# Attribute-Aware Loss Function for Accurate Semantic Segmentation Considering the Pedestrian Orientations

**Mahmud Dwi SULISTIYO**[†,††a)], *Nonmember*, **Yasutomo KAWANISHI**[†b)], **Daisuke DEGUCHI**[†††c)], *Members*, **Ichiro IDE**[†d)], *Senior Member*, **Takatsugu HIRAYAMA**[††††e)], *Member*, **Jiang-Yu ZHENG**[†††††f)], *Nonmember*, *and* **Hiroshi MURASE**[†g)], *Fellow*

**SUMMARY** Numerous applications such as autonomous driving, satellite imagery sensing, and biomedical imaging use computer vision as an important tool for perception tasks. For Intelligent Transportation Systems (ITS), it is required to precisely recognize and locate scenes in sensor data. Semantic segmentation is one of computer vision methods intended to perform such tasks. However, the existing semantic segmentation tasks label each pixel with a single object's class. Recognizing object attributes, e.g., pedestrian orientation, will be more informative and help for a better scene understanding. Thus, we propose a method to perform semantic segmenta-tion with pedestrian attribute recognition simultaneously. We introduce an attribute-aware loss function that can be applied to an arbitrary base model. Furthermore, a re-annotation to the existing Cityscapes dataset enriches the ground-truth labels by annotating the attributes of pedestrian orientation. We implement the proposed method and compare the experimental results with others. The attribute-aware semantic segmentation shows the ability to outperform baseline methods both in the traditional object segmentation task and the expanded attribute detection task.

*key words: semantic segmentation, attribute-aware, pedestrian orientation, deep neural network*

## 1. Introduction

Computer vision plays an important role in perception tasks and has been widely utilized for various purposes such as autonomous driving [1], satellite imagery sensing [2], biomedical imaging [3], [4], face recognition [5], and robotics navigation [6]. For Intelligent Vehicle (IV) applications, it is very important to have a comprehensive understanding of surrounding scenes captured by in-vehicle cameras. For example, weather conditions can be predicted by recognizing road and sky [7] for vehicle safety. Ground marking detection is useful for vehicle localization purposes [8], [9] and it is also necessary to mark drivable roads for lane instructions in a smart car [10]. Moreover, recognizing the behavior of pedestrians is also important to plan the vehicle actions [11].

Obstacle detection is important for an autonomous vehicle to avoid collisions by breaking, lowering or keeping the speed. However, understanding the situation around the obstructing objects instead of only focusing on the obstacles will allow us to handle even more complicated situations. Semantic segmentation, as one of computer vision tasks, classifies all desired things in an input image and locates their areas, ultimately in pixel-level precision. This provides more decisive information than just detection to help an autonomous vehicle in achieving a better scene understanding; Semantic segmentation provides more environmental information to instruct the vehicle with an immediate breaking or further to recommend an optimal path for collision avoidance.

Only in a few years, semantic segmentation has attracted many studies in building deep neural network models. They include Fully Convolutional Network [12], SegNet [1], Bayesian SegNet [13], PSPNet [14], Mask R-CNN [15], ICNet [16], and the DeepLab from versions v1 to v3+ [17]–[19]. Several datasets consisting of input images and the ground-truth annotations such as CamVid [20], KITTI [21], and Cityscapes [22] are also publicly available for benchmarking purpose. The latter is very popular in the semantic segmentation task for autonomous driving, pushing many researchers and practitioners in such competition to reach higher ranks.

However, most of the existing semantic segmentation methods only learn to recognize object types such as road, building, car, and person. For autonomous vehicles, addi-tional information describing an object in details such as its attributes could help the better understanding of scenes [11]. Figure 1 illustrates the ultimate goal of this newly proposed task should we apply it to some moving objects. In addition, breaking down a class into several sub-classes can reduce the variation of class instances to help the classification process. Therefore, as our objective, we aim at improving the segmen-tation accuracy in pixel-level by considering attribute infor-mation attached to particular objects. This paper exploits the attribute-aware semantic segmentation, with the simultaneous combination of attribute recognition and semantic

Manuscript received March 7, 2019.
Manuscript revised July 3, 2019.

[†]The authors are with the Graduate School of Informatics, Nagoya University, Nagoya-shi, 464-8601 Japan.
[††]The author is also with the School of Computing, Telkom University, Indonesia.
[†††]The author is with the Information Strategy Office, Nagoya University, Nagoya-shi, 464-8601 Japan.
[††††]The author is with the Institutes of Innovation for Future Society, Nagoya University, Nagoya-shi, 464-8601 Japan.
[†††††]The author is with the Department of Computer Science, Indiana University-Purdue University Indianapolis, USA.

a) E-mail: mahmuds@murase.is.i.nagoya-u.ac.jp
b) E-mail: kawanishi@i.nagoya-u.ac.jp
c) E-mail: ddeguchi@nagoya-u.jp
d) E-mail: ide@i.nagoya-u.ac.jp
e) E-mail: takatsugu.hirayama@nagoya-u.jp
f) E-mail: jzheng@iupui.edu
g) E-mail: murase@i.nagoya-u.ac.jp

This is the author's manuscript of the article published in final edited form as:
Sulistiyo, M. D., Kawanishi, Y., Deguchi, D., Ide, I., Hirayama, T., Zheng, J.-Y., & Murase, H. (2020). Attribute-Aware Loss Function for Accurate Semantic Segmentation Considering the Pedestrian Orientations. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E103.A(1), 231–242. https://doi.org/10.1587/transfun.2019TSP0001

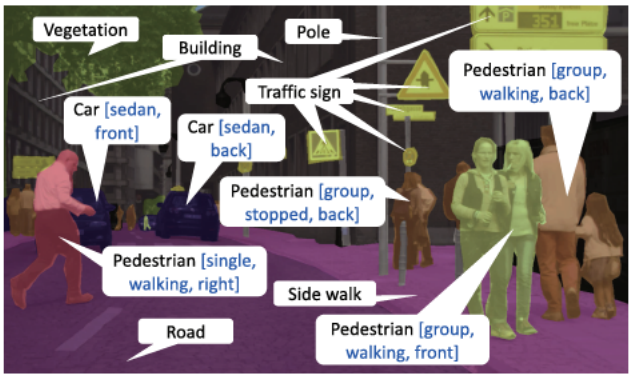


**Fig. 1** A general view illustrating the purpose of the newly proposed attribute-aware semantic segmentation for a better scene understanding.

segmentation. It is implemented as a multi-task learning paradigm and is expected to be able to enhance environment perception tasks for ITS purposes. In a preliminary work [23], this task was designed in three successive stages, i.e., semantic segmentation, object detection, and attribute recognition, but our current work proposes an approach to train the model in an end-to-end process. The concept was initially introduced in our previous paper [24], while this paper reveals additional aspects along with a newly extended dataset and base models.

This paper focuses on pedestrian detection and selects body orientation as its attribute. Simply detecting the existence of a pedestrian is necessary for an autonomous vehicle to take immediate actions before collision. However, we con-sider that the pedestrian's body orientation is an important sign of walking direction to anticipate possible risks from a distance. The information of body orientation becomes a clue to the pedestrian movement and hence helps potential collision avoidance. They are useful for a motion planning module to plan a vehicle's future actions. Although pedestri-ans might suddenly change their direction or movement state, an accurate segmentation system will recognize the changes in pedestrian attribute and make the decision again. This ex-plains why simply knowing the existence of a pedestrian is sometimes inadequate for a smart vehicle. Predicting the op-timal path is difficult without gaining pedestrian attributes; Thus, the attribute-aware semantic segmentation task is more beneficial to ITS applications.

Here, we also show the re-annotation process on an existing dataset, documenting our effort to provide the ground-truth required in this task. Although *only* pedestrian orientation is added as a new semantic information, it gives hint and insight to other object attributes when the method is extended. In summary, our contributions are as follows:

1. We introduce a new concept of attribute-aware semantic segmentation that enables an end-to-end training process.
2. The proposed attribute-aware loss function is able to treat both object classes and attribute classes in the same manner and hence is applicable to all base models.
3. We construct the CityWalks dataset with an attribute annotation of pedestrian orientations as an extension to the Cityscapes dataset.
4. We show experimental results proving that the proposed approach achieves a better performance over the baseline methods and is capable of providing attribute information in the segmented results.

