Int. J.Adv.Sig.Img.Sci, Vol. 9, No. 2, 2023

REAL-TIME EMBEDDED SYSTEM OF MULTI-TASK CNN FOR ADVANCED DRIVING ASSISTANCE

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

Submitted: *Jul, 18, 2023* ***Revised:*** *Nov, 22, 2023* ***Accepted:*** *Dec, 11, 2023*

Abstract: In this research, we've engineered a real-time embedded system for advanced driving assistance. Our approach involves employing a multi-task Convolutional Neural Network (CNN) capable of simultaneously executing three tasks: object detection, semantic segmentation, and disparity estimation. Confronted with the limitations of edge computing, we've streamlined resource usage by sharing a common encoder and decoder among these tasks. To enhance computational efficiency, we've opted for a blend of depth-wise separable convolution and bilinear interpolation, departing from the conventional transposed convolution. This strategic change reduced the multiply-accumulate operations to 23.3% and the convolution parameters to 16.7%.Our experimental findings demonstrate that the decoder's complexity reduction not only avoids compromising recognition accuracy but, in fact, enhances it. Furthermore, we've embraced a semi-supervised learning approach to heighten network accuracy when deployed in a target domain divergent from the source domain used during training. Specifically, we've employed manually crafted correct answers only for object detection to train the whole network for optimal performance in the target domain. For the foreground object categories, we generate pseudo-correct responses for semantic segmentation by employing bounding boxes from object detection and iteratively refining them. Conversely, for the background categories, we rely on the initial inference outcomes as pseudo-correct responses, abstaining from further adjustments. Semantic segmentation of object classes with widely different appearances can be achieved thanks to this method, which tells the rough position, size, and shape of each object to the task. Our experimental results substantiate that the incorporation of this semi-supervised learning technique leads to enhancements in both object detection and semantic segmentation accuracy. We implemented this multi-task CNN on an embedded Graphics Processing Unit (GPU) board, added multi-object tracking functionality, and achieved a throughput of 18 fps with 26 Watt power consumption.

Keywords: Object detection, semantic segmentation, disparity estimation, multi-task CNN, advanced driving assistance, embedded GPU, multi-object tracking.

I. INTRODUCTION

Image recognition stands as a pivotal technology in pattern recognition, focusing on the examination of an image to discern the features of objects portrayed within it. Tasks such as category classification, object detection, and semantic segmentation are standard elements within the realm of image recognition. Additionally, the technique of stereo vision for disparity estimation plays a crucial role in gauging the distance of objects from the camera, leveraging the principles of trigonometry for this purpose.

Deep Neural Networks (DNNs), characterized by having more than three layers, are undergoing swift advancements. Among these, CNNs represent a specific class of DNNs that have significantly elevated recognition accuracy in applications like image classification, object detection, semantic segmentation, and motion/disparity estimation [1-4]. The training and inference stages of CNNs demand substantial processing power, prompting ongoing research aimed at crafting CNN architectures capable of diminishing computational complexity without compromising accuracy [5].

Advanced driving assistance technology leverages image recognition to augment overall driving safety. CNNs have found application in advanced driving assistance systems, specifically for tasks such as object detection, semantic segmentation, and disparity estimation. The development and implementation of multi-task CNNs, capable of concurrently executing various Artificial Intelligence (AI) tasks, are actively underway and are being integrated into advanced driving assistance frameworks [6]. Low power consumption and low computational complexity are essential for the in-vehicle computer and the CNN that runs on it.

Inaccuracies may arise when there are disparities in the traits of the target domain, where a dataset is applied, and the source domain, utilized for training purposes. Transfer learning is a common strategy employed to enhance the accuracy of inferences in the target domain. This machine learning approach entails leveraging acquired knowledge from a different domain or task. Domain adaptation, a specific form of transfer learning, comes into play when there are distinctions in the marginal probability distributions between the source and target domains.

Several approaches exist for data preparation in the realm of machine learning. In supervised learning, accurate data is supplied for every input, contrasting with unsupervised learning where no correct data is necessary for any input. Semi-supervised learning takes a middle ground, necessitating correct data for only a subset of the input. The adoption of unsupervised and semi-supervised learning can significantly alleviate the effort needed for the creation of accurate data.

In the past few years, there has been a rising fascination with unsupervised learning within the target domain [7-11]. Yet, attaining commendable recognition performance through unsupervised learning poses challenges, particularly when there's substantial dissimilarity in the visual characteristics between the source and target domains. For example, recognizing horizontal traffic lights using a model trained solely on vertical traffic lights, without any supervision, proves to be challenging especially in the context of on-vehicle image recognition, the focal point of this research. Although supervised learning can yield high performance, the creation of accurate data demands more effort, particularly for multi-task networks.

During this study, we engineered a real-time embedded system tailored for advanced driving assistance. This system integrates a multi-task CNN designed to execute object detection, semantic segmentation, and disparity estimation concurrently. The following points were devised in developing this system.

- Designed a multi-task CNN dedicated to advanced driving assistance, executing three concurrent tasks: object detection, semantic segmentation, and disparity estimation. The shared utilization of a single encoder and decoder is a distinctive feature, leveraging a blend of depth-wise separable convolution and bilinear interpolation. This strategic approach effectively trims the count of multiply-accumulate operations and convolution parameters. Empirical validations affirm that the reduction in decoder complexity not only avoids compromising recognition accuracy but indeed enhances it.

- Devised a semi-supervised learning approach tailored for the target domain. During training, precise answers are exclusively supplied for object detection. In the case of foreground categories, a pseudo-correct response for semantic segmentation is generated by utilizing the bounding box from object detection, iteratively refining it. Conversely, for background categories, the initial inference result is designated as a pseudo-correct response and remains unaltered. Object classes with widely different appearances can be semantically segmented thanks to this method, which gives the rough position, size, and shape of each object to the task. Empirical assessments affirm that the adoption of this methodology not only enhances object detection accuracy but also refines the precision of semantic segmentation estimation.

The deployment of this CNN onto an embedded GPU board, coupled with the integration of multi-object tracking capabilities, resulted in a throughput of 18 frames per second—an essential requirement for driving assistance—at a power consumption of 26 watts. To the best of our knowledge, no other embedded system concurrently executes these multiple functions within a comparable timeframe. This work builds on our previous work [12] and adds a semi-supervised learning method and an embedded GPU implementation.

