ISPRS Journal of Photogrammetry and Remote Sensing 178 (2021) 124–134

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Real-time Semantic Segmentation with Context Aggregation Network

Michael Ying Yang *, Saumya Kumaar, Ye Lyu, Francesco Nex

Faculty of Geo-Information Science and Earth Observation (ITC), University of Twente, the Netherlands

ARTICLE INFO

Keywords:
Semantic segmentation
Real-time
Convolutional neural network
Context aggregation network

ABSTRACT

With the increasing demand of autonomous systems, pixelwise semantic segmentation for visual scene understanding needs to be not only accurate but also efficient for potential real-time applications. In this paper, we propose Context Aggregation Network, a dual branch convolutional neural network, with significantly lower computational costs as compared to the state-of-the-art, while maintaining a competitive prediction accuracy. Building upon the existing dual branch architectures for high-speed semantic segmentation, we design a high resolution branch for effective spatial detailing and a context branch with light-weight versions of global aggregation and local distribution blocks, potent to capture both long-range and local contextual dependencies required for accurate semantic segmentation, with low computational overheads. We evaluate our method on two semantic segmentation datasets, namely Cityscapes dataset and UAVid dataset. For Cityscapes test set, our model achieves state-of-the-art results with mIOU of 75.9%, at 76 FPS on an NVIDIA RTX 2080Ti and 8 FPS on a Jetson Xavier NX. With regards to UAVid dataset, our proposed network achieves mIOU score of 63.5% with high execution speed (15 FPS).

1. Introduction

The last decade has witnessed the rapid development of scene understanding in the field of computer vision and photogrammetry, especially the fundamental semantic segmentation task. Semantic segmentation is to assign labels for each pixel in the image, which has extensive applications, including scene understanding (Cong et al., 2020; Yang et al., 2017), autonomous vehicles and driver assistance (Cordts et al., 2016; Geiger et al., 2012), augmented reality for wearables. With the advance of deep learning (Krizhevsky et al., 2012), convolutional neural network (CNNs) are becoming the preferred approaches for semantic segmentation in recent years, e.g. Fully Convolutional Network (FCN) (Long et al., 2015), Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2017). These approaches produce highly accurate segmentation results, but often at the cost of reduced computational efficiency. For many applications, real-time performance is in fact often necessary, since semantic labeling is usually employed only as a preprocessing step of other time-critical tasks (Poudel et al., 2019).

Low-latency semantic segmentation becomes a challenging task as the optimal balance between accuracy and efficiency, i.e. computational complexity, memory footprint and execution speed, is hard to achieve. Conventional real-time semantic segmentation architectures usually address only one of the above perspectives, thereby making high-accuracy designs computationally expensive and high-speed models relatively inaccurate. These high-speed models tend to have a relatively lower prediction accuracy, e.g. Poudel et al. (2018, 2019), whereas the more accurate models tend to have lower inference speeds and higher computational overheads, e.g. Orsic et al. (2019), Yu et al. (2018). There is a significant gap between the high-speed and high-accuracy architectures, in terms of computational expenses and execution speeds (See Table 1).

There are several challenges that are very commonly associated with real-time segmentation designs. Firstly, high-accuracy designs like Orsic et al. (2019), Yu et al. (2018) rely heavily on dense feature extractors such as ResNet-18 (He et al., 2016). Secondly, the shallow extractors utilized in the relatively high-speed algorithms such as Poudel et al. (2018, 2019), provide for lower computational expenses but are unable to extract sufficient features for accurate segmentation. Thirdly, even though the computationally expensive models are accurate, they suffer from some local and global inconsistencies during inference. These aforementioned inconsistencies are usually not found in non-real time methods like Li et al. (2019), Zhu et al. (2019), but while designing low-latency architectures, the trade-offs are sometimes unfavourable. In these regards, inspired by popular dual-branch architectures, we propose a novel framework called Context Aggregation Network, where we design two branches, one for fast and effective spatial detailing and the

* Corresponding author.
E-mail address: michael.yang@utwente.nl (M.Y. Yang).

https://doi.org/10.1016/j.isprsjprs.2021.06.006
Received 15 April 2021; Received in revised form 30 May 2021; Accepted 9 June 2021
Available online 19 June 2021

Table 1
Comparison with state-of-the-art on the Cityscapes dataset (Cordts et al., 2016). For all the network models, mIOU are taken directly from the original publications. FPS indicates the average model run-times on a single RTX 2080Ti with an input resolution of 2048 × 1024. We recompute the MAdd (Multiply–add operations) count and Flops (floating point operations) count on the full resolution image from the official implementations, except GAS (Lin et al., 2020). Note that FPS* indicates the average model run-times on a single Jetson Xavier NX with an input resolution of 2048 × 1024 (see Section 4.1.6).

