IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 1, 2020, accepted February 11, 2020, date of publication February 21, 2020, date of current version March 2, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2975640

Real-Time Driving Scene Semantic Segmentation

WENFU WANG, YONGJIAN FU, ZHIJIE PAN, XI LI, AND YUETING ZHUANG
College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China
Corresponding author: Wenfu Wang (wangwf73@zju.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61751209.

ABSTRACT Real-time understanding of surrounding environment is an essential yet challenging task for autonomous driving system. The system must not only deliver accurate result but also low latency performance. In this paper, we focus on the task of fast-and-accurate semantic segmentation. An efficient and powerful deep neural network termed as Driving Segmentation Network (DSNet) and a novel loss function Object Weighted Focal Loss are proposed. In designing DSNet, our goal is to achieve the best capacity with constrained model complexity. We design efficient and powerful unit inspired by ShuffleNet V2 and also integrate many successful techniques to achieve excellent balance between accuracy and speed. DSNet has 0.9 million of parameters, achieves 71.8% mean Intersection-over-Union (IoU) on Cityscapes validation set, 69.3% on test set, and runs 100+ frames per second (FPS) at resolution 640 × 360 on NVIDIA 1080Ti. In order to improve performance on minor and hard objects which are crucial in driving scene, Object Weighted Focal Loss (OWFL) is proposed to deal with the serious class imbalance issue in pixel-wise segmentation task. It could effectively improve the overall mean IoU of minor and hard objects by increasing loss contribution from them. Experiments show that DSNet performs 2.7% points higher on minor and hard objects compared with fast-and-accurate model ERFNet under similar accuracy. These traits imply that DSNet has great potential for practical autonomous driving application.

INDEX TERMS Autonomous driving perception, efficient neural networks, semantic segmentation.

I. INTRODUCTION

An autonomous vehicle must immediately, accurately and comprehensively understand the complex surrounding environment, which poses great challenge to driving perception system. Thanks to the remarkable progress of deep learning, computer vision is playing an increasingly important role in driving perception task [1], [2]. Image semantic segmentation could obtain exhaustive information such as object categories, shape, spatial location at pixel level, thus is especially beneficial for comprehensive driving scene understanding.

The task of image semantic segmentation is to densely label each pixel in an image to its object category, and result in an image with non-overlapping meaningful regions. Many computer vision and machine learning methods have been proposed [3]. In recent years, Convolutional Neural Network (CNN) based methods achieve remarkable progress on image semantic segmentation [4]–[6], significantly improving accuracy even efficiency [7], and have become the de facto solution. However, current state-of-the-art semantic segmentation methods are not practical for autonomous driving application, since they can not fulfill the low latency requirement in autonomous driving system. These methods are pursing higher scores with increasingly larger number of parameters and complex modules [6], [8], [9]. However, finer segmentation results come at the expense of very long inference time, some methods even take more than one second to process an image. This seriously limits the application of semantic segmentation methods. Therefore, the quest for fast-and-accurate methods is becoming a very active research direction.

Intuitively, some lightweight methods are designed with much less parameters for real-time performance, for instance, ENet [7] and ESPNet [10]. They replace cumbersome 3 × 3 convolutions with point-wise (1 × 1) convolutions and factorized convolutions, and also adopt other techniques to drastically reduce the number of parameters. As a result, ENet [7] and ESPNet [10] which both have about 0.4M (million) parameters are 180 times lighter than PSPNet [8], and 79 times than SegNet [5]. ENet [7] is able to achieve 58.3% mean IoU and ESPNet [10] 60.3%. The results are significantly lower than PSPNet [8] of 80.2%, but surpass SegNet [5] of 57.0%. See Table 1 for more detailed comparison. Lightweight models demonstrate the potential to outperform some cumbersome methods at the same time running real-time, however, in our opinion, excessively reducing

The associate editor coordinating the review of this manuscript and approving it for publication was Vivek Kumar Sehgal.

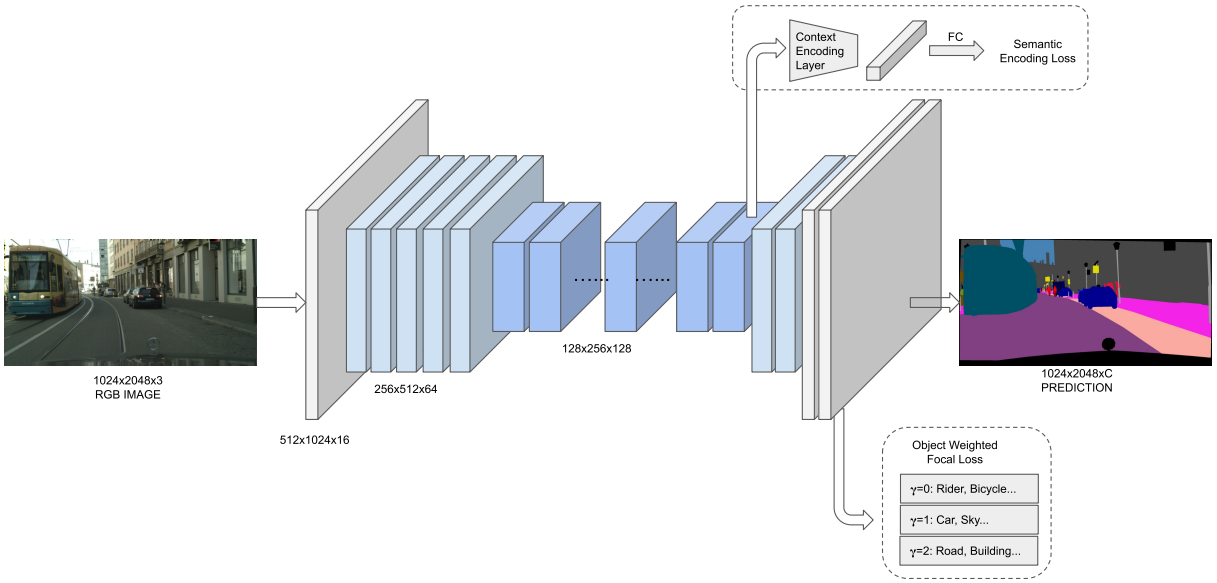

FIGURE 1. Diagram of the proposed method (DSNet) for an example input and its corresponding output (C = 19), and γ is the object order-of-magnitude weight which is introduced in Section III. All spatial resolution values are with example input of 1024 × 2048, the network can perform on arbitrary image sizes.

parameters like ENet will sacrifice much capability of the model. In our trial experiments, with the number of parameters under 0.4M we can not achieve mean IoU higher than 62% on Cityscapes dataset. Such few parameters could lead to unsatisfying result of critical objects in the driving scene, for example bicycle in ENet [7] scores 34.1% which is too low to provide accurate information for safe autonomous driving. Some efficient methods achieve satisfying accuracy at the same time delivering excellent runtime performance. For instance, ERFNet [11] and EDANet [12] propose efficient and powerful units with reasonable parameters (ERFNet and EDANet are 5.8 and 1.8 times larger than ENet), and achieve impressive mean IoU close to 70%. ICNet [13] and BiSeNet [14] optimize upon existing advanced model with novel modules for the trade-off between accuracy and speed. Refer to Section II for more discussion and Table 1 for detailed comparison.

Although some methods could achieve impressive balance between accuracy and speed, they lack specific tool to handle hard and minor objects which are crucial for autonomous driving. Take ERFNet [11] for example, its mean IoU is as high as 69.7% over all objects, however the IoUs of critical objects such as truck, motorcycle and train are around 50% which is far from its mean IoU. It would provide confusing information of the scene, and such model is not ready for perception application especially on urban street scene. One of the main reasons for low performance on these objects is the serious class imbalance issue naturally in pixel-wise image segmentation task. For example, in a street image, large objects such as road, building, sky would occupy most of the image, resulting in very imbalance distribution of objects. Fig.2 shows the percentage of objects count in pixel on Cityscapes train dataset. As we can see, top six objects account over 78% of all the pixels, while minor objects such as train, bus, truck, motorcycle, and rider in total account less than 1.2%. Training on such imbalanced dataset could lead to a very biased model.