The rest of this paper explores related works in Sect. 2, describes details of the proposed method and its implementation in Sect. 3, and explains the dataset construction in Sect. 4. Experimental results and analyses are discussed in Sect. 5 and Sect. 6 respectively, followed with a conclusion in Sect. 7.

## 2. Related Work

In recent years, many studies have been conducted to develop deep neural network models for the semantic segmentation task. Liu et al. provides a comprehensive review to summarize the recent progress of semantic segmentation approaches and datasets [25]. For the attribute-aware semantic segmentation task, which can basically be regarded as a multi-task learning goal, we have been conducting a study despite some limitations. A part of work regarding this initial concept has been published in paper [24].

### 2.1 Semantic Segmentation Datasets

One of publicly available semantic segmentation datasets, namely Cityscapes [22], [26], is constructed mainly for autonomous driving research and development purpose. It adequately depicts the complexity of real-world urban scenes and therefore exceeds previous efforts in terms of dataset size, annotation richness, and scene diversity. The vehicle-mounted cameras, while moving, recorded videos of traffic scenes during the daytime from 50 cities in Germany and some neighboring countries. The data collection covers season changes during several months-span including spring, summer, and fall. The captured road scenes include cloudy and sunny weathers, but do not include adverse weather con-ditions such as heavy rain or snow [22]. The first row of Fig. 6 shows examples of the Cityscapes images. A number of frames are then extracted from the recorded videos, which results in 5,000 images with high quality pixel-level annota-tions and 20,000 additional images with coarse annotations. The fine-annotated images are divided into 2,975 images for training and 500 images for validation, while the remain-ing 1,525 images are for testing that is available to the public without annotations as they are designated for benchmarking purpose through the Cityscapes Web site[†]. Each of the three split sets is carefully organized to have equal distributions in some properties such as the city crowd levels and geo-graphical locations. However, there is no information about the distribution of the data for each weather condition. The dataset challenges researchers to classify 19 different classes in the pixel-level semantic labeling task.

[†]https://www.cityscapes-dataset.com/benchmarks/

### 2.2 Semantic Segmentation Models

A Fully Convolutional Network (FCN) was introduced by Long et al. as a model of semantic segmentation trained in an end-to-end process [12]. FCN extends the existing models containing fully-connected layers for classification to the semantic segmentation task. This model takes an arbitrary size of input image and produces a pixel-wise segmentation map. To reduce the computational cost, Badrinarayanan et al. presented SegNet [1] with two deep convolutional streams, namely encoder and decoder. It adapts the VGG16 network [27] without employing fully-connected layers. Each encoder layer in the model has a corresponding decoder layer and is ended with a final pixel-wise classification layer. It was then enhanced by Kendall et al. as Bayesian SegNet [13] to obtain better performance by considering a probabilistic element.

The Pyramid Scene Parsing network (PSPNet) architec-ture by Zhao et al. [14] uses ResNet [28] as its backbone, fol-lowed by a pyramid module pooling. It extracts features from four different scales, and then merge them after up-sampling each feature to the same resolution. This pipeline effec-tively produces combined multi-scale feature maps without performing multiple convolutions. It showed its remark-able performance on PASCAL VOC [29] and Cityscapes datasets [22]. Meanwhile, the Mask R-CNN by He et al.[15] arose with a new conceptual framework. It performs objects detection in parallel with segmentation mask cre-ation for each object instance. It modifies the Faster R-CNN [30] model by adding a segmentation branch to work together with the existing bounding-box detection branch. Very recently, DeepLab-v3+ [19] was developed by Chen et al. from the previous DeepLab-v3 [18] version by em-ploying an encoder-decoder structure. The encoder module utilizes multi-scale atrous convolution to encode contextual information at multiple scales, while the decoder effectively improves the segmentation results.

### 2.3 Multi-Task Learning

Multi-Task Learning (MTL) is a machine learning paradigm that leverages useful information by solving multiple related tasks simultaneously. Zhang and Yang made an extensive survey [31] on numerous MTL algorithms, classified them into some categories, and summarized the results including applications in the area of computer vision. Meanwhile, Ruder described that for the deep learning context, MTL would typically be performed with either two concepts: *hard* or *soft parameter sharing* [32].

Mask R-CNN [15] is one of deep learning models that applies the MTL paradigm by adding the functionality of instance segmentation. The idea is simple but powerful enough to place top ranks in three challenges including instance segmentation, object detection, and person key-point detection. Recently, the MTL paradigm is widely implemented for var-ious purposes. An MTL-based CNN was presented by Lu et al. [33] to address food segmentation, recognition, and volume estimation, which successfully outperforms the baseline methods. Another MTL architecture with heavy sharing of weights and features was introduced in [34] to perform four tasks: 2D pose estimation, 3D pose estimation, 2D action recognition, and 3D action recognition.

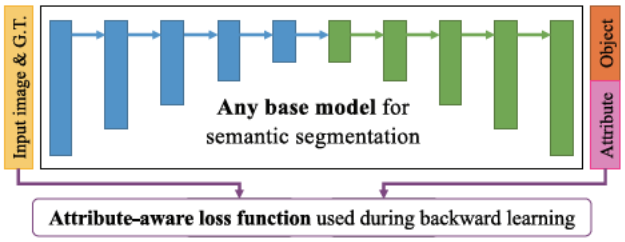


**Fig. 2** Our general approach of multi-task learning for attribute-aware semantic segmentation; Unlike other existing works, an attribute-aware loss function is proposed to deal with the attributes (either exists or not), and applicable to any arbitrary base model.

In some references related to MTL, each task is treated in a different manner and solved by modifying the classifier model. Commonly, the loss function can handle multi-task conditions of the input image, yet is not considered for non-multi-task conditions simultaneously. The concept proposed in this paper will overcome this limitation. Concretely, we design a loss function that is general as in Fig. 2 to be applied to all input images through the same calculation flow without any change of network architecture.

## 3. Attribute-Aware Semantic Segmentation

We propose a new task named "attribute-aware semantic segmentation" to yield a richer output than a conventional pixel-wise labeling task, which can be implemented as an MTL framework. Through this task, in addition to recognizing the object class of each pixel, we can also obtain the attribute information associated with the object. Additionally, this newly proposed task is also aimed at improving the segmentation performance. Typically, the instances of a particular class, for example, person, have a high variance in shape and appearance. This makes an algorithm difficult to classify heterogeneous instances into a correct class. Our idea is to divide such a class into several sub-classes according to its attribute values. This will reduce the diversity in each sub-class and thereby simplify the classification task.

Instead of reconstructing the model architecture, the proposed approach basically deals with the training process. Since the original class, i.e. person, might be necessary if the attribute value looks unknown, we simply add the sub-class set corresponding to the attribute values at the same level of object classes in the base model's output layer, yet maintain each attribute as a *still-derived* class using a loss function. Thus, any base semantic segmentation model can be made use of in the proposed method.