The organization of this paper encompasses a survey of related works in Section 2, an exposition of the multi-task CNN structure utilized, and the semi-supervised learning method applied in the target domain of the multi-task network in Section 3. Section 4 outlines the experimental setup and presents the obtained results, while Section 5 delves into the specifics of the implementation on a GPU board. The concluding remarks of the paper are encapsulated in Section 6.

II. RELATED WORKS

Category classification stands as a foundational AI task, seeking to discern the nature of an object within an image. MobileNet v1, a category classification CNN, has been crafted with a keen awareness of the constrained resources prevalent in embedded applications, all the while striving to optimize accuracy [5]. A distinctive attribute is its adoption of a two-stage convolution known as depth-wise separable convolution, leading to a substantial reduction in computation costs—approximately 1/8 to 1/9 when compared to conventional convolution employing a 3 by 3 kernel.

On the other hand, object detection endeavors to identify the position, dimensions, and category of each object within an image. SSD (Single Shot Multi-Box Detector) represents a dedicated CNN tailored for this particular task [2]. The extraction of image features involves both the base network (VGG-16) [1], a renowned CNN designed for category classification) and the supplementary network, generating diverse feature maps with distinct resolutions. Across each pixel of these feature maps, default boxes of varying sizes and shapes are strategically positioned. In each default box, the model infers the difference between the bounding box surrounding the detected object and the default box (regression) and the class probability distribution of the object (classification).

Semantic segmentation, another fundamental AI objective, endeavors to partition an image through pixel-wise category classification. U-Net stands out as one of the most widely adopted CNNs tailored for this specific task [3]. Comprising an encoder, a decoder, and skip connections, U-Net operates in a distinctive manner. The encoder systematically conducts feature extraction through multiple convolution layers and max pooling layers, culminating in the generation of a feature map. Meanwhile, the decoder progressively up-scales the feature map using various transposed convolution layers, ultimately producing a segmentation image matching the original image size. The incorporation of skip connections serves to link the encoder's output to the decoder's input at the same resolution level, thereby enhancing the accuracy of the resulting image.

FlowNet, designed for optical flow estimation, adopts a fully convolutional network structure resembling a U-Net with encoder-decoder architecture [4]. In this design, the encoder undertakes feature map extraction from a pair of images, progressively reducing resolution, while the decoder incrementally enhances resolution to produce motion flow for each pixel derived from the encoded feature map. Additionally, the network is adaptable for disparity estimation through stereo vision. Recent investigations have explored CNNs specifically tailored for monocular motion estimation [13-15].

Multi-task learning encompasses the simultaneous acquisition of proficiency in various tasks. Its benefits include implicit data augmentation, mitigation of overfitting through regularization, and the potential acceleration of the learning process [16-18]. The exploration of auxiliary tasks and the optimization of multi-task network structures have been undertaken by researchers to enhance the accuracy of primary tasks such as semantic segmentation [19-20], along with the refinement of multi-task network architectures [21-22]. Kendall et al. have proposed a multi-task network capable of concurrently handling semantic segmentation, instance segmentation, and disparity estimation [6]. Their approach involves dynamically assigning weights to the loss of each task during the learning process. In a similar vein, other researchers have implemented double-task networks for object detection and semantic segmentation specifically tailored for in-vehicle images on GPU platforms [23]. Our implemented triple-task network addresses three key tasks: disparity estimation, object detection, and semantic segmentation. This CNN has been successfully deployed on an embedded GPU board, featuring added multi-object tracking capabilities, and achieves real-time performance.

Explorations have been undertaken in the realm of domain adaptation through unsupervised learning within the target domain. Adversarial networks employ a discriminative network to compute loss in the target domain without the aid of correct answers [7-9]. Another technique, known as self-training, leverages the model acquired in the source domain to generate pseudo-correct responses in the target domain. It involves discerning the marginal probability distribution of the target domain and adapting the model accordingly [10-11]. However, these methods suffer from poor recognition accuracy when the appearance of objects of the same category is significantly different in the source and target domains. Our approach can semantically segment object classes with such appearances, which are difficult to learn without supervision, by providing the correct answer only for object detection during the learning process.

The exploration of domain adaptation via semi-supervised learning has extended to multi-task networks tailored for in-vehicle images [24-26], specifically concentrating on tasks like semantic segmentation and disparity estimation. Nevertheless, these investigations typically operate under the assumption that correct data is absent for one of the two tasks in the target domain. In contrast, our research delves into a multi-task network proficient in three tasks: object detection, semantic segmentation, and disparity estimation. We make the distinctive assumption that only object detection benefits from correct data. This choice is grounded in the relative ease of annotating object detection with accurate data compared to the more intricate tasks of semantic segmentation and disparity estimation.

III. ADOPTED METHODS

A. Network Structure

We've created a multi-task CNN tailored for advanced driving assistance, capable of simultaneously executing object detection, semantic segmentation, and disparity estimation. Envisaging deployment at the edge, we've implemented the following techniques to curtail computational load without compromising 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 depth-wise separable convolution and bilinear interpolation.

The structure of the multi-task CNN is depicted in Figure 1. The legend in the figure indicates that the arrow symbolizes the process, while the cube represents the feature map. The numerical value above the arrow signifies the kernel size, and the figures below the cube indicate the number of channels. On the left side of the figure, you can observe the width and height of the feature map. Meanwhile, the minimum and maximum bounding box sizes for object detection are displayed on the right side of the figure.