In this paper, we aim to propose a fast-and-accurate model for practical use. It should not only achieve excellent balance between accuracy and inference speed, but also focus on improving the performance of hard and minor objects. In designing lightweight model, different from extensively reducing the number of parameters like ENet [7], we determine to increase the number of parameters slightly (it is still a tiny model) and adopt efficient yet powerful modules and techniques to ensure decent quality and speed at the same time. We design our units based on basic unit of ShuffleNet V2 [15], and adapt it to the task of semantic segmentation. The ShuffleNet V2 unit first splits the channel of input, and employs a residual architecture where one branch consists of point-wise convolution and depth-wise convolution, the two branches are finally concatentated and shuffled. ShuffleNet V2 aims to reduce computation complexity at the same ensuring its powerful expressiveness. To better handle minor and hard objects in segmentation task, we propose Object Weighted Focal Loss (OWFL). It first adopts normalized object frequency weight to balance the biased loss value, then object order-of-magnitude weight further increases the loss contribution gap between minor-and-hard and major-and-easy objects, guiding the network to concentrate on minor and hard objects. The two weights are derived from object distribution of the dataset. We also integrate Semantic Encoding Loss from [16]. The whole method is depicted in Fig. 1.

TABLE 1. Evaluation results on cityscapes test set. "Sub": the downsampling factor of the input images. "ImN": ImageNet dataset. "coarse": the coarse annotation set of cityscapes dataset.

Method	BaseModel	Extra data	meanIoU(%)	Sub	Time(s)	Speed(FPS)	Params
Dilation10 [17]	VGG16	ImN	67.1	no	4	0.25	140.8M
FCN-8s [4]	VGG16	ImN	65.3	no	0.5	2	134.5M
Deeplab [6]	VGG16	ImN	63.1	no	4	0.25	n/a
PSPNet [8]	ResNet	ImN	80.2	no	no	no	65.7M
SegNet [5]	VGG16	ImN	57.0	4	0.060	16.7	29.5M
ENet [7]	no	no	58.3	2	0.032	30.8	0.36M
ESPNet [10]	no	ImN	60.3	2	0.009	112.9	0.36M
ICNet [13]	PSPNet	coarse+ImN	69.5	no	0.033	30.3	n/a
BiSeNet [14]	Xception39	ImN	68.4	no	0.009	105.8	5.8M
CGNet [18]	no	no	64.8	no	0.056	18.0	0.5M
EDANet [12]	no	no	67.3	no	0.009	108.7	0.68M
ERFNet [11]	no	ImN	69.7	2	0.024	41.7	2.1M
DSNet	no	coarse	69.3	2	0.027	36.5	0.91M

In summary, our main contributions are as following.

- We design efficient and powerful unit and asymmetric encoder-decoder architecture inspired by ShuffleNet V2 [15] and ENet [7], and propose a lightweight model Driving Segmentation Network (DSNet).
- The proposed Object Weighted Focal Loss could effectively improve the overall accuracy of minor and hard objects by a large margin.
- DSNet has 0.9M parameters, runs 100+ FPS at resolution 640×360 on NVIDIA 1080Ti with 69.3% accuracy on Cityscapes test dataset, which achieves excellent balance between accuracy and speed.

The rest of this paper is organized as follows. Section II reviews related works, Section III discusses computation complexity, and introduces the units, architecture and loss function of DSNet. Section IV reports our experimental results on Cityscapes dataset. Section V draws the conclusion.

II. RELATED WORK

In this section, we briefly review literature on classical and lightweight semantic segmentation models, and class imbalance issue. The comparison with related methods in detail is summarized in Table 1, including mean IoU, inference time, the number of parameters if provided, and base model.

A. SEMANTIC SEGMENTATION MODELS

The first CNN model successfully applied on image semantic segmentation is Fully Convolutional Network (FCN) [4]. It achieves great improvement in accuracy than traditional methods on PASCAL VOC [19] dataset, and demonstrates how to use a CNN model to solve image semantic segmentation problems. This triggers a research boom of CNN-based methods on image semantic segmentation, to name a few representative works, SegNet [5], Dilation10 [17], DeepLab V3+ [20], PSPNet [8] and ICNet [13]. RNN could also be applied in semantic segmentation task and is able to successfully model global context [21]–[23]. For example, Byeon *et al.* [22] proposes a simple 2D LSTM based architecture in which the input image is divided into non-overlapping windows which are fed into four separate LSTM memory blocks. Mask R-CNN is proposed for instance segmentation task. It builds upon Faster R-CNN and adds an additional branch for predicting segmentation masks on each pixel of Region of Interest (RoI).

SegNet [5] and U-Net [24] adopt encoder-decoder architecture. Dilation10 [17] first employs dilated convolution (also called atrous convolution) in cascade in semantic segmentation CNN models. Compared with pooling operation, dilated convolution could have various receptive field by employing different dilation rates, while pooling operation does not have any parameters to save. In addition, compared with standard convolution, dilated convolution could gain larger receptive field without increasing parameters and computation but in the price of local spatial information. PSPNet [8] proposes Pyramid Pooling Module in semantic segmentation task, which uses pyramid pooling module to generate global scene prior upon the final feature map of the network at four different scales. In DeepLab series [6], [25], the authors highlight the use of atrous convolution and propose Atrous Spatial Pyramid Pooling (ASPP) module to aggregate object and context information at different scales. RefineNet [9] proposes Refine module which takes one feature map and its lower scale feature map in the encoder, and fuses them as feature map in the decoder.

B. LIGHTWEIGHT SEMANTIC SEGMENTATION MODELS

Efficient methods are to seek the balance between accuracy and real-time performance, which can be classified into two main categories: methods which are designed or utilized a light model with fewer parameters, and methods which optimize other advanced networks with novel techniques or modules.

In the first category, ENet and ESPNet are very light models which both have about 0.4M parameters. ENet [7] designs its efficient units using point-wise convolution or factorized convolution, and a simple decoder also helps reduce parameters, and ESPNet [10] proposes efficient spatial pyramid

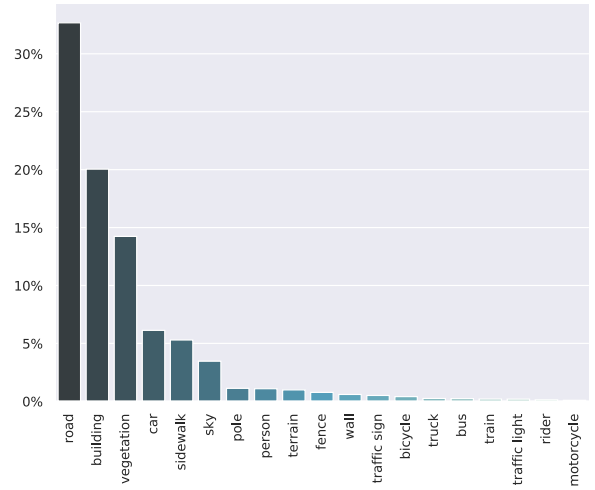

FIGURE 2. Objects pixel-wise percentage of cityscapes train dataset. Horizontal axis represents objects' names, and vertical axis represents their corresponding percentage in the train dataset. The distribution shows a clear long-tail effect, which top 6 objects dominate the train dataset.

module where 3×3 convolution is replaced by point-wise convolution and spatial pyramid of dilated convolution. These techniques massively reduce parameters. However, drastically reducing parameters could sacrifice much capability of the model for dense pixel-wise semantic segmentation task. Reference [11] utilizes factorized convolution to its best, and proposes ERFNet which has 5.8 times more parameters than ENet and achieves excellent accuracy and speed balance. EDANet [12] employs an asymmetric convolution structure incorporating the dilated convolution and the DenseNet-like architecture to attain high efficiency. CGNet [18] proposes efficient Context Guided block, and scores 64.8% mean IoU on Cityscapes with only 0.5M parameters. In the second category, for example, ICNet [13], BiSeNet [14], and ShelfNet [26] propose novel modules or techniques to optimize advanced models. ICNet [13] proposes a PSPNet-based architecture. The authors input three scales images, small scale image goes through deeper networks, large scale shallower, and fuses three scales of features through cascade feature unit. BiSeNet [14] builds upon Xception 39 [27] and ResNet [28], and proposes spatial path and context path, and FFM (Feature Fusion Module) and ARM (Attention Refine Module) modules, where ARM in context path employs global average pooling to capture global context and generates an attention vector to guide the feature learning and FFM fuses features from spatial path and context path. ShelfNet [26] is based on ResNet [28], and has multiple encoder-decoder branch pairs at each spatial level.