Model	mIOU		Memory	MAdd	Flops	Params	FPS	FPS*
	val	test						
CGNet Wu et al. (2018)	–	64.8	3134.91 MB	55.01G	27.05G	0.5 M	34.91	2.91
ContextNet Poudel et al. (2018)	–	66.1	1429.43 MB	13.98G	6.74G	0.88 M	118.65	10.49
SINet Park et al. (2020)	69.4	68.2	**672.00 MB**	**2.99G**	**1.24G**	**0.12 M**	68.61	**12.02**
Fast-SCNN Poudel et al. (2019)	–	68.4	1239.33 MB	13.85G	6.72G	1.14 M	**128.97**	11.49
DFANet Li et al. (2019)	71.3	70.1	1778.09 MB	30.68G	15.28G	2.19 M	47.88	4.71
LedNet Wang et al. (2019)	71.5	70.6	3031.75 MB	90.71G	45.84G	0.93 M	24.72	0.7
ESNet Lyu et al. (2019)	–	70.7	1176.29 MB	66.81G	33.81G	1.81 M	55.65	4.65
ShelfNet Zhuang et al. (2019)	75.2	74.8	1158.12 MB	187.37G	93.69G	14.6 M	44.37	2.59
SwiftNet Orsic et al. (2019)	75.4	75.5	1671.66 MB	207.64G	103.37G	11.80 M	45.40	2.61
BiSeNet Yu et al. (2018)	74.8	74.7	1941.39 MB	208.18G	103.72G	12.89 M	65.50	2.42
GAS Lin et al. (2020)	72.4	71.8	–	–	–	–	108.40	–
Ours	**76.6**	**75.9**	1256.18 MB	24.37G	12.03G	2.64 M	76.50	8.21

other for dense context embedding. We further address the issue of local and global inconsistencies by reformulating global aggregation and local distribution (GALD) blocks (Li et al., 2019) for real-time applications. Our speed-accuracy trade-offs and effective spatial-contextual feature fusion allow us to outperform the previous state-of-the-art approaches for real-time semantic segmentation on two public datasets, namely Cityscapes dataset (Cordts et al., 2016) and UAVid dataset (Lyu et al., 2020). For Cityscapes dataset, we also evaluate our approach on an embedded device. Codes and training models will be made publicly available.

The remainder of the paper is structured as follows. In Section 2, we review related work in both accurate and real-time semantic segmentation, with focus on recent deep learning methods. Section 3 presents the architecture of our Context Aggregation Network. In Section 4, we show our results on the Cityscapes dataset (Cordts et al., 2016) and compare our model against other state-of-the-art models. Extensive ablation experiments are carried out on the Cityscapes dataset (Cordts et al., 2016) to evaluate our proposed method. We also test our model with the UAVid dataset (Lyu et al., 2020). Section 5 concludes this paper.

2. Related work

Semantic segmentation is one of the fundamental problems of computer vision and photogrammetry. A common approach before 2012 is to use pixel-based classifiers, such as random forests (Schroff et al., 2008), and pixel-based conditional random fields (CRFs) (Lafferty et al., 2001) to improve performance by modelling neighbourhood dependencies. On the other side, CNNs are driving advances in computer vision in recent years, such as image classification (Krizhevsky et al., 2012), detection (Zhang et al., 2014), recognition (Agrawal et al., 2014; Oquab et al., 2014), semantic segmentation (Long et al., 2015; Jafari et al., 2017), pose estimation (Toshev et al., 2014; Krull et al., 2015). A number of popular approaches, including Recurrent CNN (Pinheiro and Collobert, 2014) and FCN (Long et al., 2015), have shown a significant boost in accuracy by adapting state-of-the-art CNN based image classifiers to the semantic segmentation problem. Pinheiro and Collobert (2014) presented a feed-forward approach for scene labelling based on Recurrent CNN. The system is trained in an end-to-end manner over raw pixels, and models complex spatial dependencies with low inference cost. Long et al. (2015) has addressed the coarse-graining effect of the CNN by upsampling the feature maps in deconvolution layers, and combining fine-grained and coarse-grained features during prediction. Chen et al. (2015) proposed a framework to minimize coarse-graining by skipping multiple sub-sampling layers and avoid introducing additional parameters by using sparse convolutional kernels in the layers with large receptive fields, with a dense CRF (Krähenbühl and Koltun, 2011) as post-processing step. Zheng et al. (2015) has addressed coarse-graining by expressing a mean-field CRF with Gaussian pairwise potentials as a Recurrent CNN, and concatenating this Recurrent CNN behind a FCN, for end-to-end training of all parameters.