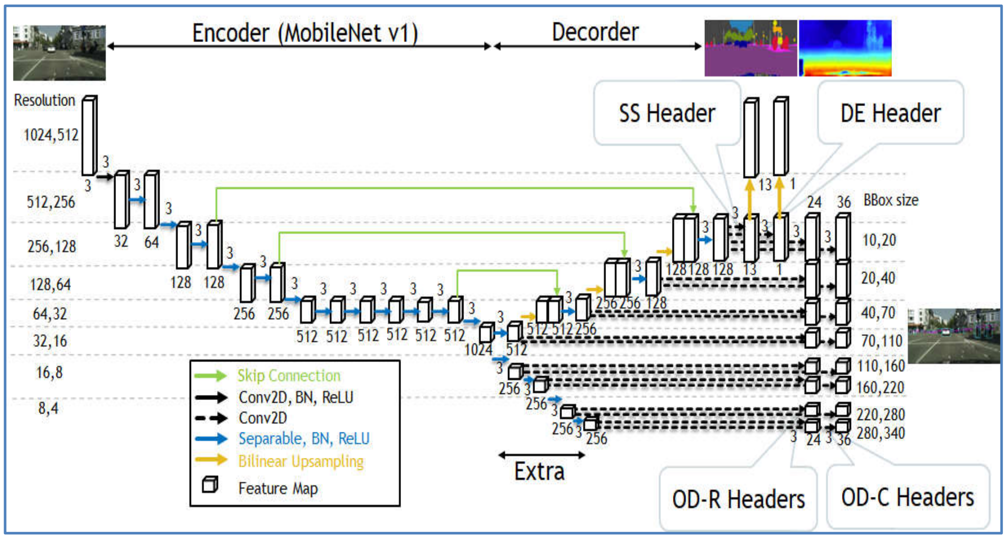


Fig. 1 Multi-task CNN for advanced driving assistance

We employed the SSD base network with input comprising three image channels. MobileNet v1 is utilized for feature extraction, and the extra network incorporates two-level resolution depth-wise separable convolutions. In a departure from the conventional transpose convolution, the fully shared decoder for the three tasks is fashioned through a combination of depth-wise separable convolution and bilinear interpolation. Similar to U-Net, skip connections link the encoder and decoder. Following the shared decoder, the headers for each task (Semantic Segmentation (SS) Header, Disparity Estimation (DE) Header, Object Detection Regression (OD-R) Header, and Object Detection Classification (OD-C) Header) are connected. Ultimately, the network executes a 4× bilinear interpolation to derive the outputs for semantic segmentation and disparity estimation, aligning them with the input's resolution.

The semantic segmentation task yields 13 output channels, corresponding to the following classes: "road," "sidewalk," "building," "fence," "pole," "traffic_light," "traffic_sign," "vegetation," "terrain," "sky," "person," "car," and "bike." For the disparity estimation task, a single output channel is generated. In the case of object detection, eight feature maps are employed. The object detection headers utilize outputs from the decoder for high-resolution feature maps and outputs from the extra network for low-resolution feature maps. Given the existence of six default box types and six detection classes ("traffic_light," "traffic_sign," "person," "car," "bike," and "background"), the number of output channels for classification is 6 × 6 = 36. Simultaneously, the number of output channels for regression is 6 × 4 (representing center x, y coordinates, width, and height) = 24.

Table 1 illustrates the correlation among network configurations, the count of multiply-accumulate operations, and the number of convolution weight parameters. Network I_T features an independent decoder for each task, utilizing transposed convolution in its decoding process. In contrast, network S_T employs a shared decoder for all tasks, also utilizing transposed convolution in the decoding stage. The adopted network, S_SI, incorporates a shared decoder for all tasks, employing a combination of depth-wise separable convolution and bilinear interpolation in the decoding process. Relative to the I_T network, the chosen S_SI network achieves a noteworthy reduction to 23.3% in multiply-accumulate operations and a decrease to 16.7% in weight parameters. Additionally, the proportion of the decoder in the overall structure significantly diminishes from 71.0% to 21.7%. As elucidated in the forthcoming experiments section, the selected network attains accuracy levels comparable to those employing transposed convolution, despite the considerably simplified decoder.

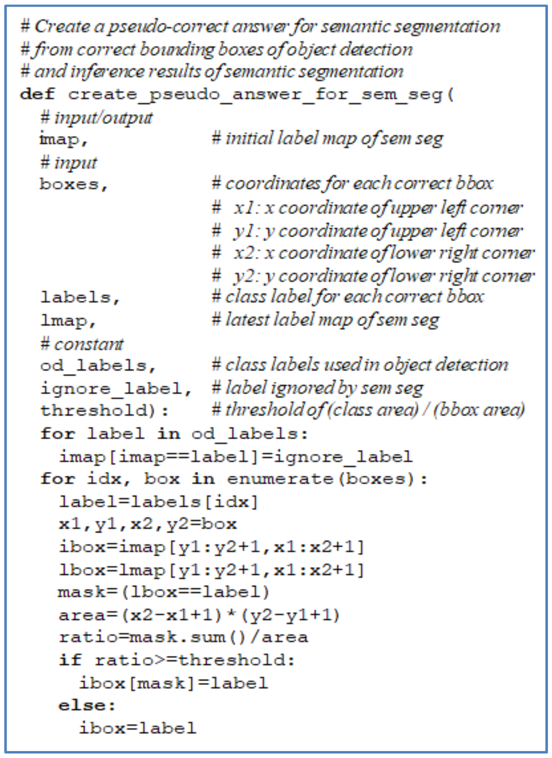
TABLE 1 Relationship between the decoder configuration, the number of operations, and the number of parameters

	#Operations						#Parameters					
Subnet	I_T[M/%]		S_T[M/%]		S_SI[M/%]		I_T[K/%]		S_T[K/%]		S_SI[K/%]	
Encoder	5,390	10.6	5,390	20.1	5,390	45.3	2,127	7.7	2,127	17.2	2,127	45.9
Extra	48	0.1	48	0.2	48	0.4	475	1.7	475	3.8	475	10.2
Decoder	36,239	71.0	12,080	45.0	2,582	21.7	23,003	83.0	7,668	62.0	910	19.6
SS Head	1,021	2.0	1,021	3.8	531	4.5	30	0.1	30	0.2	15	0.3
DE Head	78	0.2	78	0.3	40	0.3	2	0.0	2	0.0	1	0.0
OD-R Head	3,302	6.5	3,302	12.3	1,320	11.1	829	3.0	829	6.7	442	9.5
OD-C Head	4,953	9.7	4,953	18.4	1,980	16.7	1,244	4.5	1,244	10.1	664	14.3
total	51,031	100.0	26,871	100.0	13,107	100.0	27,711	100.0	12,376	100.0	4,634	100.0
total ratio	100.0%		52.7%		**23.3%**		100.0%		44.7%		**16.7%**	

B. Semi-supervised learning of multi-task network

We aim to apply our multi-task network to a target domain different from the source domain on which it was trained. However, it is well-known that applying a network trained on a source domain to a target domain generally results in a degradation of accuracy. While high accuracy can be achieved by supervised learning in the target domain, it is often challenging and time-consuming to create correct data. To overcome this, we adopt a semi-supervised learning approach that requires less effort in creating correct data. Specifically, we only prepare correct data for object detection, which is relatively easier to annotate. For semantic segmentation and disparity estimation, we generate pseudo-labels from the network's inference results.