C. CLASS IMBALANCE

Class imbalance issue refers to the problem where the disparity in the proportion of different classes in the whole dataset is overwhelming [29], [30]. As mentioned above, there is severe class imbalance issue inherited in segmentation task [31]. This imbalance is especially difficult for lightweight models, since with much constrained capacity compared with large models, the minor classes would be more easily drowned during training. Class imbalance problem is also prevalent in other computer vision tasks, for example anchor-based object detection task [32], and depth estimation task [33].

Approaches dealing with class imbalance problem could be summarized into two main categories: data level methods and classifier level methods [29]. Data level methods aim to increase the volume of minor samples by data augmentation or over-sampling, or decrease major samples from under-sampling. For example, [34] proposes class-aware sampling which controls the selection from each class and ensure uniform distribution of each mini-batch.

Classifier level methods, in the context of deep learning, mainly refer to cost-sensitive re-weighting and novel loss function designs. Cost-sensitive re-weighting assigns relatively higher cost to minor classes. ENet [7] proposes a class re-weighting scheme which affects the loss function by assigning weights according to the inverse of proportion of each class. ERFNet [11] also adopts this re-weighting scheme. Reference [35] proposes class re-balancing scheme based on effective number of samples. As for designing loss functions, Gradient Harmonizing Mechanism [36] further suggests a novel loss function to balance the gradient norm of each class. Focal loss [37] is designed to dynamically adjust higher cost to hard classes and lower to easy classes during training. In [38], the authors propose online bootstrapping of hard training pixels, which drops pixels with small loss value. Reference [39] proposes Online Hard Example Mining (OHEM) to select hard regions-of-interest (RoIs) for object detection, and OCNet [40] adopts OHEM in semantic segmentation.

III. DESIGNING DRIVING SEGMENTATION NETWORK

In designing DSNet, we keep in mind that both accuracy and speed are important, and aim to achieve the best capacity with constrained and reasonable model complexity. Many previous successful techniques in [41] and others are integrated. We first discuss important runtime performance metrics, and then explain in detail about DSNet units, architecture, and design choices. At last, we propose the loss function design.

A. COMPUTATION COMPLEXITY

Inference speed (FPS) is the direct metric to evaluate computation complexity of CNN based approaches. Inference speed could vary in different software and hardware settings, hence two indirect metrics are usually evaluated in lightweight CNN models: number of parameters and number of float-point operations (FLOPs). Another vital metric, memory access cost (MAC), refers to the number of memory access operations on physical device. If we assume that the cache in the computing device is large enough to store the feature maps and parameters, MAC for 1×1 convolution could be theoretically calculated by equation $\text{MAC} = hw(c_1 + c_2) + c_1c_2$, where c_1 and c_2 are the input and output channel number, h and w are the spatial size of the feature map.

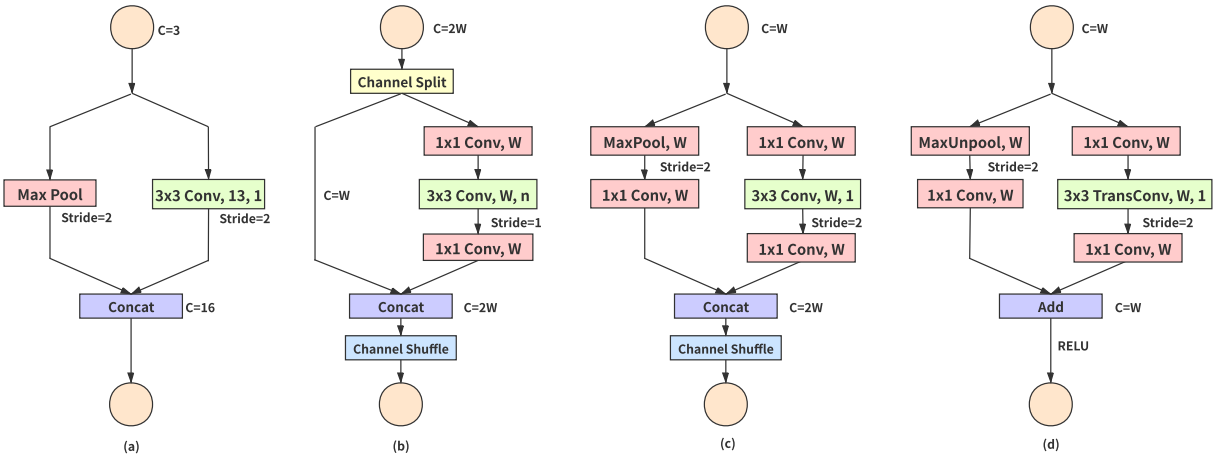

FIGURE 3. DSNet Units. (a) Init unit. (b) Basic unit. (c) Down unit. (d) Up unit. *C* represents the number of channels. In 3 × 3 Conv and TransConv, "$k_1 \times k2$, w, d" indicate the kernel size ($k_1 \times k_2$), the output channel number (w), and the dilation rate (d). All convolutions are followed by Batch Norm and ReLU, and all 1 × 1 Conv of units have stride = 1 and dilation = 1.

TABLE 2. Different size of DSNets performance. Params is short for the number of parameters, and K indicates thousand of parameters. FPS is evaluated following inference speed experiment in Section IV at resolution 640 × 360.

	Params(K)	mIoU	FPS
DSNet0.5	4.5	56.1%	121.3
DSNet1.0	9.1	62.8%	100.5
DSNet1.5	13.6	64.0%	77.9
DSNet2.0	18.1	64.2%	65.3

The sensible paradigm of designing efficient CNN models is not to achieve light by drastically reducing parameters, but to design efficient and powerful modules with reasonable amount of parameters. We need to significantly reduce the number of parameters compared to cumbersome models. However more importantly, we should also avoid over-reducing the number of parameters such as ENet. We design basic units mainly by modifying ShuffleNet V2 module enjoying its high efficiency in reducing MAC and FLOPs [15] at the same time remaining powerful expressiveness.

To evaluate the trade-off between computation complexity and accuracy, we conduct experiments of training DSNet with increasing parameters, termed as DSNet0.5, DSNet1.0, DSNet1.5, DSNet2.0, where the number indicates the ratio of the model's parameters to the proposed parameters, and we achieve this by adjusting the number of units. DSNet0.5 reduces to 6 Basic units with dilate rate scheme of 2, 5, 9, 5, 9, 17, DSNet1.5 adds another 10 Basic units compared with DSNet1.0, and DSNet2.0 adds another 10 Basic units compared with DSNet1.5. See Section IV for training and evaluation details.

From Table 2, we can see that DSNet0.5 scores 6.7% points lower than DSNet1.0, we contemplate that limited parameters and shallow depth of the network are the main reason. While compared with DSNet1.0, DSNet1.5 and DSNet2.0 have increased 1.2% and 1.3% points which indicates that the increased depth and number of parameters could not promise proportional improvement of accuracy. As accuracy does not improve proportionally with respect to the number of parameters but FPS decreases linearly, we therefore choose DSNet1.0 as it has enough parameters and network depth to achieve good accuracy at the same time running fast at inference.

B. DSNET UNITS

DSNet Unit is shown in Fig. 3. We adopt initial unit from ENet, which use max pooling and convolution with stride 2 to down-sample the input. The Basic unit develops from ShuffleNet V2 unit where input channel is first split into two. Depth-wise separable convolution in ShuffleNet V2 is replaced with dilated convolution to enlarge receptive field, which is vital for semantic segmentation task. The feature channel of convolution layer in the units has equal channel width following guidelines in ShuffleNet V2 [15] to reduce MAC. In Down unit, input is max pooling following 1 × 1 convolution in left branch of the unit, and in up-sample unit, input is un-pool from corresponding down-sample unit. In the final part, down-sample unit perform concatenation and channel shuffle like basic unit, while up-sample unit adds left and right branch features. The add operation introduces little additional computation, as we only have two such units in the whole architecture. We also would like to highlight that the basic unit achieves feature reuse like DenseNet [42], since half of the features directly go through the block and join the next block.

C. DSNET ARCHITECTURE

The architecture of DSNet is shown in Table 3. We determine to adopt asymmetric encoder-decoder architecture like ENet [7]. The asymmetric architecture has three main stages as encoder, two light stages as decoder. The structure of

TABLE 3. The architecture of DSNet. Output shape is given for an example input of 800 × 800. C is the number of classes.