### 3.1 Designing an Attribute-Aware Loss Function

In the semantic segmentation task, a pixel-wise cross-entropy loss has been the most commonly used loss function so far. The final layer of a segmentation model produces output tensor channels as many as the number of object classes. Each output channel representing a class contains the probabilities of all pixels to be classified into that c lass. The loss function ($L$) examines every pixel individually according to its class predictions ($\rho_k$) by comparing a depth-wise pixel vector to the encoded target vector ($y_k$) as follows.

$$L = \frac{1}{N}\sum_{i}^{N} \ell_i \tag{1}$$

$$\ell_i = -\sum_{k}^{|C|} y_k \log(\rho_k) \tag{2}$$

The log loss for the $i$-th pixel is calculated on the $k$-th class and then summed over all possible classes ($C$). This scoring is repeated over all $N$ pixels and averaged to obtain the overall image loss. To improve the object segmentation result, attribute information is considered to be important. By dividing a particular class into sub-classes according to its distinct attribute values, a more appropriate pixel-wise labeling can be obtained. Therefore, we need to consider attribute loss in addition to object class loss when evaluating the prediction of pixels containing attribute information.

$$\ell_i = \begin{cases} \ell_i^o + \ell_i^a & \text{if the } i\text{-th pixel has an attribute} \\ \ell_i^o & \text{otherwise} \end{cases} \tag{3}$$

For example, suppose that three objects form the target classes $C = \{X, Y, Z\}$, and class $X$ has two attribute classes $A_X = \{X_1, X_2\}$. Pixel area or instances labeled as the attributed object $X$ will be valued additionally to either of those attributes, $X_1$ or $X_2$. However, in some conditions due to visual limitation, we cannot always assign every instance of an attributed class with an attribute value. To allow this, there will be some pixels annotated as class $X$ but with an unknown attribute. Label $X$ remains as a target class and the set of targets is obtained by combining $C$ and $A_X$ (or just $A$) to be $C \cup A = \{X, Y, Z, X_1, X_2\}$. This clarifies Eq. (3) that $\ell_i^o$ and $\ell_i^a$ are distinguished based on which the class set is used such that $\ell_i^o = -\Sigma_k^{|C|} y_k \log(\rho_k^o)$ and $\ell_i^a = -\Sigma_k^{|C\cup A|} y_k \log(\rho_k^a)$.

With the general class setting described above, we can calculate two independent losses: $L_o$ penalizes every pixel across all object classes and $L_a$ does so to all combined object and attribute classes. These loss functions allow us to evaluate object and attribute classes simultaneously while maintaining the dependency of attributes to an object. To solve the problem in one calculation flow, we combine the two losses via key parameters of weights, namely $\hat{\beta}_o$ and $\hat{\beta}_a$, to construct the attribute-aware loss function:

$$L = \frac{\hat{\beta}_o}{\hat{\beta}_o + \hat{\beta}_a} L_o + \frac{\hat{\beta}_a}{\hat{\beta}_o + \hat{\beta}_a} L_a, \tag{4}$$

where the values for $\hat{\beta}_o$ and $\hat{\beta}_a$ can be arbitrarily assigned any real number to represent the amount of contributions from $L_o$ and $L_a$ toward the final loss calculation. We then simplify $\frac{\hat{\beta}_o}{\hat{\beta}_o+\hat{\beta}_a}$ and $\frac{\hat{\beta}_a}{\hat{\beta}_o+\hat{\beta}_a}$ with $\beta_o$ and $\beta_a$, respectively, to equally re-form the formula as follows:

$$L = \beta_o L_o + \beta_a L_a. \tag{5}$$

According to Eq. (4), $\beta_o$ and $\beta_a$ in Eq. (5) should satisfy the condition of $\beta_o + \beta_a = 1$. Since $L$ is proportional to $\ell$ as defined in Eq. (1), the loss for each pixel is

$$\ell_i = \beta_o \ell_i^o + \beta_a \ell_i^a. \tag{6}$$

By replacing $y_k \log(\rho_k)$ in Eq. (2) with $\mathcal{E}_k$ for simplification, we then break down Eq. (6) with object and attribute classes mentioned in the above case as follows:

$$\begin{aligned} \ell_i &= \beta_o \Sigma_k^{|C|} \mathcal{E}_k^o + \beta_a \Sigma_k^{|C\cup A|} \mathcal{E}_k^a, \\ &= \beta_o(\mathcal{E}_X^o + \mathcal{E}_Y^o + \mathcal{E}_Z^o) + \\ &\quad \beta_a(\mathcal{E}_X^a + \mathcal{E}_Y^a + \mathcal{E}_Z^a + \mathcal{E}_{X_1}^a + \mathcal{E}_{X_2}^a). \end{aligned}$$

The proposed framework puts object and attribute classes together in the output layer so it is easy to calculate $L_a$. Meanwhile, to calculate $L_o$, the likelihood $\rho_k^o$ for each object class can be copied from $\rho_k^a$, since $X$, $Y$, and $Z$ in the calculation for $L_a$ are designated to the same classes in the calculation for $L_o$. This makes $\rho_X^o = \rho_X^a$ that implies $\mathcal{E}_X^o = \mathcal{E}_X^a$, and the same also applies to $\mathcal{E}_Y^o$ and $\mathcal{E}_Z^o$ to continue and simplify the derivation as

$$\begin{aligned} \ell_i &= \beta_o(\mathcal{E}_X^a + \mathcal{E}_Y^a + \mathcal{E}_Z^a) + \\ &\quad \beta_a(\mathcal{E}_X^a + \mathcal{E}_Y^a + \mathcal{E}_Z^a + \mathcal{E}_{X_1}^a + \mathcal{E}_{X_2}^a), \\ &= \beta_o\mathcal{E}_X^a + \beta_o\mathcal{E}_Y^a + \beta_o\mathcal{E}_Z^a + \\ &\quad \beta_a\mathcal{E}_X^a + \beta_a\mathcal{E}_Y^a + \beta_a\mathcal{E}_Z^a + \beta_a\mathcal{E}_{X_1}^a + \beta_a\mathcal{E}_{X_2}^a, \\ &= (\beta_o + \beta_a)\mathcal{E}_X^a + (\beta_o + \beta_a)\mathcal{E}_Y^a + (\beta_o + \beta_a)\mathcal{E}_Z^a + \\ &\quad \beta_a\mathcal{E}_{X_1}^a + \beta_a\mathcal{E}_{X_2}^a, \\ \ell_i &= \mathcal{E}_X^a + \mathcal{E}_Y^a + \mathcal{E}_Z^a + \beta_a\mathcal{E}_{X_1}^a + \beta_a\mathcal{E}_{X_2}^a, \\ &= \sum_{k}^{|C|} \mathcal{E}_k^a + \beta_a \sum_{k}^{|A|} \mathcal{E}_k^a. \end{aligned} \tag{7}$$

The formula derivation along with the aforementioned scheme and class setting ensures that the proposed loss function with two entire-image losses in Eq. (4) will penalize each pixel with object and attribute losses as defined in Eq. (6). Hence, it should be able to treat both attributed and non-attributed objects simultaneously in one calculation flow.

Figure 3 illustrates the proposed loss function and compares it to the basic semantic segmentation, while the remaining part of this section will explain the implementation especially in the study discussed in this paper.

### 3.2 Implementation of the Semantic Segmentation

The difference between baseline methods and the proposed method lies inside the loss calculation process in the training phase, while other processes remain basically the same.

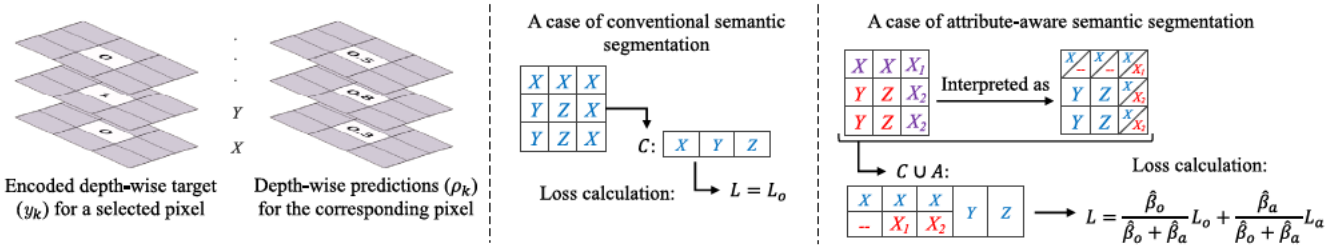


**Fig. 3** Illustration of the proposed method in Sect. 3, and comparison of conventional semantic segmentation and the proposed attribute-aware semantic segmentation via a simple case, where objects and attributes are $C = \{X, Y, Z\}$ and $A_X = \{X_1, X_2\}$, respectively.