Figure 2 is a python-like pseudo code to create a pseudo-correct answer for semantic segmentation. For non-target categories of object detection, such as "sky" and "road", the initial inference result is used as a pseudo-correct answer for semantic segmentation. For target categories such as "car" and "people", we create pseudo-correct answers using correct bounding boxes for object detection. Specifically, the area ratio between the set of pixels in the correct bounding box classified as label (inference region) and the box for the same label is calculated. If it is less than the threshold, the entire bounding box (ibox) is regarded as a pseudo-correct answer. If it is greater than or equal to the threshold, the inference region (ibox[mask]) is regarded as a pseudo-correct answer. Judgment using threshold can learn the shape of the object more accurately. Each pixel of the inference region outside the bounding box is assigned an ignore_label and no loss is computed. This prevents misestimated foregrounds from being pseudo correct answers. For disparity estimation, the initial inference result is used as a pseudo-correct answer for all categories.



```
# Create a pseudo-correct answer for semantic segmentation
# from correct bounding boxes of object detection
# and inference results of semantic segmentation
def create_pseudo_answer_for_sem_seg(
  # input/output
  imap,           # initial label map of sem seg
  # input
  boxes,          # coordinates for each correct bbox
                  #  x1: x coordinate of upper left corner
                  #  y1: y coordinate of upper left corner
                  #  x2: x coordinate of lower right corner
                  #  y2: y coordinate of lower right corner
  labels,         # class label for each correct bbox
  lmap,           # latest label map of sem seg
  # constant
  od_labels,      # class labels used in object detection
  ignore_label,   # label ignored by sem seg
  threshold):     # threshold of (class area) / (bbox area)
  for label in od_labels:
    imap[imap==label]=ignore_label
  for idx, box in enumerate(boxes):
    label=labels[idx]
    x1,y1,x2,y2=box
    ibox=imap[y1:y2+1,x1:x2+1]
    lbox=lmap[y1:y2+1,x1:x2+1]
    mask=(lbox==label)
    area=(x2-x1+1)*(y2-y1+1)
    ratio=mask.sum()/area
    if ratio>=threshold:
      ibox[mask]=label
    else:
      ibox=label
```

Fig.2 Python-like pseudo code to create a pseudo-correct answer for semantic segmentation

Figure 3 shows an example of creating a pseudo-correct answer for semantic segmentation using the BDD100K dataset [29-30]. Figure 3(b) overwrites the correct bounding boxes on the initial inference result of the semantic segmentation (imap). Figure 3(c) uses the latest inference result (lmap) to rewrite inside the bounding box of (b). For example, for the pedestrian on the right side of the image, the area ratio of the inference region to the bounding box is smaller than the threshold, so the entire bounding box becomes a pseudo answer. The inference region of the foreground (category targeted for object detection) outside from the bounding box is assigned by ignore_label and displayed in black.

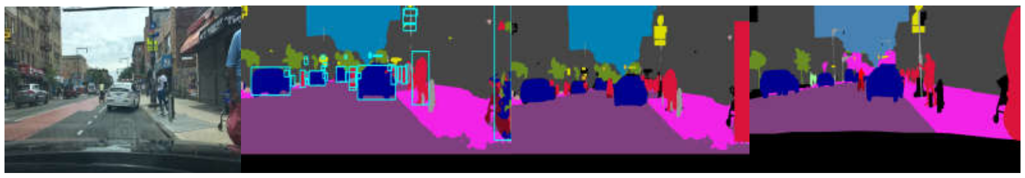


Fig.3 An example of creating a pseudo-correct answer for semantic segmentation using the BDD100K dataset

IV. EXPERIMENTS

A. Setup

Three multi-task networks for object detection, semantic segmentation, and disparity estimation were compared:

- I_T: Independent decoder, Transposed convolution.
- S_T: Shared decoder, Transposed convolution.
- S_SI (adopted method): Shared decoder, Separatable convolution + bilinear Interpolation.

The following three datasets were used.

- Cityscapes: has correct answers for all tasks.
- BDD100K: has correct answers for object detection and semantic segmentation.
- Kanazawa: has correct answer for object detection.

Cityscapes constitute a dataset comprising images captured from the driver's perspective using a dashcam across various cities in Germany. The dataset includes accurate annotations for both object detection and semantic segmentation [27-28]. Disparity estimation is supported by pseudo data generated through the SGM algorithm. The dataset comprises 2975 images for training and 500 images for validation. However, we utilized 2970 images for training and 497 images for validation, excluding images devoid of discernible objects for detection. The original resolution of the images is 2048 by 1024, but for our purposes, we resized them to 1024 by 512.

Initially, Cityscapes comprises 9 classes for object detection and 19 classes for semantic segmentation. In the context of semantic segmentation, we amalgamated categories such as "car," "truck," "bus," and "train" into a singular category named "car." Likewise, "bicycle" and "motorbike" were combined into "bike," and "people" and "rider" were consolidated into "people." For object detection, similar aggregation was conducted. The resultant six classes for object detection are as follows: "traffic_light," "traffic_sign," "person," "car," "bike," and "background." Accurate object detection data for "traffic_light" and "traffic_sign" were crafted from polygon data initially intended for semantic segmentation.

BDD100K stands as a comprehensive dataset [29-30], boasting an extensive collection of 100,000 driving videos compiled from over 50,000 rides, encompassing a grand total of more than 100 million frames. Although the original resolution is 1280 by 720, we opted for resized images at 1024 by 512.

Our preprocessing involved creating a 1280 by 640 image by selectively cropping the top segment, subsequently resizing it to 1024 by 512. The training set comprises 1500 images, with an additional 175 images reserved for validation. These images come equipped with accurate annotations for both object detection and semantic segmentation, albeit the subset with correct answers for both tasks constitutes a small fraction of the entire dataset. Notably, the imagery was predominantly captured during daylight conditions, except in instances of inclement weather, such as rain or snow. Figure 3(d) is a ground-truth example of semantic segmentation, but it is wrong in places. From what we have seen, the correct answer for BDD100K's semantic segmentation may contain obvious mistakes.

We curated the Kanazawa dataset, comprising images captured from the driver's perspective using a dashcam in Kanazawa, Japan. The dataset is equipped with accurate annotations for object detection. It encompasses 333 images designated for training and an additional 223 images set aside for validation, all with a resolution of 640 by 480. The object detection aspect encompasses six classes: "traffic_light," "traffic_sign," "person," "car," "bike," and "background."There are three methods for creating pseudo-correct answers for semantic segmentation.