Unit	Type	Output Shape
Init Unit		$16 \times 400 \times 400$
Down Unit 1.0		$64 \times 200 \times 200$
$4\times$ Basic Unit 1.x	$Dilation = 1$	$64 \times 200 \times 200$
Down Unit 2.0		$128 \times 100 \times 100$
Basic Unit 2.1	$Dilation = 1$	$128 \times 100 \times 100$
Basic Unit 2.2	$Dilation = 2$	$128 \times 100 \times 100$
Basic Unit 2.3	$Dilation = 5$	$128 \times 100 \times 100$
Basic Unit 2.4	$Dilation = 9$	$128 \times 100 \times 100$
Basic Unit 2.5	$Dilation = 1$	$128 \times 100 \times 100$
Basic Unit 2.6	$Dilation = 2$	$128 \times 100 \times 100$
Basic Unit 2.7	$Dilation = 5$	$128 \times 100 \times 100$
Basic Unit 2.8	$Dilation = 9$	$128 \times 100 \times 100$
Basic Unit 3.x repeats Basic Unit $2.1 - 2.8$ with dilated rate $2, 5, 9, 17, 2, 5, 9, 17$		
Up Unit 4.0		$64 \times 200 \times 200$
Basic Unit 4.1	$Dilation = 1$	$64 \times 200 \times 200$
Up Unit 5.0		$16 \times 400 \times 400$
Basic Unit 5.1	$Dilation = 1$	$16 \times 400 \times 400$
Fullconv		$C \times 800 \times 800$

ENet's architecture is a thoroughly considered choice, and it is also adopted by ERFNet [11].

For dilation rate scheme, in DRN [43], dilation rate scheme of $2, 4$ is applied at last two blocks of ResNet, in Deeplab V3 [25], dilation rate scheme of $2, 4, 8, 16$ is applied, similar dilation rate scheme is also adopted in ENet [7], in Dilation-bigger of Hybrid Dilated Convolution (HDC) [44], consecutive dilation rate scheme of $1, 2, 5, 9$ and $5, 9, 17$ is applied at *res*4*b* and *res*5*b* of ResNet respectively. In determining dilation rate scheme, we follow the scheme of HDC, which performs better in overcoming the "gridding" issue in our experiments (see ablation study for proposed architecture and visual comparison).

1) MULTI-SCALE FEATURE FUSION

Multi-scale feature fusion refers to the technique of merging feature maps from different scales in a network. For example, FCN-8s [4] fuses $1/8$ feature maps from $1/16$ and $1/32$ scales to obtain a fine-grain output. Pyramid pooling module in PSPNet [8] fuses features under four different pyramid pooling scales.

Multi-scale feature fusion has been proved a beneficial technique to achieve better accuracy. However, our concern is that multi-scale feature fusion usually adds more paths, which violates the degree of parallelism and brings additional computation. In designing DSNet architecture, we do not to utilize multi-scale feature fusion. See ablation study experiment which adds pyramid pooling module on top of the output of DSNet's encoder in Section IV.

2) FEATURE MAP SIZE

$1/8$ feature map size is adopted, as it is consistently proven to achieve better accuracy than other sizes [8], [20]. As smaller ones lose too much spatial information which is impossible to recover when only using methods such as bi-linear up-sampling or transposed convolution in decoder, otherwise decoder may need to fuse features from encoder to make up spatial information loss which certainly adds more computation. Hence, we determine to keep $1/8$ feature map size in our main layers to remain spatial information as much as possible.

D. LOSS FUNCTION

A major class is usually large and easy to classify, and quickly contributes little useful information during training. However, the overwhelming number makes it account for most of the loss value. While a minor class is often underrepresented and at the same time hard to classify. The numerous imbalance gap in number makes minor class drowned in the loss value contribution. It is the minor class that should attract more attention during the training process.

We propose a novel Object Weighted Focal Loss (OWFL) to handle the class imbalance issue in semantic segmentation task, and Semantic Encoding Loss (SEL) is also adopted to aggregate more context information. Our final loss is shown in Equation (1), where L is the total loss, and we experimentally set $\lambda_1 = \lambda_2 = 1$, as we value both the imbalance class and the context information of the network.

$$L = \lambda_1 \text{OWFL} + \lambda_2 \text{SEL} \tag{1}$$

1) OBJECT WEIGHTED FOCAL LOSS (OWFL)

Underrepresented object is often difficult to classify, thus requires more attention of the model. The motivation of OWFL is to make the minor and hard objects contribute more information to the loss function without affecting other objects. To achieve that, normalized object frequency weight and object order-of-magnitude weight are jointly utilized to control the loss value contribution of the object.

Object frequency weight is obtained by $\omega_i = \frac{1}{ln(f_i+c)}$, which is proposed in ENet [7], and we set $c = 1.02$ following ENet. f_i is the frequency of the ith object appeared in the dataset. Different from ENet controlling ω_i in the range of $[1, 50]$, we normalize the weights to $[0, 1]$ by dividing the maximum:

$$\alpha_i = \frac{\omega_i}{max(\omega_i)} \tag{2}$$

Object order-of-magnitude weight is calculated by equation below:

$$\gamma_i = \text{OM}(\frac{f_i}{min(f_i)}) \tag{3}$$

where function OM is to calculate the order of magnitude of a given number. Finally, the Object Weighted Focal Loss (OWFL) is derived based on focal loss [37]:

$$\text{OWFL}(p_i) = -\alpha_i \cdot (1 - p_i)^{\gamma_i} \cdot log(p_i) \tag{4}$$

where p_i is the probability of a sample belonging to the ith object predicted by the network. The α_i balances the

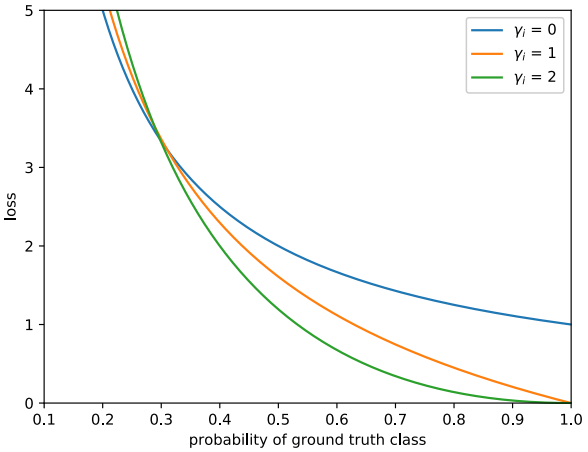

FIGURE 4. Figure of derivative function of OWFL in Equation 5 with $\gamma_i = 0, 1, 2$. It is better viewed in color.

loss contribution of each class, and the γ_i down-weights well-classified object. Their joint effect is to make hard and minor objects contribute more loss value, and immensely down-weight major and easy objects. The general derivative function for OWFL with respect to p_i is given in Equation (5). We plot the derivative function with $\gamma_i = 0, 1, 2$ in Fig.4.

$$\frac{d\text{OWFL}(p_i)}{dp_i} = \gamma_i(1-p_i)^{\gamma_i - 1} log(p_i) - \frac{(1-p_i)^{\gamma_i}}{p_i} \quad (5)$$

It shows that when objects are not well-classified, for instance its confidence is below 0.3, objects all contribute similar large gradients value. However, as confidence approaches 1.0, the gradient contribution begins to diverge, the major objects which have larger γ_i generate much small gradients, and α_i would further enlarge the gap of the gradients contribution. In this way, minor and hard objects dominate the loss value contribution. It should be noted that when class distribution is balanced, OWFL becomes class-balanced cross entropy, and when unbalanced, OWFL actually expands into multiple loss functions for different groups of objects.

