DSNet for Real-Time Driving Scene Semantic Segmentation

Preprint, compiled December 19, 2018

Wenfu Wang\textsuperscript{1} and Zhijie Pan\textsuperscript{1}

\textsuperscript{1} Department of Computer Science, Zhejiang University

Abstract

We focus on the very challenging task of semantic segmentation for autonomous driving system. It must deliver decent semantic segmentation result for traffic critical objects real-time. In this paper, we propose a very efficient yet powerful deep neural network for driving scene semantic segmentation termed as Driving Segmentation Network (DSNet). DSNet achieves state-of-the-art balance between accuracy and inference speed through efficient units and architecture design inspired by ShuffleNet V2 and ENet. More importantly, DSNet highlights classes most critical with driving decision making through our novel Driving Importance-weighted Loss. We evaluate DSNet on Cityscapes dataset, our DSNet achieves 71.8\% mean Intersection-over-Union (IoU) on validation set and 69.3\% on test set. Class-wise IoU scores show that Driving Importance-weighted Loss could improve most driving critical classes by a large margin. Compared with ENet, DSNet is 18.9\% more accurate and 1.1+ times faster which implies great potential for autonomous driving application.

1 Introduction

Driving perception is a vital and challenging task for urban autonomous driving. An autonomous vehicle must comprehensively understand the driving scene in the highly complex and dynamic urban traffic environment, then make the safest driving decision. With the remarkable progress of deep learning and rich information contained in image, computer vision is playing an increasingly more important role in driving perception. Image semantic segmentation could obtain exhaustive information such as object categories, shape, spatial location at pixel level, thus are especially beneficial for 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, and in recent years, Convolutional Neural Network (CNN) based methods achieve remarkable progress in image semantic segmentation tasks, significantly improving accuracy even efficiency, and has become the de facto solution. However, state-of-the-art semantic segmentation methods are pursing higher accuracy and finer quality with increasingly larger number of parameters and complex modules. Although vehicles are getting increasingly more computation power, many semantic segmentation models are still too heavy to run on embedded devices. For safe-critical driving perception system, it must deliver real-time performance.

The quest for light-weight semantic segmentation models has gained more attention recently. Many of the methods focus on light aspect of semantic segmentation, for example ENet, ESPNet, Light-weight RefineNet and so on. These methods achieve real-time performance by heavily compressing parameters but sacrifice too much accuracy that their overall accuracy is about 60\% where non-critical objects such as sky and road contribute much to it. Their segmentation quality especially of driving-critical classes is quite low that it can not provide sufficient information for safe autonomous driving. Efficient semantic segmentation models should take both accuracy and speed into account. ICNet and ERFNet achieves good balance between accuracy and real-time performance, however we argue that their generic methods overlook one important fact that objects have natural priority in driving task decision making.

Obviously, objects have different importance levels in traffic environment. For instance, it is by nature that person and vehicle have higher priority in traffic environment than sky and tree. Driving perception should focus on objects that have high priority, i.e., high importance level with driving task, since more accurate result on these objects lead to safer decision during driving. While for the task of image semantic segmentation, it assumes every object has the same importance level, and the goal is to segment every object as well as possible. Hence, driving importance priority needs to be introduced in terms of driving scene semantic segmentation, and the goal should be to segment high driving importance level objects to its best. To sum up, we argue that ideal driving semantic segmentation system poses two challenging demands: tradeoff between runtime performance and accuracy, and highlight on driving pivotal objects.

In this paper, we aim to design a practical semantic segmentation model for autonomous driving system which not only achieves great balance between accuracy and speed but highlights most driving critical objects. Different from extensively compress the number of parameters like ENet, we determine to increase the number of parameters (it is still a light model compared to others) and adopt efficient and powerful operations to ensure decent quality and speed at the same time. We also introduce the notion of driving importance level, and guide our network to highlight high driving importance objects with a novel loss function design. In summary, our main contributions are as following.

- We design efficient and powerful unit and asymmetric encoder-decoder architecture inspired by ShuffleNet V2 and ENet. We also employ Hybrid Dilation Convolution scheme to overcome “gridding” issue.