- Method 1: “latest/latest''
  - Foreground (object to be detected)
    - Latest inference region
  - Background (object not targeted for detection)
    - Latest inference region
- Method 2: “bbox/latest''
  - Foreground
    - Within the correct bounding box
      - Obtain the area ratio of the latest inference region to the bounding box
      - If ratio>= threshold

        ```
          Latest inference region
        Else
          Bounding box
        ```
    - Outside the correct bounding box
      - ignore_label
  - Background
    - Latest inference region (updating every time)
- Method 3: “bbox/init'' (adopted method)
  - Foreground
    - Same as the foreground of the Method 2
  - Background
    - Initial inference region (no periodic updates)

In methods 1, 2, and 3, the method of creating a pseudo-correct answer for disparity estimation is the same as the background method of the respective semantic segmentation.

We conducted three experiments as follows:

- Experiment 1 involved training three networks (I_T, S_T, and S_SI) on Cityscapes dataset and comparing their accuracy. These networks are explained at the beginning of this section and in section 3, and S_SI is the adopted method. As far as the author knows, there is no other network for advanced driving assistance that performs the three tasks simultaneously in one network, so we compared the adopted method with two other methods with different decoder configurations.

- In Experiment 2, we used the S_SI network model trained in Experiment 1 as the initial model and performed transfer learning using BDD100K dataset with only correct object detection data. We used three transfer learning methods (“latest/latest”, “bbox/latest”, and adopted method “bbox/init”) and compared the accuracy of estimation results obtained by these methods. We compare the adopted method with the other two methods as an ablation study.
- In Experiment 3, we used the S_SI network model trained in Experiment 1 as the initial model and performed transfer learning with the “bbox_init'' method using Kanazawa dataset. We qualitatively evaluated the results of semantic segmentation and disparity estimation.

In the Experiment 1, the number of trainings was 200 epochs, the optimization method was Adam, the learning rate was 0.001, and the learning rate was reduced by a factor of 10 every 80 epochs.JPEG compression was adopted as part of the data augmentation. Since BDD100K and Kanazawa datasets contain JPEG-encoded images, while Cityscapes has uncompressed images, using JPEG compression during training can improve the accuracy when applying the learned models to BDD100K and Kanazawa datasets. Specifically, it can largely eliminate erroneous estimations occurring in the background portion of semantic segmentation.

In the Experiment 2 and 3, the number of transfer learning was 40 epochs, the learning rate was 0.0001, and the pseudo-correct answers were updated every 2 epochs. The threshold for methods “bbox/latest” and “bbox/init” was set to 0.5. In all experiments, we used the same loss function as SSD [2] for object detection, cross-entropy for semantic segmentation, and MAE (Mean Absolute Error) for disparity estimation. We used backpropagation to adjust the weight parameters of the loss functions for the three tasks and network parameters [31]. For network learning / inference programming, we used PyTorch.

B. Results

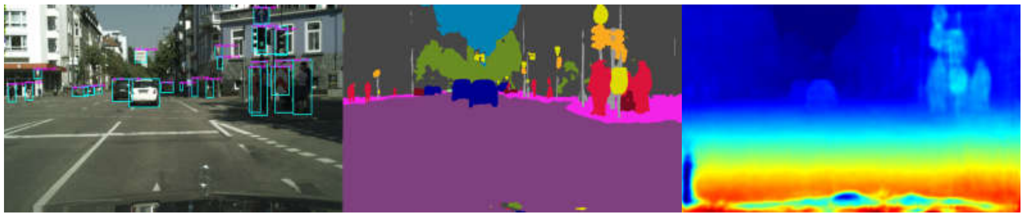
The results of Experiment 1 are presented in Table 2, where mAP (mean Average Precision) at IoU (Intersection over Union) =0.5 was used for object detection, mIoU (mean IoU) was used for semantic segmentation, and RDE (Relative Disparity Error) was used for disparity estimation. RDE was defined as follows:

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

Here, $GroundTruth_i$ is the disparity of the correct answer at point i, and $Prediction_i$ is the estimated value.The higher the mAP and mIoU and the lower the RDE, the more accurate the results.Despite the simplified network structure, including the fully shared decoder and the combination of depth-wise separable convolution and interpolation, we found that its accuracy was comparable to, or better than, the other networks tested. Figure 4 presents the inference results after training S_SI. From left to right, the Figure 4 shows the results of object detection, semantic segmentation, and disparity estimation. The results demonstrate that all inferences were successfully performed using a single network.

TABLE 2 Relationship between the network configuration and the inference accuracy

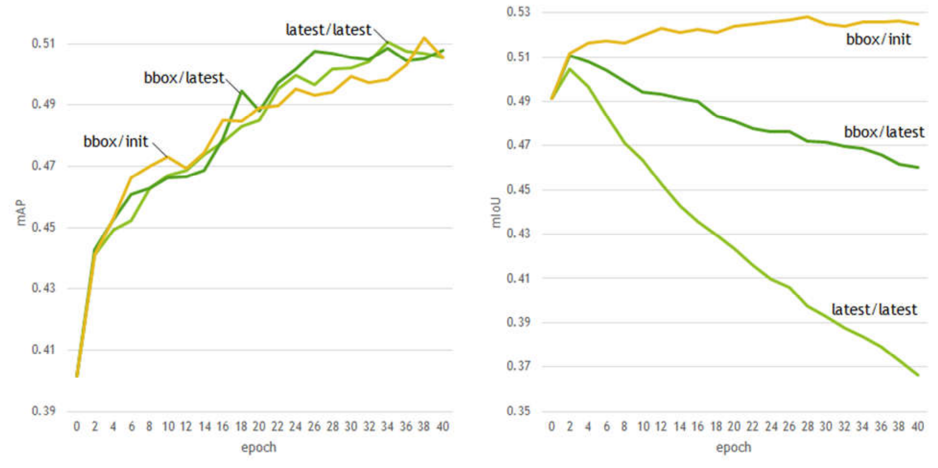
Network	mAP[%]	mIoU[%]	RDE[%]	#Operations[M]	#Prameters[K]
IT	45.9	**74.6**	10.8	51,031(100.0%)	27,711(100.0%)
ST	46.2	73.9	11.3	26,871(52.7%)	12,376(44.7%)
SSI	**47.0**	74.5	**10.5**	**13,107(23.3%)**	**4,634(16.7%)**



(a) Object detection ***(b) Semantic segmentation*** ***(c) Disparity estimation***

Fig.4 Inference results after training of the multi-task network

The results of Experiment 2 are shown in Figure 5. The left side is the result of object detection, the horizontal axis is the number of epochs, and the vertical axis is the detection accuracy (mAP). All methods are more accurate as the number of epochs increases.The accuracy of the adopted method "bbox/init" is equivalent to that of "bbox/latest" and that of "latest/latest". The right side is the result of semantic segmentation, the horizontal axis is the number of epochs, and the vertical axis is the segmentation accuracy (mIoU). It can be seen that the accuracy of the method "bbox/init" is higher than the other methods. The other methods improve the accuracy up to 2 epochs after the start of training, but the accuracy deteriorates after that.