2) SEMANTIC ENCODING LOSS (SEL)

We also introduce Semantic Encoding Loss in order to encode global semantic context of the scene. SEL is proposed in [16] as part of the Context Encoding Module which consists of encoding layer that encodes global semantic context, feature attention and semantic encoding loss. We adopt encoding layer and build semantic encoding loss by adding a fully connected layer with sigmoid activation function upon the output of encoding layer. SEL is only applied on the final output of encoder which is shown in Fig. 1. Feature attention module is not applied as it introduces additional computation. Context encoding layer considers an input feature map with the shape of $C \times H \times W$ as a set of C-dimensional input features $X = \{x_1, \ldots x_N\}$, where N is total number of features given by $H \times W$. It learns an codebook $D = \{d_1, \ldots d_k\}$, containing K number of code words (visual centers) and a set of smoothing factor of the visual centers $S = \{s_1, \ldots s_k\}$. Encoding Layer outputs residual encoder by aggregating the residuals with soft-assignment weights $e_k = \sum_{i=1}^{N} e_{ik}$, where

$$e_{ik} = \frac{\exp\left(-s_k \left\|r_{ik}\right\|^2\right)}{\sum_{j=1}^{K} \exp\left(-s_j \left\|r_{ij}\right\|^2\right)} r_{ik} \quad (6)$$

The residuals are given by $r_{ik} = x_i - d_k$, and $e = \sum_{k=1}^{K} \phi\left(e_k\right)$ where ϕ denotes Batch Norm with *ReLU* activation. An additional fully connected layer is built upon encoding layer, and the final SEL is calculated by sigmoid cross entropy:

$$\text{SEL}(s_i) = -t_i log(sigmoid(s_i)) \quad (7)$$

where t_i and s_i are the ground truth and output of the fully connected layer for each class i, and *sigmoid* is activation function.

We summarize the training process with OWFL and SEL in Algorithm 1.

Algorithm 1 Training With OWFL and SEL

Require: *traindata*: training dataset, *labeldata*: label for training dataset.

1: Iterate over *labeldata* to obtain ω_i and γ_i by Eq. 2 and Eq. 3.
2: **for** steps $t \in \{1, 2, \ldots k\}$ **do**
3: Sample a mini-batch of (*imgs*, *gts*) from *traindata* and *labeldata*
4: Feed into dsnet(*imgs*), and obtain *se_logits* and *cls_logits*
5: Obtain OWFL with *cls_logits* and *gt* by Eq. 4
6: Obtain *se_label* = [*histgram*(*gts*) > 0]
7: Obtain SEL with *se_logits* and *se_label* by Eq. 7
8: *total_loss* = λ_1OWFL + λ_2SEL
9: Backpropagate and update weights of dsnet
10: **end for**

IV. EXPERIMENTAL EVALUATION

In this section, we first report details about the experiment settings, especially on data augmentation and training protocol detail. Then we conduct experiments to evaluate the effectiveness of proposed model and loss function, finally the evaluation results on Cityscapes dataset and comparison with other methods are reported.

All experiments are conducted following the same data augmentation strategy, hyper-parameter settings and validated on the same validation dataset at full resolution. For different purposes, we adopt different schemes. To be specific, in ablation study of proposed model, ablation study of loss function and experiments of different sizes of DSNets (shown in Table 2), we train 120 epochs on fine annotation without pre-training and set batch size to 8. In Cityscapes dataset evaluation and comparing with other methods, we adopt pre-training on coarse labels for 80 epochs, then train on

fine labels for 120 epochs with the batch size of 16, and add another 40 epochs fine-tuning to obtain the best result. All experiments adopt synchronized multi-GPU Batch Normalization. Note that batch size is vital to effectively train CNN models, and has been proven crucial in [25].

A. DATASET AND EVALUATION METRICS

We use the Cityscapes dataset [45], a recent dataset of urban driving scenes that has been widely adopted in semantic segmentation benchmarks due to its highly challenging and varied scenarios. It consists 5000 fine-annotated images at the high-resolution of 1024×2048, which are split into 2975 images for training, 500 images for validation, and 1525 images for testing. There is another set of 19, 998 images with coarse annotation. The dense annotation contains 30 common class labels in which 19 classes are for training and evaluation. Evaluation metrics is mainly IoU, which is defined as $IoU = TP/(TP + FP + FN)$, where TP, FP, and FN are the numbers of true positive, false positive, and false negative pixels, respectively,

B. EXPERIMENTS SETUP

The details about experiment settings, including software and hardware settings, data augmentation strategy, and training details are reported. These details are important for reproducing our work.

1) HARDWARE AND SOFTWARE SETUP

We conduct our experiments on a server with Intel E5 2630 CPU which has 6 cores with 2.3 GHz base frequency, 32 GB memory, and four NVIDIA GTX 1080Ti GPU cards. The server runs Ubuntu 16.04, NVIDIA CUDA 9.0, cuDNN 7.05, and tensorflow 1.6. We use tensorpack [46] to implement our experiment which is a high-level training interface built upon tensorflow, and the tensorpack version is 0.8.9.

2) TRAINING PROTOCOL DETAIL

Pre-train. We train on coarse annotation set for 80 epochs as pre-training, and input images at resolution 512×1024 which down-samples original image by 2. We set initial learning rate to 5×10^{-4} which decreases 0.5 every 10 epochs, batch size to 12, weight decay to 5×10^{-4}, and use ADAM as optimizer.

Train. We train on fine annotation set for 120 epochs with the batch size of 16. It could start from scratch on fine annotation set or fine-tune on coarse annotation pre-trained model. In training fine annotation, we input images 800×800 performing data augmentation stated before, set batch size to 16, momentum to 0.9, and weight decay to 2×10^{-4}. The learning rate scheduling is $lr = baselr \times \left(1 - \frac{iter}{total\_iter}\right)^{power}$. The base learning rate is set to 1×10^{-4}, and the power is set to 0.9. For our final comparison with other methods, we further fine-tune another 40 epochs with initial learning rate $lr = 2 \times 10^{-5}$ and stochastic gradient descent optimizer, and save the best model.

3) DATA AUGMENTATION

Data augmentation is vital, as deep neural networks usually require huge amount of data for training. Our data augmentation strategy is mainly used in training fine annotation. We adopt cropping strategy which is widely adopted and proven beneficial in [25], [44] to augment fine annotation set. Specifically, we crop each training image and its corresponding ground truth label image into eight 880×880 patches with partial overlapping, augmenting fine annotation training dataset to 23800 images. The overlapping strategy ensures all regions in an image will be visited. Cropping not only enlarges fine annotation set, but also helps to fit more training images into one batch on GPU without losing spatial information. We employ multi-scale inputs (We could fit $scales = \{0.5, 1.0\}$) with random cropping 800×800 out of 880×880, and random horizon left and right flipping.

C. ABLATION STUDY OF PROPOSED MODEL

To evaluate the proposed DSNet, we conduct experiments to show the benefits of proposed units and architecture.

1) ABLATION STUDY OF PROPOSED UNITS

We perform two sets of experiments to evaluate the proposed units. The first set of experiments are to evaluate the components of units. We remove channel shuffle in Basic unit and Down unit (NOSF), replace 3×3 Conv in Basic unit and Down unit with depth-wise separable convolution (DW), and replace concatenate with add operation in Down unit and Basic unit which channel split is also removed, and double channel depth inside the unit (ADD). In the second set of experiments, we compare our proposed unit with MobileNet V2 unit (MBV2) and ENet unit (ENET). Various units possess different number of parameters, for fair comparison, we make adjustments in the number of units to ensure the number of parameters basically the same. In DW and ENET, another 4 units are added after Unit 1.4, 9 units after Unit 2.8, 9 units after Unit 3.8. In ADD and MBV2, they have 2 units in $1.x$ stage, and 4 units with dilation rate of 2, 5, 9, 17 in $2.x$ stage.

The results are summarized in Table 5. Channel shuffle is the essential operation of ShuffleNet unit, NOSF performs much worse without channel shuffle. ADD unit is "heavier" than DSNet unit, but it does not improve accuracy or speed performance. DW is 22 more deeper than DSNet, however, it performs much worse in accuracy, and more than 40% slower than DSNet. In addition, the training process for DW is also much longer. Depth-wise separable convolution possess much fewer parameters than standard 3×3 convolution, but in practice, it does not promise speed improvement proportional to the massive reduction in parameters. ENet unit performs inferior in both accuracy and FPS, and this indicates the advantage of DSNet unit over ENet unit. The MobileNet V2 unit has almost equal accuracy performance with DSNet unit in our experiments, however, FPS drops more than 35% under similar parameters. This suggests that DSNet unit is

TABLE 4. Class-wise IoU evaluation results on Cityscapes. Results of ERFNet val and test are from original paper and pre-trained on ImageNet.