- We propose Driving Importance Level and Driving Importance-weighted Loss which assigns larger...
weights to classes with higher driving importance level. We also introduce context encoding loss [16], which concentrates context information in large objects.

- The proposed Driving Segmentation Network (DSNet) achieves state-of-the-art balance between accuracy and speed. It obtains 69.3\% on Cityscapes test set, and 1.1+ times faster than ENet under the same setting.

The rest of this paper is organized as follows. Section 2 reviews CNN techniques of semantic segmentation, light-weight models and class imbalance issues. Section 3 discusses computation complexity, and introduces the units, architecture and loss function of our Driving Segmentation Network. Section 4 reports our evaluation results on Cityscapes dataset. Section 5 draws the conclusion.

2 Background and Related Work

In this section, we briefly review representative CNN based techniques of semantic segmentation, recent development on light-weight semantic segmentation models, and other related research works.

2.1 Semantic Segmentation Methods

The milestone work of CNN model successfully applied on image semantic segmentation is Fully Convolutional Network (FCN) [17]. It utilizes ImageNet [18] pre-trained models, AlexNet [19], VGG [20], and GoogLeNet [21] as feature extractors, replaces fully connected layers with convolutional ones to obtain dense pixel-wise prediction map. FCN achieves great improvement in accuracy than traditional methods on PASCAL VOC [22] dataset. But more importantly, FCN for the first time demonstrates how to use a CNN model with end-to-end training to solve image semantic segmentation problems with arbitrary input image size. This triggers a research boom of various CNN models on image semantic segmentation, to name a few representative work, SegNet [23] Dilation10 [24], DeepLab V3+ [3], PSPNet [4] and ICNet [9]. We summarize them into: Encoder-Decoder architecture, Dilated convolution, and Multi-scale feature fusion method. Other methods, such as conditional random field (CRF) and Recurrent Neural Network (RNN) are not reviewed, for more comprehensive review, please read [25].

2.1.1 Encoder-Decoder Architecture

This technique targets spatial information loss. FCN adopts pooling operation after convolution which is a standard operation in classification methods. However, pooling operation has one disadvantage of deterministically losing spatial information especially small objects, which cannot be recovered by up-sampling methods. The first works attempting to improve decoder and claiming encoder-decoder architecture are SegNet[23], U-net [26] and DeconvNet [27] where they all adopt symmetric encoder-decoder architecture. In SegNet, decoder is composed by a set of un-pooling layers which corresponds to its pooling counterpart in encoder. The un-pooling operation uses the index from its counterpart pooling operation. In DeconvNet, the convolution layer in decoder is replaced with deconvolution layers. The decoder of U-net concatenate feature maps which is cropping and copying from encoder counterpart. By leveraging feature maps from encoder, encoder-decoder architecture makes up the spatial information loss during pooling operation. However, this symmetric architecture doubles the number of parameters, and it has been demonstrated that decoder has less impact on final result, thus asymmetric architecture has been widely used.

2.1.2 Dilated Convolution

Global context information is vital for semantic segmentation task. Stacks of convolution layers and pooling layers could enlarge receptive field of convolution kernel and aggregate more context information, but lose spatial information. [24] first employed dilated convolution (also called atrous convolution) in cascade in semantic segmentation CNN models. Compared with pooling operation, dilated convolution can exponentially expand receptive field without losing spatial information and increasing parameters. In DeepLab series [28, 29], the author highlighted the use of atrous convolution and proposed Atrous Spatial Pyramid Pooling (ASPP) module to aggregate object and context information at different scales. DeepLab series achieve supreme result on many semantic segmentation datasets. While, cascading dilated convolution with even dilation rates may bring the so called "gridding" issue. In [15], the author propose Hybrid Dilated Convolution (HDC) to solve this issue. The success of dilated convolution demonstrates that large field of view is beneficial for better segmentation performance.

2.1.3 Multi-scale feature fusion

Dilated convolution help to aggregate more global context information by enlarging the view point of convolution kernel, fusing features with different scales and sub-regions would further reduce global context information loss. In [30], the author first applied spatial pyramid pooling in deep neural network. PSPNet [4] propose Pyramid Pooling Module in semantic segmentation task, which uses global average pooling to fuse feature maps at four different hierarchical scales in encoder stage. The ASPP module in DeepLab V3 combines dilated convolution and multi-scale feature fusion. RefineNet [5] propose Refine module which takes one feature map and its lower scale feature map in the encoder, and fuse them as feature map in the decoder.