(a) Object detection ***(b) Semantic segmentation***

Fig.5 Results of Experiment 2 with transfer learning from Cityscapes to BDD100K

Figure 6 shows the inference results by the network learned in Experiment 2.From the left, the inference results of "before transfer learning (model learned by Cityscapes)", "latest/latest", "bbox/latest", and "bbox/init". The top row is object detection, the middle row is semantic segmentation, and the bottom row is disparity estimation. It can be seen that the object detection accuracy after transfer learning is higher than that of "before transfer learning". Regarding semantic segmentation, "latest/latest" have no foreground objects other than cars and no small background objects. In “bbox/latest”, the pseudo-correct answer using the bounding box succeeded, and the object in the foreground did not disappear, but the poles in the background disappeared. Buildings and trees have rounded edges. The method “bbox/init” leaves poles and relatively sharp background edges. Regarding disparity estimation, “latest/latest” and “bbox/latest” are whitish overall and the contrast is low. “bbox/init” suppresses the decrease in contrast.

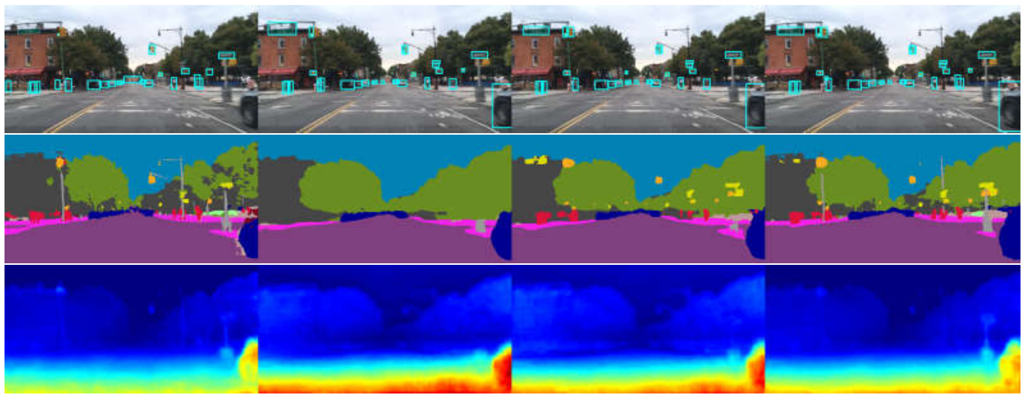
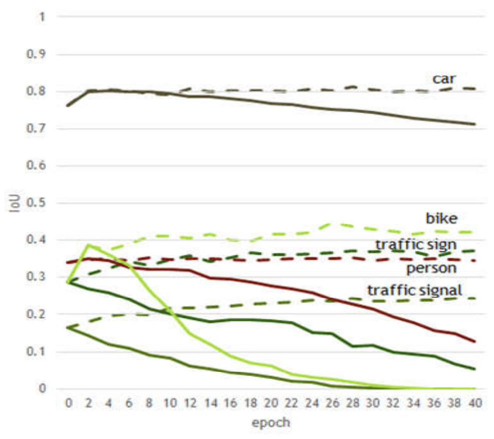

(a) Before training ***(b) latest/latest*** ***(c) bbox/latest*** ***(d) bbox/init***

Fig.6 Inference results of Experiment 2

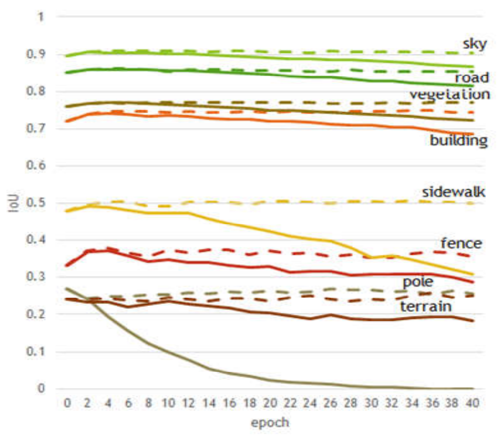
Figure 7 shows the segmentation accuracy by category. The left two are the result of “latest/latest”, the left is the foreground, and the right is the background. The two on the right are the results of “bbox/latest”, the left is the foreground, and the right is the background. In both cases, the solid line is the result of the specified method, and the dotted line is the result of the method “bbox/init”. In “latest/latest”, the accuracy of all categories deteriorates as the number of epochs increases. For “bbox/latest”, as the number of epochs increases, the foreground category is more accurate, but the background category is less accurate. The method “bbox/init” improves the accuracy of all categories.

Figure 8 shows the inference results of the network learned in Experiment 3.The upper row is the inference result by the model before transfer learning (after learning with Cityscapes), and the lower row is the result of “bbox/init”. From the left, the inference results of object detection, semantic segmentation, and disparity estimation. Transfer learning correctly recognizes traffic lights in both object detection and semantic segmentation and eliminates noise on the near side road in both semantic segmentation and disparity estimation.

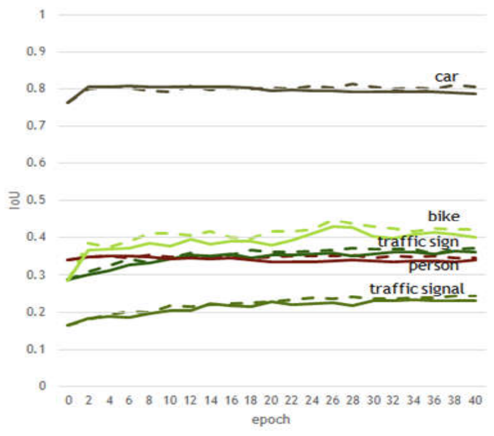
Traffic lights are vertical in Cityscapes and BDD100K, but horizontal in Kanazawa. Horizontal traffic lights are falsely detected and segmented as traffic signs before transfer learning in Figure 8. It can be seen that horizontal traffic lights are correctly detected and segmented by transfer learning of the method “bbox/init”. The adopted method can semantically segment object classes with different appearances. This is because our method teaches semantic segmentation the approximate location, size, and shape of each object.