	Wall	Fence	Traf. Light	Traf. Sign	Rider	Truck	Bus	Train	Motorcycle	Bycycle	Sider	Pole	Terrain	Sky	Person	Car	Road	Building	Vegetation
	$\gamma_i = 0$										$\gamma_i = 1$						$\gamma_i = 2$		
ERFNet(val)	54.6	54.1	62.5	71.6	55.3	67.0	77.4	59.8	41.9	68.4	82.1	59.0	69.4	94.2	78.5	93.4	97.9	90.7	91.9
ERFNet(test)	42.5	50.4	59.8	68.4	59.8	52.3	60.8	53.7	49.9	64.2	81.4	59.8	62.9	93.1	75.2	92.9	97.5	90.9	91.3
DSNet(val)	55.0	58.6	61.2	71.7	58.1	64.4	77.7	72.8	52.5	69.4	76.9	57.4	55.2	92.4	73.3	90.9	96.7	90.3	90.4
DSNet(test)	50.3	53.5	45.9	65.3	55.9	64.4	71.6	61.1	50.3	69.2	77.2	57.1	57.7	92.1	73.6	90.8	96.6	89.8	89.9

TABLE 5. Ablation study of proposed model. FPS is evaluated following inference speed experiment in Section IV at resolution 640 × 360.

Methods	mIoU	FPS	Params(M)
NOSF	50.1%	110.3	0.91
ADD	60.9%	88.2	1.15
DW	58.0%	59.2	0.89
MBV2	62.9%	64.7	1.17
ENET	60.4%	89.8	0.95
INIT	62.5%	99.1	0.91
PSP	63.2%	87.8	1.22
SKIP	62.8%	93.3	0.91
DILA	62.4%	100.5	0.91
DSNet	62.8%	100.5	0.91

more computation efficient, and equally powerful compared with MobileNet V2 unit under the same parameters and architecture.

2) ABLATION STUDY OF PROPOSED ARCHITECTURE

To evaluate our proposed units architecture and dilation rate scheme, we conduct experiments as follows: replace Init unit with a simple 3 × 3 convolution of stride 2 (INIT), add pyramid pooling module of PSPNet at the end of encoder (PSP), fuse feature maps between encoder and decoder by long range skip connections (SKIP), and adopt dilation rate scheme of 2, 4, 8, 16 at Basic unit 2.x and Basic unit 3.x (DILA).

The results are summarized in Table 5. The INIT slightly drops 0.3% point compared with DSNet, and it also slows down a little in FPS due to introduced computation. This suggests Init unit has better performance in both accuracy and speed compared with simply 3 × 3 convolution with stride of 2. Adding additional pyramid pooling module improves accuracy by 0.4% point, however increased computation results in 12% drop in speed performance. Skip connection between encoder and decoder does not bring positive improvement in our experiment, similar result is also found in ERFNet [11]. For dilation rate scheme, HDC performs better than the scheme of Deeplab V3 [25] or DRN [43]. Training with OWFL and SEL also improves ''gridding'' issue, see visual comparison in Fig. 6.

D. ABLATION STUDY OF LOSS FUNCTION

To show the effectiveness of the proposed loss function, we conduct experiments with four different loss functions: class weighted cross entropy (WCE), class weighted cross entropy and semantic encoding Loss (WCE+SEL), focal loss and semantic encoding loss (FL+SEL), and OWFL and semantic encoding loss (OWFL+SEL). 19 trainable classes in Cityscapes dataset are grouped into 3 categories according to the γ_i value which represents the object's frequency in the whole dataset. At last, 10 objects are in $\gamma_i = 0$ group, 6 objects in $\gamma_i = 1$ group, 3 objects in $\gamma_i = 2$ group, the grouping details are shown in Table 4. $\gamma_i = 0$ group represents the minor objects, and $\gamma_i = 2$ group the major objects. The mean accuracy of $\gamma_i = 0$ group is far lower than that of $\gamma_i = 1$ and $\gamma_i = 2$ group.

The result is shown in Table 6. As we can see, simple class re-weighting scheme alone performs the worst for handling seriously imbalanced dataset. Adding semantic encoding loss forces the network to aggregate more global semantic context information, and it improves the accuracy over all 3 categories for free as it does not bring any computation in inference. It also greatly alleviated the issue of misclassification inside an object, as show row 3, column c and d in Fig. 6. Focal loss with semantic encoding loss gets worse than class weighted cross entropy and semantic encoding. Our contemplation is that focal loss has unstable issue due to its ability to dynamically adjust loss value which may lead to large fluctuation during training. OWFL with semantic encoding loss achieves the best result, and significantly improves accuracy in $\gamma_i = 0$ group by 2.9% points over 10 minor objects compared with WCE. The visual improvement is shown in Fig. 6. With 3 γ_i groups, OWFL actually expands into 3 loss functions. Objects in $\gamma_i = 0$ group employ class weighted cross entropy loss function, while in $\gamma_i = 2$ adopt class weighted focal loss which the well-classified objects are heavily suppressed, thus minor and hard objects dominates the loss value contribution, and leading to the best performance. It is also worth highlighting the pre-training on coarse annotation dataset. As coarse annotation dataset mainly consists of large geometric shapes, large and easy objects are already well-classified in pre-training phase, therefore training on fine annotation could almost entirely focus on minor and hard objects (See Fig. 5).

TABLE 6. mean IoU results with four different loss function settings.

Loss Function	$\gamma_i = 0$	$\gamma_i = 1$	$\gamma_i = 2$
WCE	58.5%	72.5%	91.3%
WCE+SEL	60.0%	74.1%	91.8%
FL+SEL	59.2%	73.3%	91.2%
OWFL+SEL	61.4%	74.1%	92.0%

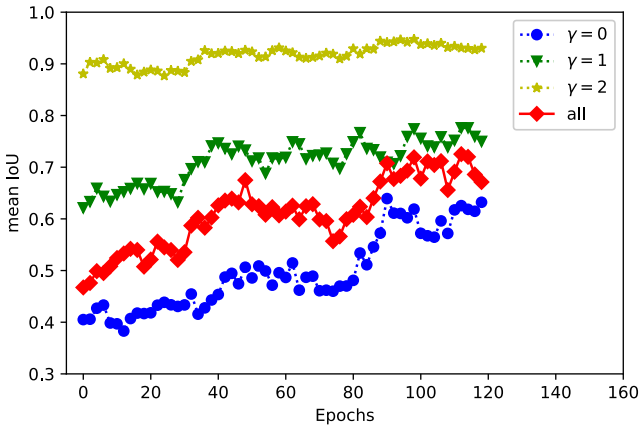

FIGURE 5. Validation mean IoU of group $\gamma_i = 0, 1, 2$ and all classes during training.

E. CITYSCAPES DATASET EVALUATION

We show main evaluation results on Cityscapes dataset, and compared with other semantic segmentation methods. Results include comprehensive metrics of DSNet with other methods, class-wise IoU results of DSNet and ERFNet, as they have comparable accuracy, and a qualitative results which displays visual comparison of DSNet with only class weighted cross entropy and DSNet with OWFL, SEL and HDC.

1) MEAN IOU

We list comprehensive metrics and results of DSNet and other methods including mean IoU, inference time and number of parameters, shown in Table 1. DSNet without any base model or ImageNet pre-training could achieve 69.3% mean IoU, which is one of the excellent results among lightweight semantic segmentation methods. DSNet is much higher in accuracy than lightweight semantic segmentation methods which focus on reducing the number of parameters, such as ESPNet [10] and ENet [7], and is also more accurate than some classical cumbersome models, such as Dilation10 [17], FCN-8s [4] and DeepLab V1 [6]. To be specific, DSNet has 148 times fewer parameters than Dilation10, but 2.1% points higher in accuracy. DSNet is close to ICNet [13] and ERFNet [11] which pre-trained on large-scale ImageNet dataset. With 69.3% mIoU and 0.91M parameters, DSNet achieves excellent trade-off.

2) CLASS-WISE IOU

Class-wise IoU is shown in Table 4 where we compare DSNet with ERFNet on every trainable classes on validation and test set, since the result of ERFNet which pre-trained on ImageNet has very close mean IoU result with DSNet. The mean IoUs of ENet, ERFNet and DSNet over $\gamma_i = 0$ group are displayed in Table 7. ENet* and ERFNet* are trained using the same protocol as DSNet, but with WCE as loss function. We could obtain ENet* as high as 61.5%, and ERFNet* 68.6% which is 1.4% points lower than the result 70.0% of ERFNet without ImageNet pretraining. Our training hyper parameters and protocol may not be the best fit for ERFNet.