2.2 Light-weight Models

The research of light-weight semantic segmentation models which run fast without sacrificing much accuracy is far behind compared to research pursing higher accuracy, we would like to point out that this line of research is as important as finer models. Since it could inspire many promising real world applications where limited computation power is available.

ENet first implement a fast semantic segmentation CNN model which is 18 times faster and 1.3 points more accuracy than SegNet. ENet adopts many techniques to massively reduce parameters, for instance adopting a large number of factorized convolutions, employing asymmetric encoder-decoder architecture. This leads to 79 times reduction in parameters compared to SegNet. Many following works focus on reducing parameters, such as ESPNet [7] and SQ [31]. However, drastically reducing parameters would lead to under-fitting issue [9] for dense pixel-wise semantic segmentation task.
Recent works consider accuracy and speed at the same time. ERFNet utilize factorized convolutions to its best, and the proposed ERFNet has 5.8 times more parameters than ENet and achieves good balance on Cityscapes dataset. ICNet propose a PSPNet-based architecture. They input three scales of image, small scale image goes through deeper networks, large scale shallower, and fusion three scales of features through cascade feature unit. However, ICNet focuses on high resolution input, this hierarchical structure may not achieve proportional performance at lower resolution input, and ICNet achieves this speed by aggressive model compression.

Light-weight classification CNN models have been an active research field where researchers propose and employ many efficient operations which could be important inspiration for semantic segmentation task. MobileNet [32] proposes depth-wise separable convolution to replace standard convolution. ShuffleNet [33] uses group convolution to reduce parameters and channel shuffle operations. ShuffleNet V2 [14] further proposes a very efficient unit which reduces memory access cost.

2.3 Class Imbalance

Class imbalance issue refers to the problem where the disparity in the proportion of different classes in the whole dataset is overwhelming, and it is common among datasets. For example, in Cityscapes dataset [11], classes such as road and sky appear in every image and occupy a large region of the image, thus they have far more training pixels than other small and infrequent classes such as pole and fences. The minority classes would be drowned during training. By assigning frequency-balanced weights to cross entropy loss, this issue could be alleviated. Another issue related with class imbalance is easy and hard sample. In online bootstrapped cross entropy loss [34], it simply drops loss value from easy examples. In focal loss [35], it down-weights the loss assigned to well-classified classes.

The re-weighting techniques ease class imbalance issue, while the classes still have equal importance level. In [36], the author propose the notion of driving importance and importance-aware loss. This inspires us to combine class re-weighting techniques and human assigned driving importance.

3 Designing Driving Segmentation Network

In designing DSNet, we keep in mind that both accuracy and speed are important. We first discuss important runtime performance metrics, and then explain in detail about DSNet units, architecture, and the design choice. At last, we propose our novel loss function design.

3.1 Computation Complexity

Inference speed or frames per second (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 light-weight CNN models: number of parameters and number of float-point operations (FLOPs). Another vital metric, memory access cost, is also crucial but seriously underestimated.

Designing efficient CNN models should first consider the appropriate number of parameters which is one of the most fundamental factors affecting both accuracy and speed. It is common wisdom that larger number of parameters would usually promise higher accuracy in deep learning. While larger number of parameters means heavier computation burden that it is not affordable for mobile devices. We need to significantly reduce the number of parameters compared to large models. However more importantly, we should also avoid under-fitting issue incurred by over-reducing the number of parameters such as ENet. FLOPs is also an important metric. Less FLOPs promise less runtime of cuda kernels.

Besides parameters and FLOPs, memory access cost (MAC) is also a very crucial metric. MAC is difficult to quantize and hardware platform specific and thus are often severely overlooked. MAC metric is introduced in ShuffleNet V2 [14], and in the paper it shows with excessive experiments that MAC is vital to speed performance. It also proposes 4 practical guidelines to design efficient CNN mainly in order to reduce MAC. The 4 guidelines are listed below for completeness, we follow these guidelines, and adopt ShuffleNet V2 module as our main reference, since this module is highly efficient and powerful.