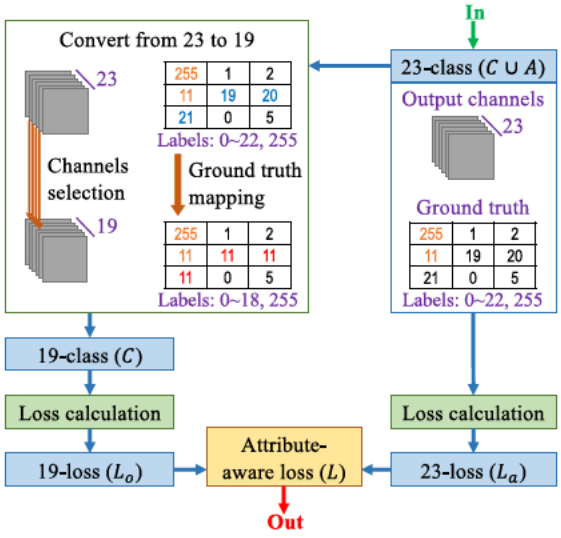


**Fig. 4** Block diagram of the proposed loss calculation in the semantic segmentation network; pixels with class "person" are annotated as 11, while ignored pixels are annotated as 255.

**Algorithm 1:** The attribute-aware loss function

**Function** `GtMapping`($gt$):
  $gt$[($gt$=`attribute_id`)] ← `person_id`
  **return** $gt$

**Function** `ChannelSelection`($out$):
  $num$ ← `num_all_class` − `num_attribute`
  $out$ ← $out$[0:$num$]
  **return** $out$

**Function** `StandardLoss`($gt$, $out$):
  **return** `CrossEntropyLoss`($gt$, $out$)

**Function** `AttributeAwareLoss`($gt$, $out$, $\hat{\beta}_o$, $\hat{\beta}_a$):
  $gt_o$ ← `GtMapping`($gt$)
  $out_o$ ← `ChannelsSelection`($out$)
  $L_o$ ← `StandardLoss`($gt_o$, $out_o$)
  $L_a$ ← `StandardLoss`($gt$, $out$)
  **return** $(\hat{\beta}_o / (\hat{\beta}_o + \hat{\beta}_a))\, L_o + (\hat{\beta}_a / (\hat{\beta}_o + \hat{\beta}_a))\, L_a$

Figure 4 depicts the block diagram explaining the modification applied to the standard loss function. In the Cityscapes challenge, images are annotated with 19 object classes. We add four attribute values correspondingly to pedestrian body orientations (*back*, *right*, *front*, and *left*), resulting in the ground-truth labels re-annotated in 23 classes. Before counting $L_o$ and $L_a$ separately, we convert the ground-truth maps and output channels from 23-class to 19-class mode. In the ground-truth conversion, the proposed algorithm maps all four orientation labels to one-person label, while in the outputs conversion, it selects channels corresponding to 19 object classes and omits any attribute channel.

Algorithm 1 shows this process. We can see that the input parameters (i.e. output channels and target map) between the standard and the proposed loss functions are the same. Therefore, in terms of algorithm complexity, the proposed method is not very different compared to the standard one. The actual codes for training and testing were based on Deng's implementation [35] obtained from a GitHub repository†, of which some modules were modified to implement the proposed algorithm properly.

†https://github.com/zijundeng/pytorch-semantic-segmentation/

## 4. Construction of the CityWalks Dataset

There are numerous datasets publicly available for the semantic segmentation challenge, but none of them includes attribute information together with the object class in the given labels. Therefore, for the proposed attribute-aware semantic segmentation task, a modification on top of an existing dataset is required. Here, we use the Cityscapes dataset as the base and re-annotate the labels as needed by dividing a certain object into a number of attribute labels.

### 4.1 Pedestrian Orientation as Attributes

As mentioned in Sect. 1, we choose the class *person* as a target because a driver must pay attention to moving objects, especially a pedestrian, while driving. Since the walking direction is also important for the driver to determine his/her actions, body orientation is designated as an attribute to en-rich the class of *person*. We introduced the initial concept of our work in [24], but it was not effective enough to achieve good accuracy in classifying the orientations. In this paper, we consider four orientation classes as considered in devel-oping an autonomous vehicle [36], including *back* (0°), *right* (90°), *front* (180°), and *left* (270°). Information regarding

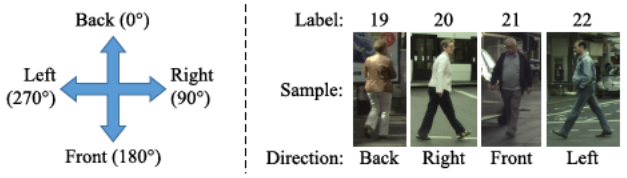


**Fig. 5** Guidance for four orientations annotation.

**Table 1** Statistics of the CityWalks dataset.

| | Training set | Validation set |
|---|---|---|
| #image | 2,975 | 500 |
| #image with *person* | 2,345 | 402 |
| #image with 4 orientation attributes | 2,083 | 371 |
| %image with *person* in the dataset | 78.82 % | 80.40 % |
| %image with attributes in the dataset | 70.02 % | 74.20 % |
| %*person* pixels in the dataset | 1.08 % | 1.15 % |
| %attribute pixels in the dataset | 1.03 % | 1.09 % |

these directions of a pedestrian is meaningful and especially important for ITS purposes, because it can provide information on whether a pedestrian is intending to walk across the street, is aware of the vehicle, or walks on a sidewalk along the street, and so on. We use the guidance as shown in Fig. 5 to re-annotate *person* instances in the Cityscapes dataset.

### 4.2 Extending Dataset Annotation to the CityWalks Dataset

Unlike the preliminary work [23] that utilizes the CityPersons [37] bounding box, we completely re-annotate the Cityscapes dataset's ground-truth labels manually. We achieved this re-annotation in a much better quality by completing missing pedestrians as well as small body parts such as toe, hand, and head. In addition, through this, the number of pixels annotated with attribute values drastically increased for training and evaluation purposes.

We performed the re-annotation using a conventional image editor. It took around 1.5 minutes per image for 'easy' or 'moderate' cases including those that covered quite a few persons in one image. However, for 'difficult' images which contains a crowd of pedestrians with various orientations, it took approximately ten minutes per image.

Statistics on this new annotation is summarized in Table 1, while some visual results before-and-after re-annotation are shown in Fig. 6. The annotated colors corresponding to all classes are visualized in Fig. 7. We name the dataset CityWalks, a modified Cityscapes dataset with additional labels corresponding to pedestrian orientations, and will make it available to the public.

In addition to new annotations, this process also provides some corrections to the original ground-truth labels given in the Cityscapes dataset. The mistakes we found include several labels swapped between objects, missed or ignored pedestrians, and inconsistent labels for items attached to a pedestrian such as umbrella, bag, etc. Figure 8 shows some samples of incorrect ground-truth labels found in the Cityscapes dataset.


**Fig. 6** Comparing ground-truth annotations of scenes in the original Cityscapes and the extended CityWalks datasets.

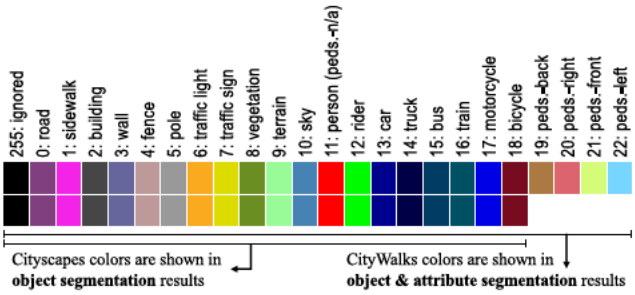


**Fig. 7** Color annotations for Cityscapes and CityWalks datasets.

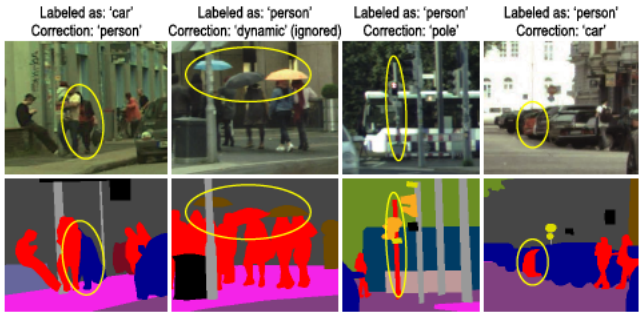


**Fig. 8** Samples of incorrect annotations in the original Cityscapes dataset.

## 5. Experimental Results

### 5.1 Methods Comparison

We evaluated the following methods in our experiments:

(1) **FCN8s** [12] and **PSPNet** [14] are the baseline methods with the same base semantic segmentation models built in [12] and [14]. We re-train each of them using the standard loss function and the Cityscapes dataset to output 19 object classes.