Generally speaking, with similar mean IoU result, DSNet scores better at $\gamma_i = 0$ group both at validation set and test set, which is shown is Table 7. DSNet is consistently 2.7% points higher than ERFNet [11]. This suggests that DSNet with OWFL does improve minor and hard objects, and generalize well to test set. In Table 4, we also observe significantly drop in some certain minor and hard objects between validation and test dataset. For instance, wall, truck and bus in ERFNet, traffic light and train in DSNet drop more than 10% points. The performance drop is mainly due to the difference between validation and test dataset distribution. Besides, minor objects are severely short for diversity, and the model may not to able to learn well-generalized features from limited data. The validation mean IoU during the training process is depicted in Fig. 5. The training starts upon pre-training phase. As we can see, $\gamma = 2$ group which consists of major and easy objects has very high accuracy after pre-training on coarse labels, thus the training specially focuses on minor and hard objects with OWFL as the $\gamma = 0$ group improves significantly during training. The IoU during training also exhibits fluctuation for minor and hard objects which could explain DSNet's IoU of traffic light is worse than ERFNet. OWFL could bring benefit to the overall improvement of the group of minor and hard objects, but can not guarantee every object is better than ERFNet.

3) VISUAL COMPARISON

To intuitively understand the performance of proposed DSNet, we select some images from validation set, and visually shows our proposed methods beyond metrics. In Fig. 6, column *c* is prediction results by DSNet with class weighted cross entropy and the dilation rate scheme of $2, 4, 8, 16$ (DSNet (WCE)), column *d* is DSNet with OWFL and semantic encoding loss and dilation rate scheme of HDC (DSNet (OWFL+SEL+HDC)). Both results are delivering fine quality of the driving scene. However, if we focus on the white boxes which most are minor and hard objects, DSNet (OWFL+SEL+HDC) performs much finer. For example, in the second row, DSNet (OWFL+SEL+HDC) is segmenting the contour of a rider much finely. In the third row, without context aggregation provided by context encoding layer, DSNet (WCE) makes wrong predictions in the window of the train. In the fifth row, train and bus are misclassified in DSNet (WCE), DSNet (OWFL+SEL+HDC) could precisely handle. In the last row, DSNet (WCE) shows "gridding" issue, it is very much improved with the help of HDC and context encoding layer. Overall speaking,

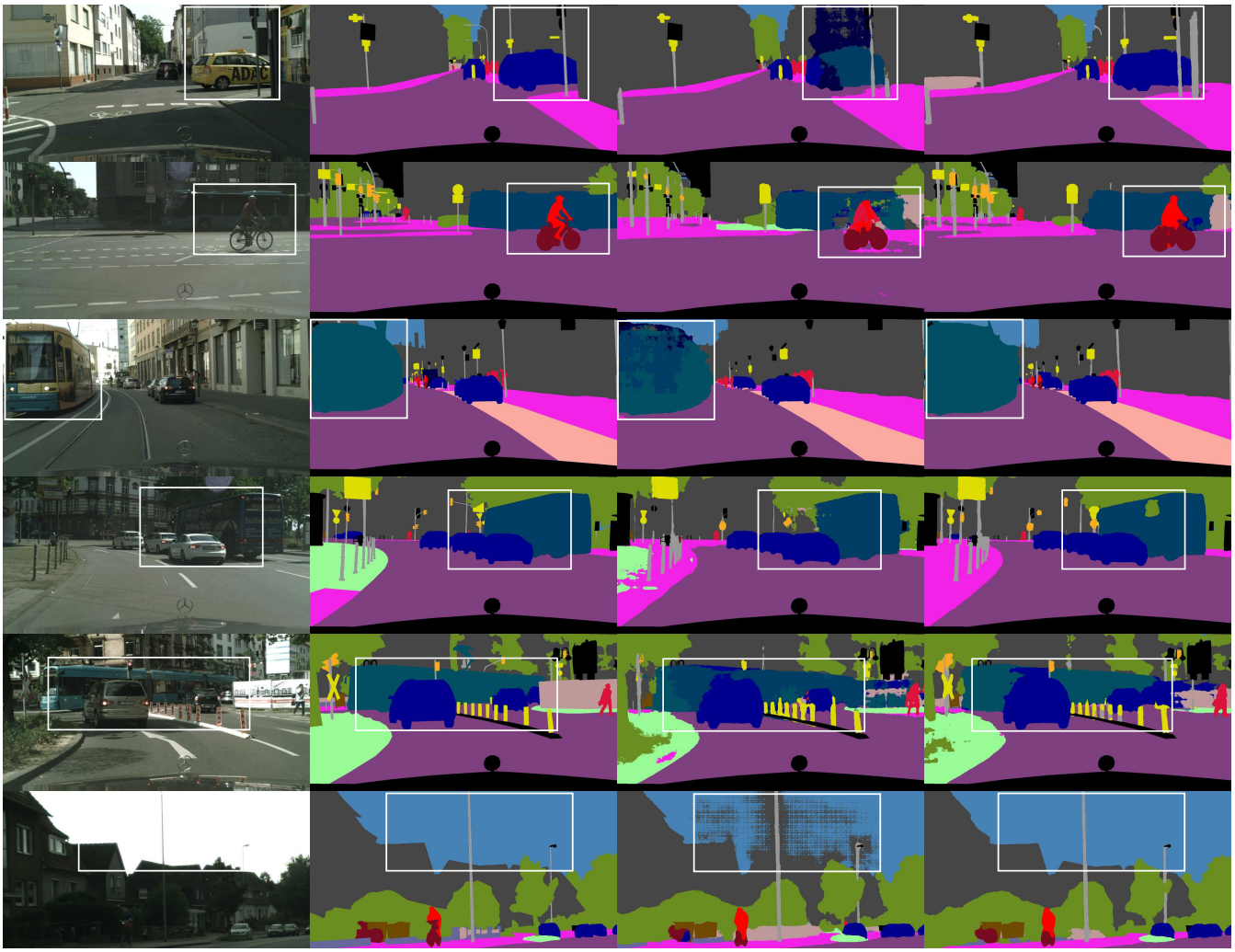

FIGURE 6. Qualitative results of proposed method on the Cityscapes validation dataset, better viewed in color. (a) Input image. (b) Ground truth. (c) DSNet(WCE). (d) DSNet (OWFL+SEL+HDC). It is better viewed in color.

TABLE 7. mean IoU results on cityscapes of ERFNet and DSNet. ENet* and ERFNet* are trained using the same training protocol with DSNet except that using WCE as loss function. Results of ERFNet val and test are pre-trained on ImageNet.

Methods	mIoU($\gamma_i = 0$)	mIoU(all)
ENet*(val)	48.0%	61.5%
ERFNet*(val)	57.3%	68.6%
ERFNet(val)	61.3%	71.5%
DSNet(val)	64.1%	71.8%
ERFNet(test)	56.1%	69.7%
DSNet(test)	58.8%	69.3%

TABLE 8. Speed analysis of ENet, ERFNet and DSNet. ENet* and ERFNet* are re-implemented in tensorpack and speed are tested under the same settings for fair comparison, results may be different from original paper.

Methods	NVIDIA 1080Ti					
	640 × 360		1280 × 720		1920 × 1080	
	ms	FPS	ms	FPS	ms	FPS
ENet*	11.1	89.7	57.8	17.3	126.6	7.9
ERFNet*	12.5	79.9	74.3	13.4	165.2	6.0
DSNet(Ours)	9.9	100.5	49.8	20.1	102.0	9.8

DSNet (OWFL+SEL+HDC) is more capable of refining hard and minor objects, and delivering fast and accurate semantic information of the driving scene.