- Guideline 1: Equal channel width minimizes MAC
- Guideline 2: Excessive group convolution increases MAC
- Guideline 3: Network fragmentation reduces degree of parallelism
- Guideline 4: Element-wise operations are non-negligible

To sum up, the sensible paradigm is not to achieve light by drastically reducing parameters, but seek to design efficient and powerful models with reasonable amount of parameters and FLOPs. We compare DSNet with other methods in Table 2. DSNet is about 2.25 times in parameters and 1.8 times in GFLOPs larger than ENet, but still it is a tiny model compared with many state-of-the-art models such as Dilation10 and FCN-8s, even some light-weight models such as ERFNet which is 2 times larger than our DSNet. Our experiments will show that DSNet with more than 2 times larger parameters outperform ENet in speed and much more accurate. We believe this is the contribution of the efficient and powerful design.

3.2 DSNet Units

Our DSNet Unit is shown in Fig. 1. 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. Depthwise separable convolution in ShuffleNet V2 is replaced with dilated convolution to enlarge receptive field, which is vital for semantic segmentation task. Note that although depthwise separable convolution is proven to be an efficient operation in MobileNet V2 [37], we find this operation is not well implemented in current tensorflow [38] library, and decide to use standard convolution, as 3 × 3 convolution is the most efficient implementation. The feature channel of convolution layer in the units has equal channel width following Guideline 1 to reduce MAC. Equal channel width almost doubles the number of parameters compared with ENet. We also adopt factorized convolution when dilation rate is 1 for further reducing parameters. In down-sample unit, input
is max pooling with index following $1 \times 1$ convolution in left branch of the unit, and in up-sample unit, input is un-pool with pooling index from corresponding down-sample unit. In the final part, down-sample unit perform concatenate and channel shuffle like basic unit, while up-sample unit adds left and right branch features. The adding 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 [39], since half of the feature channels directly go through the block and join the next block. As stacks of units have different receptive field, we believe this unit design achieves multi-scale feature fusion in some sense. This explains our accuracy outperforms many works which employ multi-scale feature fusion.

### 3.3 DSNet Architecture

The DSNet architecture is shown in Table 1. We determine to adopt asymmetric encoder-decoder architecture as ENet. The asymmetric architecture has three main stages as encoder, two light stages as decoder. The structure of ENet’s architecture is a thoroughly considered choice, and it is also adopted by ERFNet [10]. Our early experiment in designing architecture also proves its success. In determining dilation rates, we follow the scheme in Hybrid Dilated Convolution [15] which successfully overcome “gridding” issue.

#### 3.3.1 Multi-scale feature fusion

Multi-scale feature fusion is indeed beneficial for better accuracy, as we discuss in Section 2.1.3. However, our concern is that multi-scale feature fusion usually adds more paths, and this violates the degree of parallelism in Guideline 3 and brings additional runtime. In designing DSNet architecture, we decide not to utilize multi-scale feature fusion, since our basic unit design in some sense achieves it, and we also adopt large feature map size. These give us confidence to give up multi-scale feature fusion.

#### 3.3.2 Feature map size

We adopt $1/8$ feature map size which is consistently proven to have better accuracy than other sizes [3, 4]. As smaller one loses too much spatial information which is impossible to recover when only using methods such as bilinear up-sampling or transposed convolution in decoder, or decoder may need to fuse feature from encoder to make up spatial information loss which certainly adds more computation. Although large feature map size does consume more memory, we determine to keep $1/8$ feature map size in our main layers to remain spatial information as much as possible.

### 3.4 Loss Function

We would like to improve accuracy on high driving importance classes in DSNet without adding extra computation. And we turn to loss function design to guide our network to highlight the object classes we care about. We propose Driving Importance-weighted Loss and introduce context encoding loss to achieve our goal. Our final loss is shown in equation (1), where $L$ is total loss, $DIL$ and $CEL$ is short for Driving Importance-weighted Loss and Context Encoding Loss respectively. $\lambda_1$ and $\lambda_2$ is the coefficient, and we experimentally set $\lambda_1 = \lambda_2 = 1$.