(2) **FCN8s.23cls** and **PSPNet.23cls** are same as the baseline methods (1) but trained with CityWalks dataset to

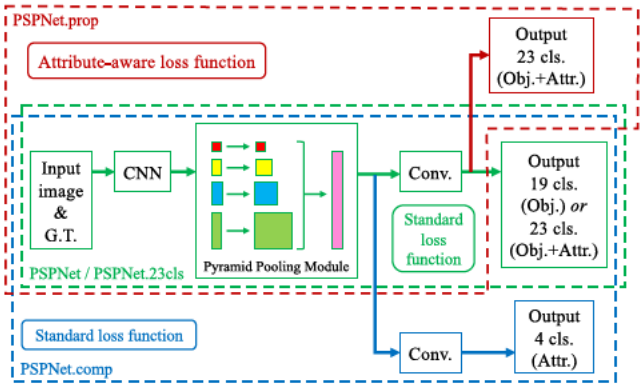


**Fig. 9** Baseline (PSPNet), proposed, and comparative methods.

output 23 classes.

(3) **PSPNet.comp** is a comparative method representing the MTL-based architecture; It modifies PSPNet.23cls by splitting convolution layers to separately output between object and attribute segmentation tasks. With the same training set as used in (2), each object and attribute branch uses the standard loss function to output (19+4) classes.

(4) **FCN8s.prop** and **PSPNet.prop** are our proposed methods which utilize the same models as in (2), but apply the proposed attribute-aware loss function during the training process. The training set and the number of output classes are the same as (2). The default values for $\hat{\beta}_o$ and $\hat{\beta}_a$ in FCN8s.prop are 0.7 and 0.3, respectively, while for PSPNet.prop, they are equally set as 0.5.

Figure 9 depicts the four methods used in our experiments with PSPNet as the base model. In the training stage, we used a batch size of 4 and the maximum iteration of 50,000 or equivalent with 66 epochs. All images and corresponding ground-truth labels were resized from 2,048 × 1,024 pixels to 1,024 × 512 pixels. We set 512 × 512 pixels as the window cropping size.

### 5.2 Segmentation Performance

The segmentation performance of each method for local eval-uation is calculated based on metrics including global Ac-curacy (gAcc), per-class Accuracy (cAcc), Intersection over Union (IoU), and mean IoU (mIoU). Moreover, we consider IoU for the *person* class ($\text{IoU}_\text{p}$) to particularly see how the attribute of pedestrian orientation affects the segmentation model in classifying the object class from which those at-tributes are derived. Detailed definitions on these metrics are provided in our previous paper [24]. In short, gAcc is obtained by the number of correctly-predicted pixels across all classes compared to the number of all pixels, and cAcc is similar to gAcc but calculated for each class, then averaged for all classes. Meanwhile, IoU of each class is the number of true positives divided by the sum of true positives, false positives, and false negatives.

The performances are measured in two ways: **object segmentation** that consists of 19 classes corresponding to all objects, and **attribute segmentation** that only focuses on four orientation classes. The output of the model trained with the Cityscapes dataset contains 19 labels and thus we only assess it with the object segmentation performance. Meanwhile, for all models trained with the CityWalks dataset, we can measure them with both object and attribute segmentation performances. Their segmentation outputs, however, contain 23 labels including all objects and attributes. Therefore, to calculate the object segmentation performance, we need to convert the output and ground-truth labels from 23-class mode to 19-class mode by substituting all attribute labels (19~22) with the *person* label (11). For attribute segmentation performance, we convert the labels into four-class mode by ignoring all object labels (0~18) before calculating each IoU. Especially for PSPNet.comp, the four attribute labels are not merged into the *person* label since object and attribute classes are already divided into two output maps.

### 5.3 Comparison of Validation Performances

We train each model using the training set containing 2,975 images and calculate the performance using 500 images in the validation set. Table 2 shows the performance comparison between object segmentation models. Performances in terms of attribute segmentation are compared in Table 3 where all models are trained with the CityWalks dataset. Note that we are not focusing between different base models here since it is already proven in the previous study [14] that PSPNet works much better than FCN8s in the pixel-wise labeling task. Therefore, we compare the performances locally for each base model.

As we can see in Table 2, FCN8s.prop only loses versus FCN8s in terms of $\text{IoU}_\text{p}$. Probably, this is due to the limitation of the FCN8s itself which is unstable in classifying a certain object. In more general measurements, i.e. gAcc, cAcc, and mIoU, the proposed FCN8s is better than the baseline. However, with PSPNet baseline, the proposed method outperforms both the baseline and the comparative methods in all metrics. Moreover, Table 4 shows the performance for each object class. In the table, the proposed method shows its ability to yield better IoU in more objects than the baselines. Therefore, we can infer that the best result for each base model is obtained from the proposed method.

In terms of attribute segmentation shown in Table 3, the proposed loss function applied to both base models generally works better than those using the standard function. For instance, for the FCN8s base model, FCN8s.prop is better than FCN8s.23cls in overall score despite the loss in a particular IoU, i.e. $\text{IoU}_{20}$. Meanwhile, for the PSPNet base model, PSPNet.prop is defeated by PSPNet.comp according to the average of all orientations' IoU. Nevertheless, if we take a detailed look at each IoU, the performances between those two methods can be said to be competitive between each other. For example, in $\text{IoU}_{19}$ and $\text{IoU}_{21}$, the proposed method loses, but on the other hand, in $\text{IoU}_{20}$ and $\text{IoU}_{22}$, it

**Table 2** Comparing performances on the object segmentation.

| Method | Loss function | Training data | gAcc | cAcc | mIoU | $IoU_p$ |
|---|---|---|---|---|---|---|
| FCN8s [12] | standard | Cityscapes | 88.44 | 57.12 | 46.33 | **48.01** |
| FCN8s.23cls | standard | CityWalks | 88.72 | 57.05 | 47.08 | 44.80 |
| **FCN8s.prop** | **proposed** | **CityWalks** | **88.84** | **57.73** | **47.64** | 47.45 |
| PSPNet [14] | standard | Cityscapes | 94.94 | 78.02 | 70.25 | 74.08 |
| PSPNet.23cls | standard | CityWalks | 94.86 | 79.48 | 70.85 | 71.69 |
| PSPNet.comp | standard | CityWalks | 92.54 | 71.45 | 59.02 | 63.72 |
| **PSPNet.prop** | **proposed** | **CityWalks** | **94.98** | **79.90** | **72.15** | **74.35** |

**Table 3** Comparing performances on the attribute segmentation; $IoU_{19}$, $IoU_{20}$, $IoU_{21}$, and $IoU_{22}$ are the IoU scores for orientations *back*, *right*, *front*, and *left*, respectively.

| Method | Average | $IoU_{19}$ | $IoU_{20}$ | $IoU_{21}$ | $IoU_{22}$ |
|---|---|---|---|---|---|
| FCN8s.23cls | 19.53 | 27.64 | **6.98** | 18.78 | 24.70 |
| **FCN8s.prop** | **21.34** | **28.52** | 5.58 | **25.51** | **25.75** |
| PSPNet.23cls | 38.52 | 54.63 | 23.26 | 49.56 | 26.63 |
| PSPNet.comp | **42.02** | **59.12** | 24.98 | **57.74** | 26.24 |
| **PSPNet.prop** | 41.17 | 56.83 | **26.85** | 50.54 | **30.48** |

**Table 4** Performances of semantic segmentation on the validation set for all object classes measured in each class's IoU and the Mean IoU.