Real-time semantic segmentation has been addressed using diverse approaches. Romera et al. (2017) proposed to use factorized convolutions with residual connections for maintaining a balance between accuracy and execution speed. Poudel et al. (2018) suggested a dual-branch network with bottlenecks to effectively capture local and global context for fast segmentation. An improved learning-to-downsample module was proposed in Poudel et al. (2019) for improved trade-offs between execution speed and accuracy. Accurate dual-branch segmentation networks were suggested by Yu et al. (2018), where novel feature fusion and attention refinement modules for accurate semantic segmentation tasks were proposed. Multiple encoder-decoder pairs with multi-scale skip connections were also studied in this regard in Zhuang et al. (2019). This ensemble of shallow and deep paths is viewed as a shelf of multiple networks allows for effective feature representation with shallower backbones like ResNet-34, as compared to Yu et al. (2018), Zhao et al. (2017). Another approach to real-time segmentation is by using depth-wise asymmetric bottlenecks (Li et al., 2019), which theoretically provides for a sufficient receptive field as well as captures dense context.

Attention modules have the capability to model long-range dependencies, and several authors have employed the concept of attention in various works (Lin et al., 2016, 2021; Lin et al., 2017; Shen et al., 2018; Vaswani et al., 2017). The introduction of attention to machine understanding was achieved first in Lin et al. (2017), where the global dependencies of inputs were learnt, which were then applied to natural language processing. Since then, a lot of works have utilized this concept for several scene understanding tasks at both single and multiple scales (Fu et al., 2019; Li et al., 2019; Zhong et al., 2020; Zhu et al., 2019; Huang et al., 2019; Ramachandran et al., 2019; Tao et al., 2020), thereby outperforming the previous conventional context embedding methodologies. Another context-focused work was published by Jiang et al. (2020) where they introduced context refinement and context integration modules for efficient scene segmentation. Light-weight feature pyramid encoding models were suggested in Liu and Yin (2019), which is an adaptation of the regular encoder-decoder architecture with depth-wise dilated convolutions. Multi-scale context aggregation was presented in Si et al. (2020), Zhang and Zhang (2020), Kumaar et al. (2021). Si et al. (2020) adopted class boundary supervision to process certain relevant boundary information. Zhang and Zhang (2020) proposed an optimized cascaded factorized atrous spatial pyramid pooling (Chen et al., 2017) module. Kumaar et al. (2021) proposed context aggregated bilateral networks to balance the trade-offs between accuracy and execution speed. Orsic et al. (2019) developed an

