimations occur when the characteristics of the inferred dataset and the characteristics of the dataset used for training are significantly different. The estimation results shown in Fig. 3 are examples of this. These images are taken in Japan and extracted from a 30-fps video with a resolution of 640 by 480. The images contain a lot of block noise. In scene 1 (upper row), the semantic segmentation (center column) ridiculously estimates the building (dark brown) on the road just in front of the car, and the disparity estimation (right column) largely misestimates at the same location. In scene 2 (lower row), the vehicle is erroneously detected due to the noise. CNNs are originally position-invariant and can easily be mis-estimated, even though the coordinates are input implicitly with zero padding.

On the other hand, as shown in Fig.4, when the coordinates are input explicitly, no erroneous estimation has occurred on the road. By adding coordinates to the input, it is possible to suppress impossible inferences in the image from the in-vehicle camera that captures the front of the vehicle. The inference with coordinates can be said to be robust for images, in which the position of the object is determined to some extent by the type of object, and which are significantly different from training, such as this in-vehicle camera image. In the experiments in reference [11] showing the effectiveness of coordinate input, the traffic sign was detected, and the images of learning and inference were similar. On the other hand, we have shown that coordinate input is effective when the images of learning and inference are very different in the three tasks for in-vehicle images.

Let us consider why both semantic segmentation and disparity estimation cause misestimation at the same point when no coordinates are entered. Both of these are inference results from one multitasking CNN. This CNN uses a monocular image to estimate disparity. In this case, disparity estimation depends on the type, size, and position of the object. Therefore, there is a strong correlation between semantic segmentation and disparity estimation.

5. Conclusion

Advanced driving assistance requires advanced image recognition, and edge processing requires low computation and low storage capacity while maintaining accuracy. In this study, we developed a multitasking CNN for advanced driving assistance that simultaneously performs three tasks: object detection, semantic segmentation, and disparity estimation. The decoder is fully shared by the three tasks. The decoder uses a combination of DP convolution and bilinear interpolation instead of the usual transpose convolution. This reduces the number of multiply-accumulate operations to 44.0% and the number of weight parameters to 38.2%. In the CNN training, the loss weights for each task were automatically adjusted by the backpropagation, and the three tasks were learned in a balanced manner. Even if the complexity of the decoder is reduced, the recognition accuracy is not degraded, but rather improved. We also found that inference with coordinates was robust to dashcam images which was significantly different from training. Real-time implementation on embedded GPU boards and FPGA boards and evaluation of them are future works.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP 18K11350.

References

[1] Rito Murase and Masanori Suganuma and Takayuki Okatani, "How Can CNNs Use Image Position for Segmentation?" ArXiv 2005.03463 [eess.IV], 2020.

[2] A. Kendall, Y. Gal, and R Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7482-7491, doi: 10.1109/CVPR.2018.00781.

[3] Andrew G. Howard and Menglong Zhu and Bo Chen and Dmitry Kalenichenko and Weijun Wang and Tobias Weyand and Marco Andreetto and Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861 [cs.CV], 2017.

[4] Liu W. et al., "SSD: Single Shot MultiBox Detector," In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9905. Springer, Cham. https://doi.org/10.1007/978-3-319-46448-0_2

[5] Ronneberger O., Fischer P., Brox T., “U-Net: Convolutional Networks for Biomedical Image Segmentation,” In: Navab N., Hornegger J., Wells W., Frangi A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science, vol 9351. Springer, Cham. https://doi.org/10.1007/978-3-319-24574-4_28

[6] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazırbas¸ V. Golkov, "FlowNet: Learning Optical Flow with Convolutional Networks," 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 2758-2766, doi: 10.1109/ICCV.2015.316.

[7] C. Godard, O. M. Aodha, and G. J. Brostow, “Unsupervised monocular depth estimation with left-right consistency,” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6602-6611, doi: 10.1109/CVPR.2017.699.

[8] S. Ruder, “An overview of multi-task learning in deep neural networks,” arXiv:1706.05098 [cs.LG].

[9] M. Crawshaw, “Multi-task learning with deep neural networks: a survey,” arXiv:2009.09796 [cs.LG].

[10] Md Amirul Islam, Sen Jia, Neil D. B. Bruce, “How Much Position Information Do Convolutional Neural Networks Encode?,” arXiv:2001.08248 [cs.CV], 2020.

[11] Liliang Ren, Zhuonan Hao, “A Simple Fix for Convolutional Neural Network via Coordinate Embedding,” arXiv:2003.10589 [cs.CV], 2020.

[12] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The Cityscapes Dataset for Semantic Urban Scene Understanding,” in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[13] https://www.cityscapes-dataset.com/

[14] Lukas Liebel, Marco Körner, “Auxiliary Tasks in Multi-task Learning,” arXiv:1805.06334 [cs.CV], 2018.

[15] https://github.com/Xilinx/Vitis-AI/tree/master/models/AI-Model-Zoo, Performance on ZCU104, 103 multi_task_v3.B. Klaus and P. Horn, *Robot Vision*. Cambridge, MA: MIT Press, 1986.