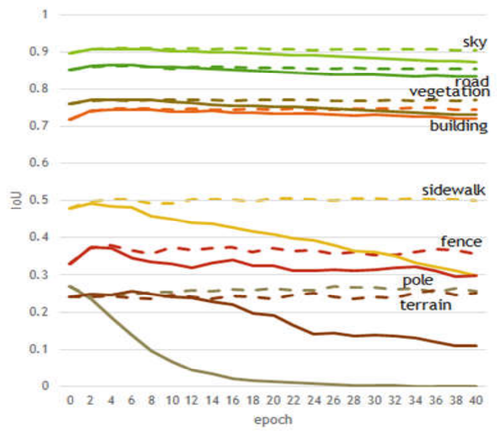
(a) latest/latest, foreground



(b) latest/latest, background



(c) bbox/latest, foreground

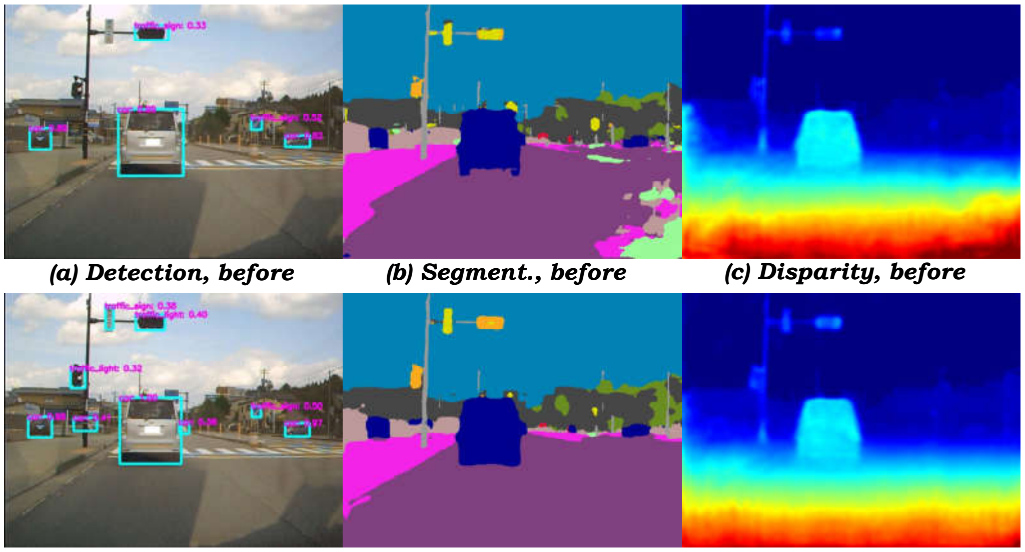


(d) bbox/latest, background

Fig.7 Semantic segmentation accuracy by category

C. Discussion

We will discuss why the segmentation accuracy temporarily improves in the early stages of transfer learning for many background categories. This phenomenon can be attributed to the removal of noise in the background through transfer learning. When training the multitasking network, we provide the correct answer for object detection and the pseudo-correct answer for semantic segmentation. If there is false detection in the background during object detection, the network will learn that part as the background instead of the foreground. Since the feature extraction part of the multi-task network is shared by all tasks, that part is also deduced as background for semantic segmentation. Since there is a high correlation between semantic segmentation and disparity estimation in the multi-task network, it can be inferred that the accuracy of disparity estimation is also improved.


(a) Detection, before ***(b) Segment., before*** ***(c) Disparity, before***

(d) Detection, after ***(e) Segment., after*** ***(f) Disparity, after***

Fig.8 Inference results of Experiment 3

We then discuss why the temporarily improved segmentation accuracy is then progressively degraded for the "latest/latest" and "bbox/latest" methods. Pixels in categories not subject to object detection are treated as background in object detection training. During the initial epochs, such misclassified pixels are correctly trained as background. However, the background segmentation accuracy does not improve further since the correct category cannot be given for training. Since the latest inference result is regarded as the correct answer, categories with high appearance probability, such as "building," are easier to infer, while categories with low appearance probability, such as "pole," are easier to miss. Figure 6 shows an example that the "pole" category disappears. In the adopted method "bbox/init", the initial inference result is fixed as the pseudo-correct answer of the background. Therefore, as shown in Figure 5(b), even if the number of epochs increases, the segmentation accuracy does not deteriorate.

V. IMPLEMENTATION

The multi-task CNN we developed was implemented on an embedded GPU board. The GPU board is Nvidia Jetson Xavier AGX. We used Ubuntu 18.04 LTS as the OS and JetPack 5.0.2 (L4T R35.1.0) as the machine learning container. We used Python as the programming language and PyTorch as the machine learning platform. The program inputs a 640 × 480 resolution video of Kanazawa, explained in the previous section, performs object detection, semantic segmentation, disparity estimation, and multi-object tracking, and displays all inference results on the display.

Figure 9 (a) shows a photograph of the real-time embedded system. The board with large black fins is Nvidia Jetson Xavier AGX, and the display shows three inference results by the multi-task CNN. Figure 9(b) and (c) show enlarged images of the object detection and tracking results. You can see that the detected object is assigned an ID and is being tracked.

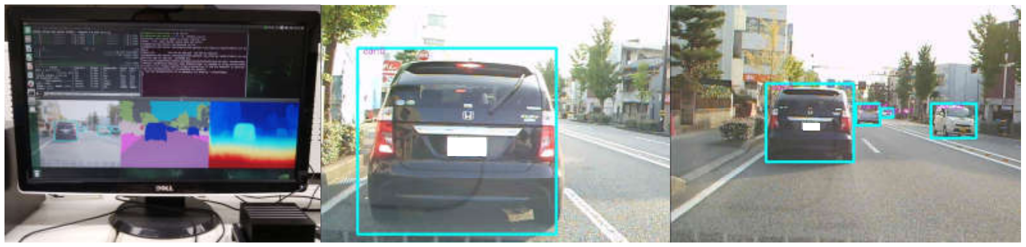


Fig.9 Real-time embedded system

To improve throughput, the program was parallelized and composed of three threads: video input, multi-task CNN inference, and inference result display (including multi-object tracking), and the threads were connected by FIFO.We used the "threading" and "queue" packages for this programming. The multi-task CNN runs on GPU version of PyTorch. During tracking, the prediction of the object position is assumed to be the same as the detected position, and no Kalman filter or the like is used. A Hungarian algorithm is used to match the detection and prediction of multi-object tracking. We used "linear_sum_assignment" from the "scypy.optimizer" package for this programming.

We measured the throughput of this program, and it was 18 fps. We measured the power consumption of the entire board with the "jtop" command, and it was 26W.The non-parallelized program had a throughput of 10 fps and a power consumption of 16 W. Parallelization increased throughput by 80%.