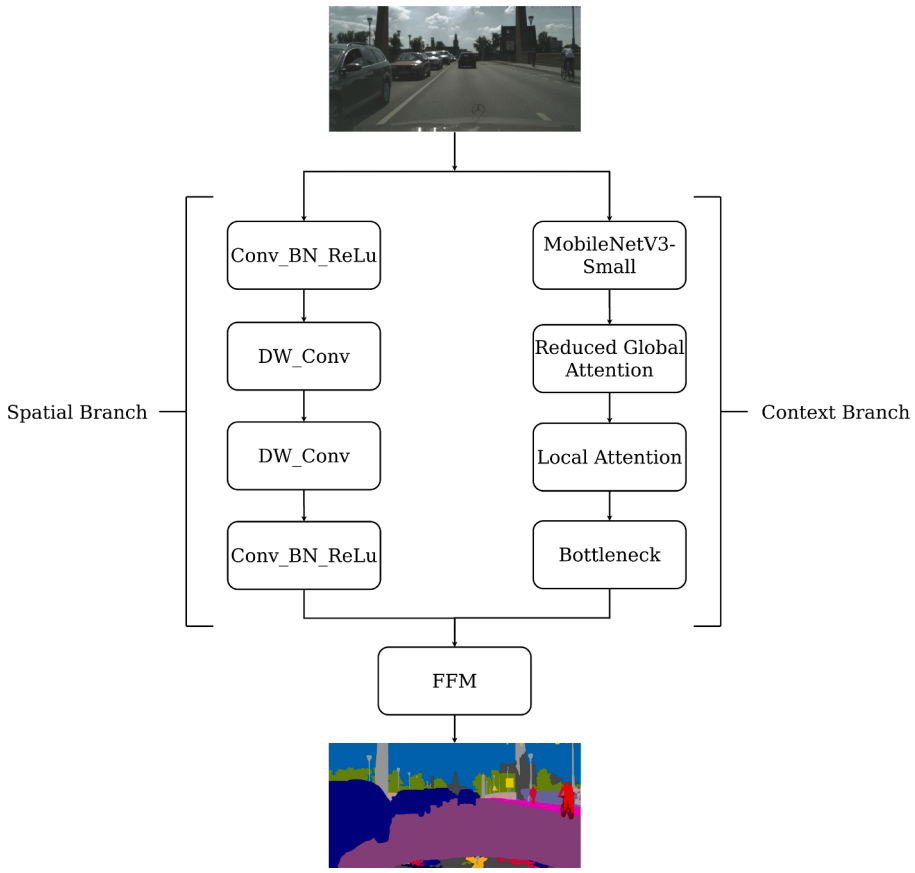

Fig. 1. Overall model architecture of Context Aggregation Network. The spatial and context branches allow for multi-scale feature extraction with significantly low computations. Fusion block (FFM) assists in feature normalization and selection for optimal scene segmentation. The bottleneck in the context branch allows for a deep supervision into the representational learning of the attention blocks.

approach which exploits light-weight upsampling and lateral connections with a residual network as the main recognition engine for real-time scene understanding. This particular algorithm is deemed as the current state-of-the-art network for real-time semantic segmentation on Cityscapes dataset.

3. Method

We illustrate the architecture of our Context Aggregation Network in Fig. 1 with two branches, one for fast and effective spatial detailing and the other for dense context embedding. The spatial and context branches allow for multi-scale feature extraction with significantly low computations. These two branches are then fused in the fusion block (FFM) for the final object category prediction. Each component of our model is described in details in the following sections.

3.1. Spatial Branch

In order to encode sufficient spatial information, multiple existing approaches (Chen et al., 2017; Peng et al., 2017; Wang et al., 2018; Chen et al., 2017) have employed the usage of dilated convolutions, while others attempt to capture large receptive fields with either pyramid pooling or large-sized kernels (Peng et al., 2017; Zhao et al., 2017; Chen et al., 2017). These approaches indicate that sufficient receptive fields and effective spatial information encoding, could be crucial for accurate semantic segmentation. It is, however, difficult to satisfy both the requirements in parallel, especially when designing real-time segmentation architectures.

Conventional real-time designs usually either downsize the image to a smaller resolution (Zhao et al., 2018) or use a lightweight reduction model (Badrinarayanan et al., 2017; Paszke et al., 2016) for speeding up the overall architecture. Downsizing the image, however, incurs a loss in the spatial information and light-weight reduction models like Xception (Chollet, 2017) tend to damage the receptive fields because of the incessant channel pruning, especially in the early stages (Badrinarayanan et al., 2017; Paszke et al., 2016). This problem was addressed in Yu et al. (2018), but at the cost of significant increase in the computations, thereby imparting a lower execution speed on mobile and embedded platforms. Based upon these observations, we propose a shallow branch that encodes rich spatial information and maintains an adequate receptive field, while maintaining a significantly low computational cost from a full-resolution image. Specifically, this path has four convolutional layers, where the first layer has a large kernel size, followed by batch normalization and ReLU, followed by two depth-wise convolutional layers. A strategic use of depth-wise convolutions results in the same outcomes as that of conventional convolutions, but with reduced computations, and the marginal loss in features can be compensated by enlarging the number of feature representations. Finally, the last layer is another convolutional layer with kernel size of one. Strides for the first three layers are fixed at two, whereas the last layer has a unit stride. This branch (Spatial Branch in Fig. 1), hence generates an output that is one eighth of the input resolution (Yu et al., 2018), thereby maintaining the required spatial information with a significant reduction in computations.