| **Method** | **Mean IoU** | **road** | **sidewalk** | **building** | **wall** | **fence** | **pole** | **traffic light** | **traffic sign** | **vegetation** | **terrain** | **sky** | **person** | **rider** | **car** | **truck** | **bus** | **train** | **motorcycle** | **bicycle** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FCN8s [12] | 46.33 | 93.56 | 60.57 | 80.45 | 27.31 | 33.28 | 12.26 | 16.93 | 28.89 | 81.42 | 40.53 | 75.47 | **48.01** | **28.23** | **80.70** | 32.11 | **41.95** | 34.51 | 17.03 | 47.10 |
| FCN8s.23cls | 47.08 | 93.57 | **60.67** | 80.94 | **29.10** | 34.30 | 12.43 | 18.93 | 29.10 | 81.80 | 41.91 | 76.89 | 44.80 | 27.86 | 80.29 | 33.30 | 39.04 | **44.09** | 18.28 | 47.12 |
| **FCN8s.prop** | **47.64** | **93.59** | 60.34 | **81.20** | 28.80 | **34.42** | **12.63** | **19.07** | **29.26** | **81.81** | **42.44** | **77.60** | 47.45 | 27.93 | 80.57 | **36.69** | 39.90 | 42.08 | **22.02** | **47.36** |
| PSPNet [14] | 70.25 | **97.59** | **80.89** | 90.50 | 49.25 | 50.22 | 51.24 | 59.83 | **71.02** | 90.89 | 60.85 | **93.82** | 74.08 | 53.18 | 93.15 | 67.01 | 79.55 | 46.25 | **55.26** | **70.14** |
| PSPNet.23cls | 70.85 | 97.41 | 79.99 | 90.37 | 48.52 | 50.06 | **51.98** | **59.86** | 70.45 | **91.01** | **63.29** | 93.74 | 71.69 | 53.89 | 93.07 | 67.54 | 81.46 | 58.58 | 53.35 | 69.89 |
| PSPNet.comp | 59.02 | 95.93 | 71.27 | 87.10 | 31.03 | 37.57 | 38.65 | 41.80 | 53.35 | 88.19 | 53.94 | 91.25 | 63.72 | 38.30 | 88.66 | 48.59 | 63.03 | 44.09 | 25.22 | 59.73 |
| **PSPNet.prop** | **72.15** | 97.50 | 80.34 | **90.67** | **52.98** | **52.23** | 50.58 | 58.86 | 70.10 | 90.83 | 62.63 | 93.47 | **74.35** | **54.53** | **93.18** | **71.08** | **83.73** | **69.67** | 54.29 | 69.91 |

**Table 5** Comparing performances of FCN8s and PSPNet models using the proposed attribute-aware loss function with various $\hat{\beta}_o$ and $\hat{\beta}_a$ values.

| $\hat{\beta}_o, \hat{\beta}_a$ | FCN8s | | | PSPNet | | |
|---|---|---|---|---|---|---|
| | gAcc | mIoU | $IoU_p$ | gAcc | mIoU | $IoU_p$ |
| 0.0, 1.0 | 88.72 | 47.08 | 44.80 | 94.86 | 70.85 | 71.69 |
| 0.1, 0.9 | 88.63 | **47.16** | 46.57 | 94.91 | 70.60 | 73.69 |
| 0.2, 0.8 | **88.77** | 46.91 | 46.70 | 94.91 | 70.84 | 73.45 |
| 0.3, 0.7 | 88.72 | 46.79 | 47.02 | 94.92 | 70.61 | 73.87 |
| 0.4, 0.6 | **88.73** | 47.00 | **47.52** | 94.84 | 70.10 | 73.65 |
| 0.5, 0.5 | 88.64 | **47.08** | 47.11 | **94.98** | **72.15** | **74.35** |
| 0.6, 0.4 | 88.69 | 46.85 | 47.36 | 94.93 | 69.51 | **74.32** |
| 0.7, 0.3 | **88.84** | **47.64** | 47.45 | **94.95** | 71.00 | 74.26 |
| 0.8, 0.2 | 88.64 | 46.56 | **47.51** | 94.94 | **71.21** | **74.34** |
| 0.9, 0.1 | 88.40 | 46.71 | 47.41 | **94.97** | **71.21** | 73.89 |
| 1.0, 0.0 | 88.44 | 46.33 | **48.01** | 94.94 | 70.25 | 74.08 |

is the best among all methods.

### 5.4 Performances with Various $\hat{\beta}_o$ and $\hat{\beta}_a$

The weights $\hat{\beta}_o$ and $\hat{\beta}_a$ are the key parameters of the proposed attribute-aware loss function. Therefore, we trained the PSPNet and FCN8s models using the proposed loss function with various combinations of $\hat{\beta}_o$ and $\hat{\beta}_a$. Each weight value ranges from 0.0 to 1.0. In this case, assigning $\hat{\beta}_o$ and $\hat{\beta}_a$ with 0.0 and 1.0, respectively, means that the base model is trained with 23 classes but using the standard loss function, which corresponds to our previous work [24]. On the other way, assigning those weights with 1.0 and 0.0, respectively, is equivalent to the base model trained with 19 classes using the standard loss function, since the loss only considers the object loss in the learning process.

Table 5 shows the results of this experiment, comparing object segmentation performances based on gAcc, mIoU, and $IoU_p$. Numbers in bold indicate the best three in each column. For the FCN8s model, we can see that the pattern is not so clear on which pair of $\hat{\beta}_o$ and $\hat{\beta}_a$ results in the best performance among all settings. However, for the PSPNet model, we can see better and best performances commonly yield from $\hat{\beta}_o$ larger or equal to $\hat{\beta}_a$. Compared to the first and last rows, which do not handle both object and attribute classes simultaneously, the proposed method with various $\hat{\beta}_o$ and $\hat{\beta}_a$ has better results in general.

## 6. Discussion

According to Table 2, we can see that for both base models, the proposed method that uses the attribute-aware loss function outperforms other methods in the object segmentation task. This is also reinforced by what is shown in Table 4 where the proposed methods for both base models dominate the higher scores over all objects classes. The experimental results also show that the comparative method performs well particularly in the attribute segmentation task but loses much in the object segmentation task. This winning case is reasonable since the comparative method splits the attribute segmentation task away from the object segmentation. The classification task to learn by the comparative method's attribute branch has only four classes, which is much easier than the proposed method has. Nevertheless, looking at the results in Table 3, the proposed method is still able to compete with it. On the other hand, the performance of PSPNet.comp in classifying 19 object classes is worse than the other methods; It might require more epochs to fit the weights during training process due to additional trainable parameters introduced by the attribute branch. Thus, we can infer that by adding attributes to enrich the information of a particular type of object and treating it with a suitable loss function provided by our proposed method, the segmentation performance improves. This not only provides more information in the segmentation using pedestrian orientation, but also increases the ability to perform pixel-wise

classification, and furthermore, improves the global segmentation accuracy. The proposed method can also be applied to other base models not included in this experiment to see the improvement of segmentation performance. For more ITS applications, this technique can be extended to the training with other datasets enriched with more variability in environment and weather in order to better understand various conditions of traffic scenes.

Based on the experimental results, we consider that the gAcc of around 85~90% is sufficient to detect and locate some objects, but insufficient for the pixel-wise segmentation task. This means that with level of gAcc, we can use it to predict whether pedestrians exist or not, but still inadequate to help plan future actions. Therefore, we expect a higher accuracy of about 95% for a better environmental understanding. With a higher pixel-level accuracy, it will allow us to precisely locate the objects including their surrounding areas and will be helpful in predicting vehicle's future path to avoid collision. Thus, a more accurate segmentation result is needed for complex ITS applications. Additionally, we consider that accuracy improvement in semantic segmentation is necessary to better classify two or more similar objects but destined to have different labels. To see the improved results, we may refer to Fig. 10 showing that *person* and *rider* are actually the same object but classified into two different classes. In this case, a more accurate segmentation helps the autonomous driving system to distinguish those two classes and make a correct decision. Any small improvement is also important to make clearer boundaries between *person* and surrounding pixels that can help predict an alternative path to avoid obstacles or other further actions. Besides, for the attribute-aware semantic segmentation task, the pixel-level accuracy is important to avoid multiple attribute values labeled on the same object instance.