$$L = \lambda_1 DIL + \lambda_2 CEL$$  

#### 3.4.1 Driving Importance-weighted Loss (DIL)

The Driving Importance-weighted Loss combines class frequency weights and driving importance level. For class frequency weights, the weighting scheme in ENet is adopted to remedy data imbalance issue. Class weights are re-balanced based on equation (2). It assigns larger weights to classes who has less proportion of samples in the whole dataset. $c$ in equation (2) is a hyper-parameter, it restricts the interval of class weights value, and also acts as smoothing factor of the class weights. We set $c = 1.02$, the same as in ENet.

$$\omega_{class} = \frac{1}{\ln (c + p_{class})}$$  

Figure 1: DSNet Units. (a) Init unit. (b) Basic unit. (c) Down unit. (d) Up unit.
We also introduce Context Encoding Loss in order to encode global contextual information. The proposed Context Encoding Module in [16] consists of context encoding Layer, feature attention and context encoding loss. We did not adopt the whole module but only the context encoding loss, as feature attention module 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

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

The residuals are given by $r_{ik} = x_i - d_k$. The final loss is, $CEL = \sum_{k=1}^{K} \phi(e_k)$ where $\phi$ denotes Batch Norm [40] with ReLU activation.

### 4 Experimental Evaluation

In this section, first we report details about the experiment setting, especially on data augmentation approaches and training protocols. Then we compare our evaluation results with other methods on Cityscapes dataset.

#### 4.1 Dataset and Evaluation Metrics

We use the Cityscapes dataset [11], a recent dataset of urban scenes that has been widely adopted in semantic segmentation benchmarks due to its highly challenging and varied scenarios. It consists of 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.

#### 4.2 Experiments setup

We evaluate our model on Cityscapes dataset with mean IoU, class-wise IoU, and FPS as metrics. We give details about experiment settings, including software and hardware settings,
Table 2: 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 |
|----------|-----------|------------|------------|-----|---------|------------|--------|
| ICNet [9] | PSPNet    | ImN        | 69.5       | no  | 0.033   | 30.3       | n/a    |
| ERFNet [10] | no       | no         | 68.0       | 2   | 0.024   | 41.7       | 2.1M   |
| Dilat10 [24] | VGG16    | ImN        | 67.1       | no  | 4       | 0.25       | 140.8M |
| FCN-8s [17] | VGG16    | ImN        | 65.3       | no  | 0.5     | 2          | 134.5M |
| Deeplab [28] | VGG16    | ImN        | 63.1       | no  | 4       | 0.25       | n/a    |
| ESPNet [7]  | PSPNet    | ImN        | 60.3       | 2   | 0.009   | 112.9      | 0.36M  |
| SegNet [23] | VGG16    | ImN        | 57.0       | 4   | 0.060   | 16.7       | 29.5M  |
| ENE [6] *  | no       | no         | 58.3       | 2   | 0.032   | 30.8       | 0.36M  |
| ENEPlus *   | no       | no         | 63.5       | 2   | 0.055   | 18.1       | 0.95M  |
| Ours(DSNet) | no       | coarse     | 69.3       | 2   | 0.027   | 36.5       | 0.91M  |

* We reimplement models based on open source code[41], and evaluate speed under the same setting with our DSNet, and speed may be different from original paper.

Table 3: Class-wise evaluation results on Cityscapes test set. Results are from original papers.

| Method   | Driving Importance Level 1 | Driving Importance Level 2 | Driving Importance Level 3 |
|----------|----------------------------|----------------------------|----------------------------|
| ENE       | Sky 75.0  Bui 43.4  Wal 33.2  Ter 64.1  Fen 43.4  Pol 88.6  Veg 34.1  Sid 44.0  TLi 96.3  TSi 38.4  Roa 65.5  Per 90.6  Car 36.9  Tru 43.1  Bus 50.5  Tra 38.8  Mot 55.4  Bic  |
| ERFNet    | 94.2  89.8  42.5  68.2  48.0  56.3  91.4  81.0  59.8  65.3  97.7  57.1  76.8  92.8  50.8  60.1  51.8  47.3  61.7  |
| DSNet     | 92.1  89.8  50.3  57.7  53.5  57.1  98.9  77.2  45.9  70.6  96.6  55.9  73.6  90.8  64.4  71.6  61.1  50.3  69.2  |

Very few training images into one batch, otherwise resizing images is adopted, such as light-weight models ENE and ERFNet. Resizing images will inevitably lose precious spatial information in fine annotation images, thus we determine to abandon it. 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.