3.2. Context Branch

Detailed spatial information coupled with adequate receptive field significantly affects semantic segmentation accuracy (Yu et al., 2018). While the spatial branch in Section 3.1 takes care of the spatial details,

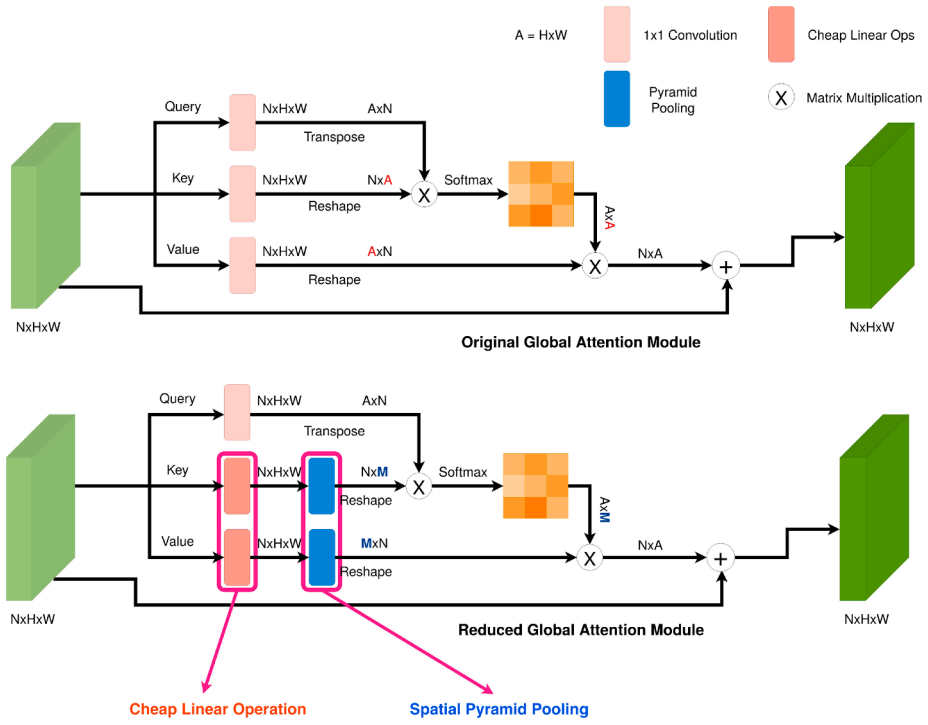

Fig. 2. Original global attention module (Top) Li et al. (2019) and reduced global attention module (Bottom). Our reduced GA module leverages the benefits of spatial pyramid pooling (SPP) Zhao et al. (2017) and cheap linear operations (CLO) Han et al. (2020), whereas SPP module allows for a strategic selection of represen. tative features.

we design a new attention-based context branch, with light-weight global aggregation (Li et al., 2019) and local attention (Li et al., 2019) blocks, for providing a sufficient receptive field and capturing both global and local context. We use a pre-trained MobileNetV3-Small (Howard et al., 2019) as the lightweight feature extractor in this branch, which can downsample the input images effectively and efficiently, to provide rich high level semantic features. These features are unrefined and need to be passed on to a refinement stage, termed as the context aggregation block comprising of reduced global attention (Li et al., 2019) and local attention sub-modules.

3.2.1. *MobileNetV3-Small*

MobileNetV3-Small (Howard et al., 2019) employs a mixture of layers suggested in MobileNetV2 (Sandler et al., 2018) and MnasNet (Tan et al., 2019), to construct the most effective and efficient neural network for mobile applications. Modified swish non-linear functions were used to improve the performance of layers, along-with hard sigmoid for squeeze-and-excitation modules. The network is fully convolutional and supports arbitrary input sizes. We use a pre-trained MobileNetV3-Small (Howard et al., 2019) as the lightweight feature extractor, which can downsample the input images and provide the semantic features.

3.2.2. *Context Aggregation Block*

The long-range and local dependencies in the representational outputs of feature extractors are crucial for accurate semantic segmentation (Li et al., 2019; Wang et al., 2018; Zhu et al., 2019). The proposed context aggregation block is to capture such inter-channel and intra-channel mappings, effectively and efficiently. Several previous works have suggested modules to effectively acquire such semantics (Fu et al., 2019; X. Li et al., 2019; Y. Li et al., 2019; Zhu et al., 2019). For our work, we adopt the global attention (GA) block from global aggregation and local distribution (GALD) (Li et al., 2019). This module is potent enough to capture long-range dependencies crucial for accurate semantic segmentation but is computationally expensive and requires significant GPU memory for execution.