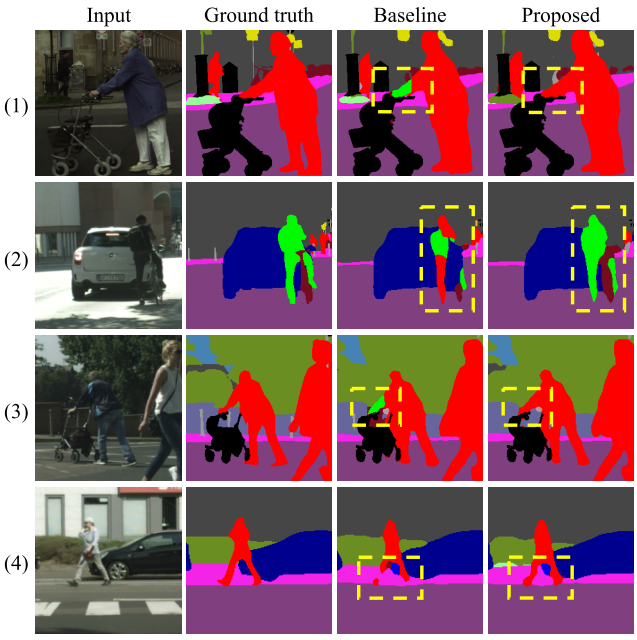


**Fig. 10** Qualitative results in the object segmentation task, comparing the baseline (PSPNet [14]) and the proposed methods (PSPNet.prop); the difference is seen inside the yellow-dashed box.

### 6.1 Qualitative Result

Figure 10 shows some results that indicate how the proposed method enhances the segmentation outputs. Some parts of *person* pixels are often misclassified as rider since they are quite similar. In input (1) and (2), the proposed method performs better while the baseline method labels incorrectly in some parts of the person pushing a cart. Sometimes, the baseline method is also confused to label a *rider* who stands and looks like a *person* as seen in input (3), while the proposed method can handle it correctly. In another case, the segmentation might fail to form a complete person. For example, at the leg part of input (4), the proposed method can form complete legs better than the baseline method. This figure demonstrates that the proposed method is capable of improving the segmentation accuracy by considering the pedestrian's attribute.

In terms of objects combined with attributes segmentation, let us inspect Fig. 11 to see sample results of the proposed method. A pedestrian group with the same orientation is shown in column (1), while a pedestrian with a different orientation from a group is shown in column (2). Columns (3) and (4) represent cases of crossing pedestrians. The proposed method shows its capability to correctly distinguish pedestrians with body orientation attributes *left* or *right*. Column (5) also shows the superiority of the proposed method in segmenting a crowd of people with various orientation attributes. Overall, these figures verify that the results yielded by the proposed method are in good quality thanks to the multi-tasking segmentation by combining object and attribute recognitions.

### 6.2 Computational Cost

We point out the load of computation to execute the training process with several settings according to GPU usage and the number of epochs completed per day. We used the NVIDIA GeForce GTX 1080 Ti GPU to train each model. Table 6 shows the summary results of computational costs recorded during each training. The first three rows indicate the costs to run the baseline, comparative, and the proposed methods, respectively. We can see that there is no significant difference among the three methods as long as the image and window cropping sizes are the same. For the comparative method, although it splits the layers, the cost of computation is not affected much since the additional branch is placed in the final layer. We can infer that the proposed method using the attribute-aware loss function is capable of performing an MTL-based task but takes no risk on the computational cost. This is in accordance with the explanation in Sect. 3.

The computational cost in the training process is, however, mostly influenced by the image size and the window cropping size. In the fourth row of Table 6, when the image size is doubled, the training speed is much slower, but the

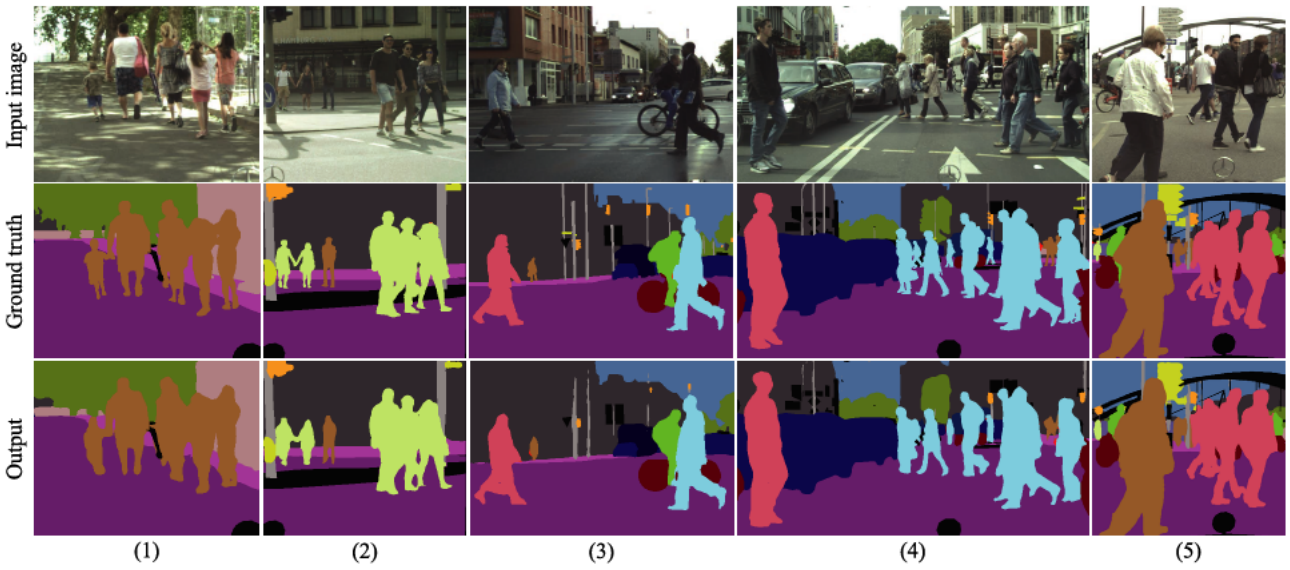


**Fig. 11** Some results of the proposed method (PSPNet.prop) in the attribute-aware semantic segmentation task, from (1) easy (uniform) to (5) complex (heterogeneous) cases.

**Table 6** Computational costs of PSPNet training with several settings.

| Method | Image size | Crop size | GPU usage | Train speed |
|---|---|---|---|---|
| PSPNet [14] | 1,024 × 512 | 512 × 512 | ~10 GB | ~33 epochs/day |
| PSPNet.comp | 1,024 × 512 | 512 × 512 | ~10 GB | ~33 epochs/day |
| PSPNet.prop | 1,024 × 512 | 512 × 512 | ~10 GB | ~33 epochs/day |
| PSPNet.prop | 2,048 × 1,024 | 512 × 512 | ~11 GB | ~7.2 epochs/day |
| PSPNet.prop | 2,048 × 1,024 | 713 × 713 | ~20 GB | ~8.4 epochs/day |
| PSPNet.prop | 2,048 × 1,024 | 916 × 916 | ~30 GB | ~6.7 epochs/day |

GPU usage only increases slightly as the cropping size is still the same. Meanwhile, the cropping size has more influence on the GPU size. As the cropping size increases, the space required on the GPU increases sharply. It also affects on the training speed, but not too significant when the image size remains the same.

For real ITS applications, here we also consider the processing speed on the testing data. Since the proposed method practically just modifies the training process, the computational cost of testing will not be influenced by the attribute-aware loss function. For the proposed methods using FCN8s and PSPNet base models, they gave around 5.00 and 2.57 frames per second (fps), respectively. These costs were calculated for images sized 1,024 × 512 pixels in the NVIDIA's GPU as previously mentioned. That FCN8s runs faster than PSPNet is presumably noticed since the PSPNet model by [14] is much complex as compared to the FCN8s by [12]. The processing speed here is apparently very slow if we compare it to a video that commonly moves with 20~30 fps, and hence insufficient for common vehicle based vision in real-time. Since a real self-driving car requires high accuracy, as well as fast processing on a limited hardware, it is recommended to implement the semantic segmentation models on FPGA, which is extensively used in embedded applications. The FPGA implementation that is faster and more efficient but still maintains its performance [38] can be a solution to the problem of the real-time processing speed.