4.2.1 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 [42] to implement our experiment which is a high-level training interface built upon tensorflow, and the tensorpack version is 0.8.9.

4.2.2 Data augmentation

Data augmentation is vital, as deep neural networks usually requires huge amount of data for training. We decide to include coarse annotation set, and enlarge fine annotation set by augmentation. Coarse annotation set is used to warm up our network for further fine-tuning on fine annotation set. We adopt cropping strategy which is widely adopted and proven beneficial in [29, 15] 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. As the high-resolution training images of Cityscapes dataset are so large that we can only fit very few training images into one batch, otherwise resizing images is adopted, such as light-weight models ENE and ERFNet. Resizing images will inevitably lose precious spatial information in fine annotation images, thus we determine to abandon it. 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.
pre-training on coarse annotation set alone leads to 4.4 points improvement in mean IoU.

4.3 Cityscapes Dataset Evaluation
We show our evaluation results on Cityscapes dataset. Our DSNet achieves 71.8% mean IoU on validation set and 69.3% on test set.

4.3.1 mean IoU
We list comprehensive metrics and results of our DSNet and other methods in mean IoU, inference time and number of parameters on Cityscapes test set, shown in Table. 2. We can see that our tiny model without any base model or extra training data such as ImageNet could achieve 69.3% mean IoU, which is one of the state-of-the-art results among light-weight semantic segmentation methods. This result is much higher in accuracy than light-weight semantic segmentation models such as ESPNet and SQ, and even higher than some large-scale classical models Dilation10, FCN-8s and DeepLab V1. To be specific, our DSNet has 148 times fewer number of parameters than Dilation10, but 2.1 points higher in accuracy. Our result is very close to ICNet which heavily compress the model and can not ensure proportional performance to lower resolution image. And in inference time, DSNet is among one of the fastest models. Note that we list speed of other semantic segmentation models published on-line only for reference, as this metric varies in different settings. This result demonstrates that DSNet achieves great balance between accuracy and speed.

4.3.2 Class-wise IoU
To verify our Driving Importance-weighted Loss, we show class-wise IoU in Table. 3 where we compare our model with ENet and ERFNet on every trainable classes in Cityscapes test set, and the results of ENet and ERFNet are from original papers. ERFNet employ the same weighting scheme as ENet and has similar accuracy as our DSNet, it could be a fair comparison to verify our Driving Importance-weighted Loss. Our DSNet has very different distribution difference with ERFNet. In Driving Importance Level 3, which is the most important with driving task, DSNet has significant higher accuracy than ERFNet, especially on classes such as Train, Bus and Truck. While in Driving Importance Level 2 and 3, ERFNet outnumbers ours but the gap is small. Overall speaking, DSNet and ERF both have 9 classes leading in accuracy. The class-wise evaluation result shows...
that our Driving Importance-weighted Loss could significantly improve classes related to driving tasks.

4.3.3 Visual Comparison

To intuitively understand the results of DSNet, we select some images from validation set, and compare the prediction results by our DSNet and ENetPlus, shown in Fig. 3, it is better viewed in color. We choose ENetPlus as baseline which has higher accuracy than ENet and similar number of parameters with DSNet to highlight our design choice in Hybrid Dilated Convolution, Driving Importance-weighted Loss and Context Encoding Loss, as we can see adding more parameters in ENet could not solve some specific issues. Generally, our DSNet has finer quality than ENetPlus. Especially, it has much less misclassified on vehicles category which is in Driving Importance Level 3, shown in the first, third and fourth row. In the third row, we can see that ENetPlus has misclassified in large objects, this is due to context information missing where our context encoding loss works. It is worth noting that context encoding loss leads to 0.8% improvement on mean IoU. In the last row, we show "gridding" issue which is common in ENetPlus prediction results. Our Hybrid Dilated Convolution strategy is here to solve this issue.

4.3.4 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 example, evaluation may be carried out on server with different CPU, GPU or memory card types, implementation may use different deep learning libraries which may have different computation efficiency on the same operation. Extensive engineering and optimization to specific platform would also improve speed by a large margin. Hence, we can not simply compare the numbers.