Reduced Global Attention Block. Fig. 2 Top shows the flow of the original global attention (GA) module (Li et al., 2019), where the output from the previous backbone stage is fed to three parallel convolutional layers to generate new embeddings. After the embeddings have been generated similarity matrices are calculated using matrix multiplications, followed by a Softmax normalization process. The output contains the semantic cues for every position in the input feature vector. There are two limitations to this GA module (Li et al., 2019) for real-time semantic segmentation. Firstly, the original design proposes to extract the contextual information directly from the outputs of the backbone using three parallel convolution layers, thereby increasing the required number of parameters. Secondly, the matrix multiplications of the Key and Value convolutions followed the by the next multiplication process after the softmax activation stage, increase the time complexity, as these computations are performed on relatively large matrices (Zhu et al., 2019).

The matrix multiplications are large because of the size of the input feature vector, A, and if it were changed to a smaller value M, (where $M \ll A$), it might help in alleviating some of the computations. Although, changes have to be made in such a way that the output size of the vector remains unchanged. Hence, we employ spatial pyramid pooling (SPP) modules (Zhao et al., 2017) in the global attention module to effectively reduce the size of the feature vectors, as shown in Fig. 2 Bottom. Instead of feeding all the spatial points to the multiplication process, it becomes more feasible to sample the points and feed only certain representative points to the process. Following Zhu et al. (2019), we use four adaptive maximum pooling at four scales and the pooling results are flattened and concatenated to serve as the input to the next layer. For our experiments, the number of sparse representations can be formulated as $M = \sum_{n \in 1,3,5,8} n^2 = 110$, thereby reducing the complexity to $\mathcal{O}(\hat{N}AM)$, which is much lower than $\mathcal{O}(\hat{N}A^2)$. For example, for the input to the GA block of $64 \times 32 = 2048$, this asymmetric multiplication saves $\frac{64 \times 32}{110} \approx 18$ times the computation cost. Furthermore, the feature statistics captured

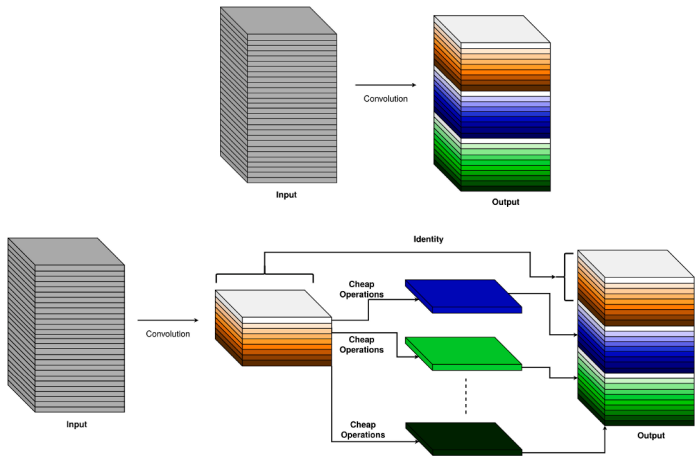

Fig. 3. Cheap linear operations (CLO) Han et al. (2020). Top: conventional 1×1 convolution layer. Bottom: CLO use multiple kernel sizes within a sin.gle convolution.