## 7. Conclusion

We introduced a new concept of attribute-aware semantic segmentation to enrich the scene understanding as well as an approach which allows a deep neural network model trained in an end-to-end process. Our main contribution is the proposal of an attribute-aware loss function capable of handling segmentation loss for both object and attribute classes in one calculation flow which can be applied to an arbitrary base model. We also enriched the most popular pixel-wise labeled dataset Cityscapes by adding body orientations of pedestrian as an attribute to the *person* class to extend the ground-truth labels, which is named the CityWalks dataset and will be publicly available on the Web. Experiments with various settings were conducted and the results showed that the proposed method successfully outperforms the baseline methods covered in this study. It shows that the proposed method is able to work well in recognizing pedestrian attributes and improving the performance of semantic segmentation for object classes as challenged in various competitions. A further extension in terms of attribute types will be our future plan as well as applying the method to other moving objects.

## Acknowledgments

The first author would like to thank Indonesia's BUDI-LN scholarship for the generous supports to his doctoral study. Parts of this research were also supported by MEXT, KAKENHI Grant Number JP 17H00745.

### References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation,"

IEEE Trans. Pattern Anal. Mach. Intell., vol.39, no.12, pp.2481–2495, Dec. 2017.
[2] A. Van Etten, D. Lindenbaum, and T.M. Bacastow, "SpaceNet: A remote sensing dataset and challenge series," Computing Research Repository arXiv preprint, arXiv:1807.01232, July 2018.
[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," Proc. 18th Int. Conf. on Medical Image Computing and Computer-Assisted Intervention, pp.234–241, Oct. 2015.
[4] H.R. Roth, C. Shen, H. Oda, M. Oda, Y. Hayashi, K. Misawa, and K. Mori, "Deep learning and its application to medical image segmentation," Med. Imag. Tech., vol.36, no.2, pp.63–71, March 2018.
[5] K. Khan, N. Ahmad, K. Ullah, and I. Din, "Multiclass semantic segmentation of faces using CRFs," Turk. J. Electr. Eng. Co., vol.25, no.4, pp.3164–3174, July 2017.
[6] B.J. Meyer and T. Drummond, "Improved semantic segmentation for robotic applications with hierarchical conditional random fields," Proc. 2017 IEEE Int. Conf. on Robotics and Automation, pp.5258–5265, May 2017.
[7] M. Elhoseiny, S. Huang, and A. Elgammal, "Weather classification with deep convolutional neural networks," Proc. 2015 IEEE Int. Conf. on Image Processing, pp.3349–3353, Sept. 2015.
[8] T. Wu and A. Ranganathan, "Vehicle localization using road mark-ings," Proc. 2013 IEEE Intelligent Vehicles Symposium, pp.1185–1190, June 2013.
[9] D. Wong, D. Deguchi, I. Ide, and H. Murase, "Vision-based vehicle localization using a visual street map with embedded SURF scale," Proc. European Conf. on Computer Vision 2014 Workshops, pp.167–179, Sept. 2014.
[10] X. Liu and Z. Deng, "Segmentation of drivable road using deep fully convolutional residual network with pyramid pooling," Cogn. Comput., vol.10, no.2, pp.272–281, April 2018.
[11] F. Shinmura, Y. Kawanishi, D. Deguchi, T. Hirayama, I. Ide, H. Murase, and H. Fujiyoshi, "Estimation of driver's insight for safe passing based on pedestrian attributes," Proc. 21st IEEE Int. Conf. on Intelligent Transportation Systems, pp.1041–1046, Nov. 2018.
[12] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," Proc. 2015 IEEE Conf. on Computer Vision and Pattern Recognition, pp.3431–3440, June 2015.
[13] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian SegNet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," Computing Research Repository arXiv preprint, arXiv:1511.02680, Nov. 2015.
[14] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," Proc. 2017 IEEE Conf. on Computer Vision and Pattern Recognition, pp.2881–2890, June 2017.
[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," Proc. 2017 IEEE Int. Conf. on Computer Vision, pp.2961–2969, Sept. 2017.
[16] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," Proc. European Conf. on Computer Vision 2018, pp.405–420, Sept. 2018.
[17] L.C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A.L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," IEEE Trans. Pattern Anal. Mach. Intell., vol.40, no.4, pp.834–848, April 2018.
[18] L.C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," Computing Research Repository arXiv preprint, arXiv:1706.05587, June 2017.
[19] L.C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," Proc. European Conf. on Computer Vision 2018, pp.801–818, Sept. 2018.
[20] G.J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," Pattern Recogn. Lett., vol.30, no.2, pp.88–97, Jan. 2009.
[21] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," Int. J. Robo. Res., vol.32, no.11, pp.1231–1237, Sept. 2013.
[22] 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," Proc. 2016 IEEE Conf. on Computer Vision and Pattern Recognition, pp.3213–3223, June 2016.
[23] M.D. Sulistiyo, Y. Kawanishi, D. Deguchi, I. Ide, and H. Murase, "A preliminary study of attribute-aware semantic segmentation for pedestrian understanding," Proc. 2017 Electric/Electronic/Information Engineering Related Society Tokai Sectors Joint Convention, no.A2-7, Sept. 2017.
[24] M.D. Sulistiyo, Y. Kawanishi, D. Deguchi, T. Hirayama, I. Ide, J.Y. Zheng, and H. Murase, "Attribute-aware semantic segmentation of road scenes for understanding pedestrian orientations," Proc. 21st IEEE Int. Conf. on Intelligent Transportation Systems, pp.2698–2703, Nov. 2018.
[25] X. Liu, Z. Deng, and Y. Yang, "Recent progress in semantic image segmentation," Computing Research Repository arXiv preprint, arXiv:1809.10198, Sept. 2018.
[26] M. Cordts, M. Omran, S. Ramos, T. Scharwächter, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes dataset," CVPR 2015 Workshop on the Future of Datasets in Vision, June 2015.
[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Computing Research Repository arXiv preprint, arXiv:1409.1556, Sept. 2014.
[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. 2016 IEEE Conf. on Computer Vision and Pattern Recognition, pp.770–778, June 2016.
[29] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," Int. J. Comput. Vis., vol.88, no.2, pp.303–338, June 2010.
[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," Proc. 28th Int. Conf. on Neural Information Processing Systems, pp.91–99, Dec. 2015.
[31] Y. Zhang and Q. Yang, "An overview of multi-task learning," Natl. Sci. Rev., vol.5, no.1, pp.30–43, Sept. 2017.
[32] S. Ruder, "An overview of multi-task learning in deep neural networks," Computing Research Repository arXiv preprint, arXiv:1706.05098, June 2017.
[33] Y. Lu, D. Allegra, M. Anthimopoulos, F. Stanco, G.M. Farinella, and S. Mougiakakou, "A multi-task learning approach for meal assessment," Proc. 2018 Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management, pp.46–52, July 2018.
[34] D.C. Luvizon, D. Picard, and H. Tabia, "2D/3D pose estimation and action recognition using multitask deep learning," Proc. 2018 IEEE Conf. on Computer Vision and Pattern Recognition, pp.5137–5146, June 2018.
[35] Z. Deng, "PyTorch for semantic segmentation," https://github.com/zijundeng/pytorch-semantic-segmentation/, 2017.
[36] M. Enzweiler and D.M. Gavrila, "Integrated pedestrian classification and orientation estimation," Proc. 2010 IEEE Conf. on Computer Vision and Pattern Recognition, pp.982–989, June 2010.
[37] S. Zhang, R. Benenson, and B. Schiele, "CityPersons: A diverse dataset for pedestrian detection," Proc. 2017 IEEE Conf. on Computer Vision and Pattern Recognition, pp.3213–3221, July 2017.
[38] A. Podili, C. Zhang, and V. Prasanna, "Fast and efficient implementation of convolutional neural networks on FPGA," Proc. 28th IEEE Int. Conf. on Application-specific Systems, Architectures and Processors, pp.11–18, July 2017.