For research purpose and fair comparison, we reimplement ENet and a new model ENetPlus using tensorflow based on open source code [41], and evaluate speed of ENet, ENetPlus and our model under the same setting. ENetPlus is built by adding 4 more stages than original ENet, making its parameters comparable to our DSNet. ENetPlus is trained using the same training protocol as ENet, and achieves mean IoU is 63.5% on validation set. We load variables necessary for inference and drop all the other variables in saved checkpoint files, and we 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. 4. 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. With similar number of parameters, DSNet is about 2 times faster than ENetPlus at every scale, moreover our DSNet is much more accurate. This result strongly demonstrate the efficiency and power of our DSNet.

| Methods       | NVIDIA 1080Ti |
|---------------|---------------|
|               |   640 × 360   | 1280 × 720 | 1920 × 1080 |
| ENet*         | 11.1         | 89.7       | 57.8       | 17.3       | 126.6     | 7.9       |
| ENetPlus*     | 19.7         | 50.7       | 99.0       | 10.1       | 208.3     | 4.8       |
| DSNet(Ours)   | 9.9          | 100.5      | 49.8       | 20.1       | 102.0     | 9.8       |

5 Conclusion

In this paper, we propose a very efficient deep neural network termed as Driving Segmentation Network (DSNet). DSNet achieves state-of-the-art trade-off between accuracy and inference speed, moreover it highlights most critical objects in driving task. It achieves 71.8% mIoU on validation set, and 69.3% mIoU on test set which is 18.5% more accuracy than ENet. Class-wise IoU shows that DSNet is much more accurate than ENetPlus on high driving importance classes. Speed experiment shows that DSNet is about 1.1 times faster than ENet.

Acknowledgment

This work is supported by National Natural Science Foundation of China (Grant No. 61751209).

References

[1] Benjamin Ranft and Christoph Stiller. The role of machine vision for intelligent vehicles. IEEE Trans. Intelligent Vehicles, 1(1):8–19, 2016. doi: 10.1109/TIV.2016.2551553.

[2] Hongyuan Zhu, Fanman Meng, Jianfei Cai, and Shijian Lu. Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation. J. Visual Communication and Image Representation, 34:12–27, 2016. doi: 10.1016/j.jvcir.2015.10.012. URL https://doi.org/10.1016/j.jvcir.2015.10.012.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII, pages 833–851. Springer International Publishing, 2018. doi: 10.1007/978-3-030-01234-2_49. URL https://doi.org/10.1007/978-3-030-01234-2_49.

[4] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid Scene Parsing Network. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6230–6239. IEEE, jul 2017. doi: 10.1109/CVPR.2017.660.

[5] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. RefineNet: Multi-path Refinement Networks for High-Resolution Semantic Segmentation. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 5168–5177. IEEE, jul 2017. doi: 10.1109/cvpr.2017.549.

[6] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation. CoRR,
abs/1606.02147, 2016. URL http://arxiv.org/abs/1606.02147.

[7] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda Shapiro, and Hamnane Hajishirzi. ESPNet: Efficient Spatial Pyramid of Dilated Convolutions for Semantic Segmentation. arXiv preprint arXiv:1803.06815, 2018.

[8] Vladimir Nekrasov, Chunhua Shen, and Ian D. Reid. Light-Weight RefineNet for Real-Time Semantic Segmentation. In British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018, page 125, 2018. URL http://bmvc2018.org/contents/papers/0494.pdf.

[9] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III, pages 418–434. Springer International Publishing, 2018. doi: 10.1007/978-3-030-01219-9_25. URL https://doi.org/10.1007/978-3-030-01219-9_25.

[10] Eduardo Romera, Jose M. Alvarez, Luis Miguel Bergasa, and Roberto Arroyo. ERFNet: Efficient Residual FactorizedConvNet for Real-Time Semantic Segmentation. IEEE Trans. Intelligent Transportation Systems, 19(1):263–272, 2018. doi: 10.1109/TITS.2017.2750080. URL https://doi.org/10.1109/TITS.2017.2750080.