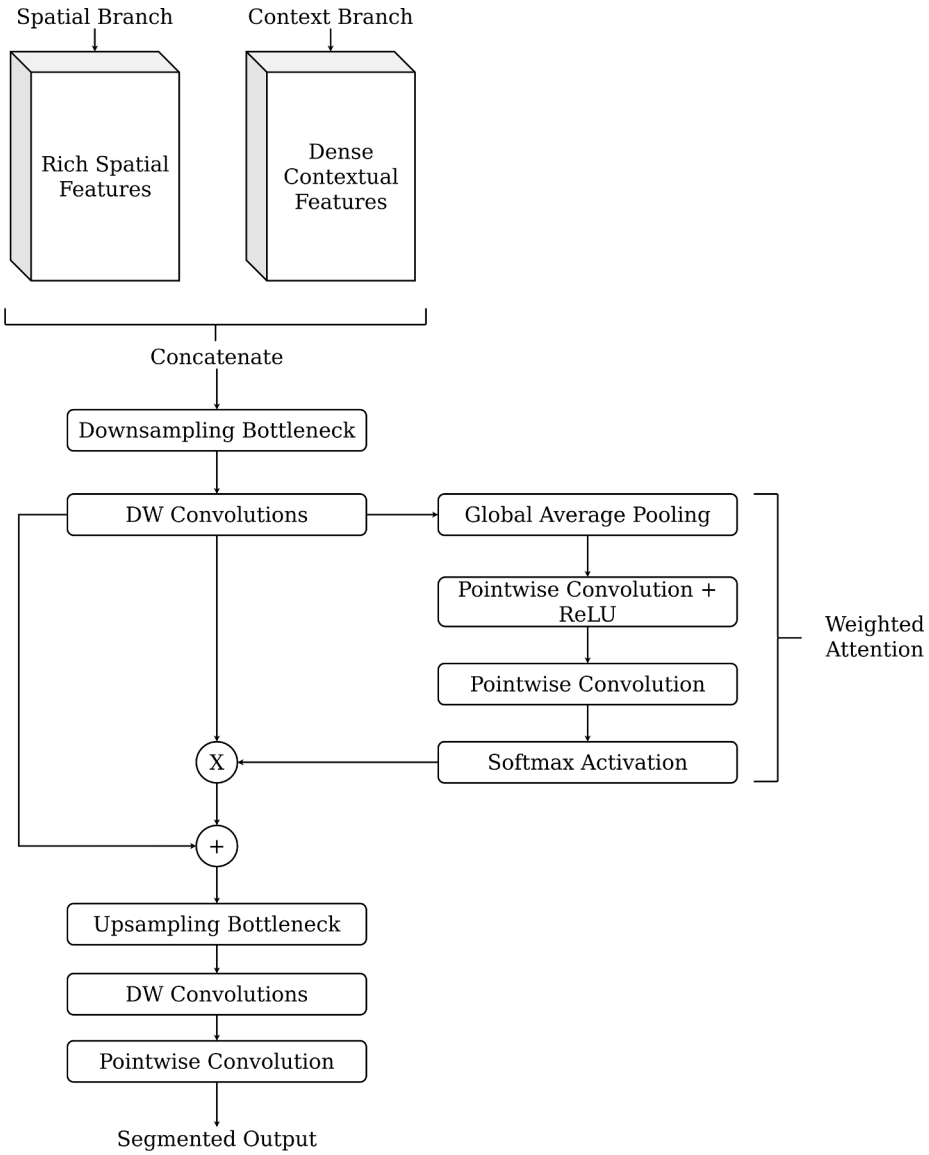

Fig. 4. Feature fusion module (FFM).

by the pooling module are sufficient to provide cues about the global scene semantics.

Secondly, in the original GA module (Fig. 2 Top), three parallel 1×1 convolution layers are used, which results in a relatively larger number of parameters. This may not have a direct influence on the overall execution speed, but a neural design with lesser parameters demonstrates the efficiency of the model. Regular convolution layers have multiple learnable filters that convolve on the input feature vector. It was suggested in Han et al. (2020) that these regular convolution layers can be replaced with a concept called cheap linear operations (CLO), which is graphically depicted in Fig. 3. The key idea is to begin with a smaller convolution, and then perform a set of linear transformations on the output of the smaller convolution to generate new representation. Afterwards, we simply stack the representations from both the stages to create a set of features that corresponds to a full convolution operation. These linear transformations significantly reduce the parameters and computations (Han et al., 2020) as compared to regular convolutions.

Local Attention. The global statistics for every group in the GA module are later multiplied back to features within. However, the windows in which the statistics are calculated are relatively large, and hence there is a possibility that the statistical cues could be biased to towards the larger patterns as there are more samples within, which can further cause over-smoothing of the smaller patterns.

In this regard, a local attention (LA) module was proposed in Li et al. (2019) to adaptively use the features, considering patterns at every position encoded by the previous global attention block. LA module is adapted directly after the GA block as a fine-tuning stage for the global semantic cues. Our ablation studies indicate that this module is efficient and fast. The LA block predicts local weights by re-calculating the spatial extent, which is primarily targeted to avoid coarse feature representation issues present in the previous GA module. Here, the predicted local weights add a point-wise trade-off between the global information and local context. Therefore, the local attention block is modelled as a set of three depth-wise convolutional layers, which allows for fine-tuning the feature representations from the previous GA module.