VI. CONCLUSION

In this paper, we developed a multi-task CNN for advanced driving assistance, which can perform object detection, semantic segmentation, and disparity estimation simultaneously. By sharing one encoder and one decoder employing depth-wise separable convolution and bilinear interpolation, we reduced the computational complexity to 23.3% and the number of parameters to 16.7% without sacrificing recognition accuracy. Furthermore, we adopted a semi-supervised learning method for the target domain, which utilizes hand-crafted correct answers for object detection and pseudo-correct answers for semantic segmentation created using bounding boxes. Experimental results demonstrate that this method improves not only object detection but also semantic segmentation accuracy. We implemented this network on an embedded GPU board. We also added a multi-object tracking function and achieved real-time performance of 18 fps at 26 W power consumption. Real-time implementation on lower power GPU boards is a future challenge.

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

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

REFERENCES

[1]. K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” Proceeding of International Conference on Learning Representations, 2015, pp. 1-14.

[2]. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu and A.C. Berg, “SSD: Single shot multibox detector,” 14th European Conference on Computer Vision, 2016, pp. 21-37.

[3]. O. Ronneberger, P. Fischer and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” 18th International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015, pp. 234-241.

[4]. A. Dosovitskiy, P. Fischer, E. Ilg, P.Hausser, C. Hazrbas and V. Golkov, “FlowNet: Learning optical flow with convolutional network,” IEEE International Conference on Computer Vision, 2015, pp. 2758-2766.

[5]. A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” arXiv:1704.04861 [cs.CV], 2017, pp. 1-9.

[6]. A. Kendall, Y. Gal and R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” IEEE International Conference on Computer Vision and Pattern Recognition, 2018, pp. 7842-7491.

[7]. T.H. Vu, H. Jain, M. Bucher, P. Cord and Perez, “ADVENT: Adversarial entropy minimization for domain adaptation in semantic segmentation,” IEEE International Conference on Computer Vision and Pattern Recognition, 2019, pp. 2512—2521.

[8]. U. Michieli, M. Biasetton, G. Agresti and P. Zanuttigh, “Adversarial learning and self-teaching techniques for domain adaptation in semantic segmentation,” IEEE Transactions on Intelligent Vehicles, vol. 5, no. 3, 2020, pp. 508-518.

[9]. S. Sankaranarayanan, Y. Balaji, A. Jain, S.N. Lim and R. Chellappa, “Learning from synthetic data: Addressing domain shift for semantic segmentation,” IEEE International Conference on Computer Vision and Pattern Recognition, 2018, pp. 3752-3761.

[10]. G. Li, W. Kang, Y. Liu, Y. Weim and Y. Yang, “Content-consistent matching for domain adaptive semantic segmentation,” 16th European Conference on Computer Vision, 2020, pp. 440—456.

[11]. P. Zhang, B. Zhang, T. Zhang, D. Chen, Y. Wang and F. Wen, “Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation,” IEEE international Conference on Computer Vision and Pattern Recognition, 2021, pp. 12409-12419.

[12]. M. Miyama, “Robust inference of multi-task convolutional neural network for advanced driving assistance by embedding coordinates,” 8th World Congress on Electrical Engineering and Computer Systems and Sciences, 2022, pp.1-9.

[13]. A. Gordon, H. Li, R. Jonschkowski and A. Angelova, “Depth from videos in the wild: unsupervised monocular depth learning from unknown cameras,” IEEE International Conference on Computer Vision, 2019, pp. 8976—8985.

[14]. B. Ummenhofer, and H. Zhou, “DeMoN: Depth and motion network for learning monocular stereo,” IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5622-5631.

[15]. C. Godard, O. M. Aodha and G. J. Brostow, “Unsupervised monocular depth estimation with left-right consistency,” IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6602-6611.

[16]. S. Ruder, "An overview of multi-task learning in deep neural networks," arXiv:1706.05098 [cs.LG], 2017, pp. 1-14.

[17]. M. Crawshaw, "Multi-task learning with deep neural networks: a survey," arXiv:2009.09796 [cs.LG], 2020, pp. 1-43.

[18]. Y. Zhang and Q. Yang, "An overview of multi-task learning," National Science Review, vol. 5, no. 1, 2018, pp. 30-–43.

[19]. L. Liebel and M. Korner, "Auxiliary Tasks in Multi-task Learning," arXiv:1805.06334 [cs.CV], 2018, pp. 1-8.

[20]. S. Chennupati, G. Sistu, S. Yogamani and S. Rawashdeh, "AuxNet: Auxiliary tasks enhanced semantic segmentation for automated driving," 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 2019, pp. 645-652.

[21]. H. Liu, D. Li, J.Z. Peng, Q. Zhao, L. Tian and Y. Shan, "MTNAS: Search multi-task networks for autonomous driving," 15th Asian Conference on Computer Vision, 2020, pp. 670—687.

[22]. P. Guo, C. Y. Lee and D. Ulbricht, "Learning to branch for multi-task learning," 37th International Conference on Machine Learning, vol. 198, 2020, pp. 1-12.

[23]. M. Reginthala, Y. Iwahori, M.K. Bhuyan, Y. Hayashi, W. Achariyaviriya and B. Kijsirikul, "Interdependent multi-task learning for simultaneous segmentation and detection," 9th International Conference on Pattern Recognition Applications and Methods, 2020, pp. 1-9 .

[24]. W.H. Li, X. Liu and H. Bilen, "learning multiple dense prediction tasks from partially annotated data," IEEE International Conference on Computer Vision and Pattern Recognition, 2022, pp. 18879-18889.

[25]. Y. Wang, Y. H. Tsai, W. C. Hung, W. Ding, S. Liu and M. H. Yang," Semi-supervised multi-task learning for semantics and depth," IEEE Winter Conference on Applications of Computer Vision, 2022, pp. 2505-2514.

[26]. F. Lu, H. Yu and J. Oh, "Domain adaptive monocular depth estimation with semantic information," arXiv:2104.05764v1 [cs.CV] , 2021, pp. 1-7.

[27]. 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," IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213-3223.

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

[29]. F. Yu, H. Chen,X.Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan,T. Darrell, "BDD100K: A diverse driving dataset for heterogeneous multitask learning," arXiv:1805.04687 [cs.CV], 2020, pp. 1-10.

[30]. https://www.bdd100k.com/

[31]. https://github.com/Xilinx/Vitis-AI/tree/master/models/AI-Model-Zoo, Performance on ZCU104, 103 multi_task_v3.