[11] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012, pages 3354–3361. IEEE Computer Society, 2012. doi: 10.1109/CVPR.2012.6248074.

[13] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. Pattern Recognition Letters, 30(2):88–97, 2009. doi: 10.1016/j.patrec.2008.04.005.

[14] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV, pages 122–138, 2018. doi: 10.1007/978-3-030-01264-9_8. URL https://doi.org/10.1007/978-3-030-01264-9_8.

[15] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison W. Cottrell. Understanding Convolution for Semantic Segmentation. In 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018, pages 1451–1460, 2018. doi: 10.1109/WACV.2018.00163. URL https://doi.org/10.1109/WACV.2018.00163.

[16] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context Encoding for Semantic Segmentation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[17] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. IEEE Trans. Pattern Anal. Mach. Intell., 39(4):640–651, 2017. doi: 10.1109/TPAMI.2016.2572683. URL https://doi.org/10.1109/TPAMI.2016.2572683.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA, pages 248–255. IEEE Computer Society, 2009. doi: 10.1109/CVPRW.2009.5206848.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States., pages 1106–1114, 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-networks.pdf.

[20] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014. URL http://arxiv.org/abs/1409.1556.

[21] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, pages 1–9. IEEE Computer Society, 2015. doi: 10.1109/CVPR.2015.7298594.

[22] Mark Everingham, S. M. Ali Eslami, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 111(1):98–136, 2015. doi: 10.1007/s11263-014-0733-5.

[23] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell., 39(12):2481–2495, 2017. doi: 10.1109/TPAMI.2016.2644615.

[24] Fisher Yu and Vladlen Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. In ICLR, 2016.

[25] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jos Garca Rodriguez. A Review on Deep Learning Techniques Applied to Semantic Segmentation. CoRR, abs/1704.06857, 2017. URL http://arxiv.org/abs/1704.06857.

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Seg-
mentation. In Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings. Part III, pages 234–241, 2015. doi: 10.1007/978-3-319-24574-4_28. URL https://doi.org/10.1007/978-3-319-24574-4_28.

[27] Hyunwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015, pages 1520–1528. IEEE Computer Society, 2015. doi: 10.1109/ICCV.2015.178.

[28] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell., 40(4):834–848, 2018. doi: 10.1109/TPAMI.2017.2699184. URL https://doi.org/10.1109/TPAMI.2017.2699184.

[29] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. CoRR, abs/1706.05587, 2017. URL http://arxiv.org/abs/1706.05587.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In David J. Fleet, Thomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III, volume 8691 of Lecture Notes in Computer Science, pages 346–361. Springer, 2014. doi: 10.1007/978-3-319-10578-9_23. URL https://doi.org/10.1007/978-3-319-10578-9_23.

[31] Michael Treml, JoséArjona-Medina, Thomas Unterthiner, Rupesh Turgesh, Felix Friedmann, Peter Schubert, Andreas Mayr, Martin Heusel, Markus Hofmarcher, Michael Widrich, and others. Speeding up semantic segmentation for autonomous driving. In MLITS, NIPS Workshop, 2016.

[32] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017. URL http://arxiv.org/abs/1704.04861.

[33] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. CoRR, abs/1707.01083, 2017. URL http://arxiv.org/abs/1707.01083.

[34] Tobias Pohlen, Alexander Hermans, Markus Mathias, and Bastian Leibe. Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 3309–3318, 2017. doi: 10.1109/CVPR.2017.353. URL https://doi.org/10.1109/CVPR.2017.353.

[35] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 2999–3007. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.324.

[36] Bi-ke Chen, Chen Gong, and Jian Yang. Importance-Aware Semantic Segmentation for Autonomous Driving System. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, pages 1504–1510, 2017. doi: 10.24963/ijcai.2017/208. URL https://doi.org/10.24963/ijcai.2017/208.

[37] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[38] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng.TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

[39] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 2261–2269. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.243.

[40] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, pages 448–456. JMLR.org, 2015. URL http://jmlr.org/proceedings/papers/v37/ioffe15.html.

[41] Kwot Sin. Tensorflow-enet. https://github.com/kwotsin/TensorFlow-ENet, 2017.

[42] Yuxin Wu et al. Tensorpack. https://github.com/tensorpack/, 2016.