F. INFERENCE SPEED

Inference speed is a very important metric in evaluating efficient CNN models. While speed is also very difficult to reproduce, as it is determined by many uncontrolled factors, especially evaluating settings vary in different research works. For research purpose and fair comparison, we re-implement ENet and ERFNet using tensorpack based on open source code [47], and evaluate speed of ENet, ERFNet and our model under the same setting. We load variables necessary for inference and drop all the other variables

in saved checkpoint files, and only count inference time for each image. We feed 100 images one by one to calculate average inference time per image for ten times. Inference evaluation is carried out on single NVIDIA 1080Ti GPU card. The results are shown in Table 8. From the results, we can see that the inference speed of DSNet outperforms ENet by a small margin at every input scale, and approximately 1.1 times faster than ENet. Compared with ERFNet, DSNet is 25%+ faster at every scale.

V. CONCLUSION

In this paper, we propose a lightweight CNN model termed as DSNet and a novel lossfunction Object Weighted Focal Loss. DSNet achieves excellent trade-off among model size, accuracy and inference speed. Specifically, DSNet has 0.9M parameters, 69.3% mean IoU on Cityscapes dataset, and runs more than 100 FPS on NVIDIA 1080Ti. In order to deal with severe class imbalance issue and improve minor and hard objects accuracy, Object Weighted Focal Loss is proposed. It adopts normalized object frequency weight and object order of magnitude weight to make minor object contribute more loss value and greatly suppress contribution from the major and well-classified objects. Experiments show that OWFL together with semantic encoding loss effectively improves minor objects accuracy. Therefore, DSNet is promising for practical application.

REFERENCES

[1] B. Ranft and C. Stiller, "The role of machine vision for intelligent vehicles," *IEEE Trans. Intell. Vehicles*, vol. 1, no. 1, pp. 8–19, Mar. 2016.

[2] J. Janai, F. Güney, A. Behl, and A. Geiger, "Computer vision for autonomous vehicles: Problems, datasets and state of the art," 2017, *arXiv:1704.05519*. [Online]. Available: http://arxiv.org/abs/1704.05519

[3] H. Zhu, F. Meng, J. Cai, and S. Lu, "Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation," *J. Vis. Commun. Image Represent.*, vol. 34, pp. 12–27, Jan. 2016.

[4] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[5] 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.

[6] 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, Apr. 2018, doi: 10.1109/TPAMI.2017.2699184.

[7] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*. [Online]. Available: http://arxiv.org/abs/1606.02147

[8] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6230–6239.

[9] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5168–5177.

[10] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," 2018, *arXiv:1803.06815*. [Online]. Available: http://arxiv.org/abs/1803.06815

[11] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized ConvNet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, Jan. 2018, doi: 10.1109/TITS.2017.2750080.

[12] S.-Y. Lo, H.-M. Hang, S.-W. Chan, and J.-J. Lin, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," 2018, *arXiv:1809.06323*. [Online]. Available: http://arxiv.org/abs/1809.06323

[13] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *Proc. 15th Eur. Conf. (ECCV)*. Munich, Germany: Springer, Sep. 2018, pp. 418–434.

[14] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 325–341.

[15] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131.

[16] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, "Context encoding for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7151–7160.

[17] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. ICLR*, 2016, pp. 1–13.

[18] T. Wu, S. Tang, R. Zhang, and Y. Zhang, "CGNet: A light-weight context guided network for semantic segmentation," 2018, *arXiv:1811.08201*. [Online]. Available: http://arxiv.org/abs/1811.08201

[19] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jun. 2014.

[20] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 801–818.

[21] P. H. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene labeling," in *Proc. 31st Int. Conf. Mach. Learn. (ICML)*, 2014, pp. 82–90.

[22] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki, "Scene labeling with LSTM recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3547–3555.

[23] B. Shuai, Z. Zuo, B. Wang, and G. Wang, "DAG-recurrent neural networks for scene labeling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3620–3629.

[24] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Int. Conf. (MICCAI)*, Munich, Germany, Oct. 2015, pp. 234–241.

[25] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*. [Online]. Available: http://arxiv.org/abs/1706.05587

[26] J. Zhuang, J. Yang, L. Gu, and N. Dvornek, "ShelfNet for fast semantic segmentation," 2018, *arXiv:1811.11254*. [Online]. Available: http://arxiv.org/abs/1811.11254

[27] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[29] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Netw.*, vol. 106, pp. 249–259, Oct. 2018.

[30] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert Syst. Appl.*, vol. 73, pp. 220–239, May 2017.

[31] M. Siam, S. Elkerdawy, M. Jagersand, and S. Yogamani, "Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–8.

[32] K. Oksuz, B. Can Cam, S. Kalkan, and E. Akbas, "Imbalance problems in object detection: A review," 2019, *arXiv:1909.00169*. [Online]. Available: http://arxiv.org/abs/1909.00169

[33] J. Jiao, Y. Cao, Y. Song, and R. Lau, "Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 53–69.

[34] L. Shen, Z. Lin, and Q. Huang, "Relay backpropagation for effective learning of deep convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 467–482. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-46478-7_29

[35] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9268–9277.

[36] B. Li, Y. Liu, and X. Wang, "Gradient harmonized single-stage detector," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 8577–8584.
[37] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy: IEEE Computer Society, Oct. 2017, pp. 2999–3007.
[38] Z. Wu, C. Shen, and A. van den Hengel, "Bridging category-level and instance-level semantic image segmentation," 2016, *arXiv:1605.06885*. [Online]. Available: http://arxiv.org/abs/1605.06885
[39] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 761–769.
[40] Y. Yuan and J. Wang, "OCNet: Object context network for scene parsing," 2018, *arXiv:1809.00916*. [Online]. Available: http://arxiv.org/abs/1809.00916
[41] A. Briot, P. Viswanath, and S. Yogamani, "Analysis of efficient CNN design techniques for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 663–672.
[42] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Honolulu, HI, USA: IEEE Computer Society, Jul. 2017, pp. 2261–2269.
[43] F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 472–480.
[44] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding convolution for semantic segmentation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Lake Tahoe, NV, USA, Mar. 2018, pp. 1451–1460, doi: 10.1109/WACV.2018.00163.
[45] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[46] Y. Wu. (2016). *Tensorpack*. [Online]. Available: https://github.com/tensorpack/
[47] K. Sin. (2017). *TensorFlow-ENet*. [Online]. Available: https://github.com/kwotsin/TensorFlow-ENet

WENFU WANG received the B.S. degree from the College of Automobile Engineering, Jilin University, Jilin, China, in 2011. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. His current research interests include computer vision and autonomous driving.

YONGJIAN FU received the bachelor's degree in computer sciences from the School of Computer Science, Wuhan University, Wuhan, China, in 2018. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. His current research interests include computer vision, deep learning, and autonomous driving.

ZHIJIE PAN is currently the Director of the Intelligent Vehicle Research Center, Zhejiang University. He is also mainly involved in the cross research fields of computer science and automotive, including AI, autonomous vehicle technology, automatic driving, and ITS. He has presented a series of new concepts in the fields of intelligent electric vehicle, chassis-control-by-wire technology, intelligent driving, unmanned systems, smart city, intelligent transportation, and applications. He has published more than 90 articles in international conferences and international magazines IEEE, SAE, and JSAE. He has more than 100 patents. He has served as the Keynote Speaker in international conferences. He has won the China Automobile Industry Science and Technology Award First Prize, the China Quality Evaluation Association Science and Technology Innovation Award First Prize, the National Federation of Industry and Commerce Science and Technology Progress Award First Prize, and the Zhejiang Science and Technology Award First Prize.

XI LI received the Ph.D. degree from the National Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing, China, in 2009. From 2009 to 2010, he was a Postdoctoral Researcher with CNRS, Telecom ParisTech, France. He was a Senior Researcher with The University of Adelaide, Australia. He is currently a Full Professor with Zhejiang University, China. His research interests include visual tracking, motion analysis, face recognition, web data mining, and image and video retrieval.

YUETING ZHUANG received the B.Sc., M.Sc., and Ph.D. degrees in computer science from Zhejiang University, China, in 1986, 1989, and 1998, respectively. From February 1997 to August 1998, he was a Visiting Scholar with the University of Illinois at Urbana-Champaign. He has served as the Dean of the College of Computer Science, Zhejiang University, from 2009 to 2017, the Director of the Institute of Artificial Intelligence, from 2006 to 2015. He is currently a Full Professor with the College of Computer Science, the Director of the MOE-Digital Library Engineering Research Center, Zhejiang University. His research interests mainly include multimedia retrieval, artificial intelligence, cross-media computing, and digital library.