3.2.3. Bottleneck

Inspired from previous works (He et al., 2016; Howard et al., 2019), we design a simple downsampling module to restrict the representation of the refined features in the depth dimension. Furthermore, this restriction, when adapted directly post the context aggregation stage allows us to supervise the representational learning of the attention blocks and context branch.

3.3. Feature Fusion Module (FFM)

It is to be noted that the features extracted from both the branches are at different scales of representation and require a scale normalization for effective fusion. Hence, a simple addition of both features (Poudel et al., 2018, 2019) to save computations, is unlikely to produce desirable accuracy. Therefore in this work, we implement a feature fusion technique as suggested in Yu et al. (2018), with certain adaptations. In order to fully utilize the vector representations from both the branches, we concatenate both the features first, followed by a downsampling bottleneck. After the concatenation, the final feature representation has large dimensions, which increases the amount of required computations. Adding a downsampling bottleneck reduces these computations in the later stages of feature selection (weighted attention) by a significant margin, without causing damage to the overall accuracy (see Table 6). The weighted attention inspired from Yu et al. (2018), Hu et al. (2018) is added to selectively weigh the features in terms of their contribution to the overall prediction accuracy. These selected features are later upsampled to generate the same number of representations as in Yu et al. (2018), but with significant reduction in computations. The final two layers after the upsampling bottleneck generate the final output predictions. We use only two layers in this case because for a simple class-wise separation, multiple layers become unnecessary, hence one depth-wise separable convolution and one point-wise convolution are sufficient. A detailed schematic is shown in Fig. 4.

3.4. Loss Functions

For training the Context Aggregation Network, we use three cross entropy loss functions with online hard example mining (Shrivastava et al., 2016), one (primary) for the final output and two (auxiliary) for the context branch. The auxiliary loss functions allow for a deep supervision of the learning of the context branch and attention modules. The overall joint loss representation of our model $L(X; W)$ can be formulated as:

$$L(X; W) = l_p(X; W) + l_{c1}(X_1; W) + l_{c2}(X_2; W) \tag{1}$$

where, l_p is the principal loss for monitoring the overall output, l_{c1} is the auxiliary loss for the reduced global attention module of the network, l_{c2} is the auxiliary loss for the local attention module, X is the final feature output, X_1 is the feature output from the reduced global attention module, X_2 is the feature output from the local attention module, W are the network parameters, and p is the final output of the network prediction. Utilizing a joint loss makes it easier to optimize the model, as suggested in Yu et al. (2018), Zhu et al. (2019). Note that, we only use the two auxiliary loss functions in the training phase.

4. Experiments

In this section, experiments are shown to evaluate the effectiveness of the proposed Context Aggregation Network on two publicly available datasets. We compare the performance of our model against that of other the state-of-the-art models on the Cityscapes dataset (Cordts et al., 2016). We also conduct a comprehensive ablation study on the Cityscapes dataset to show the effectiveness of our proposed method and evaluate how network structure influence the model performance. Next the UAVid dataset (Lyu et al., 2020) is used to further demonstrate the advantages of our method.

4.1. Experiments on Cityscapes dataset

4.1.1. Dataset

We benchmark our proposed approach on the Cityscapes dataset [1]. Cityscapes is an urban scene understanding dataset. The dataset contains a total of 5000 images (fine-grained) out of which, 2975 are for training, 500 for validation and the remaining 1525 for testing. The dataset also contains additional 20 K coarsely annotated images, but we do not use them in this work. The image size for this dataset is 2048×1024, collected from across 50 different cities. The pixel-level annotations consist of layered polygons and have been realized in-house to guarantee highest quality levels. Annotation and quality control take more than 1.5 h on average for a single image. The densely annotated data contains 35 classes, out of which 19 are used for the semantic segmentation, i.e., road, sidewalk, building, wall, fence, pole, traffic light, traffic sign, vegetation, terrain, sky, person, rider, car, truck, bus, train, motorcycle, and bicycle.

4.1.2. Training Setting

For optimizing the network, we use Stochastic Gradient Descent (SGD) (Kiefer and Wolfowitz, 1952) and set the initial learning rate as e^{-4} for Cityscapes. We employ the poly-learning rate strategy, where during training, the learning rate is multiplied with $1 - \left(\frac{iter}{max\_iter}\right)^{power}$, with power being equal to 0.9. For Cityscapes image, we randomly crop

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

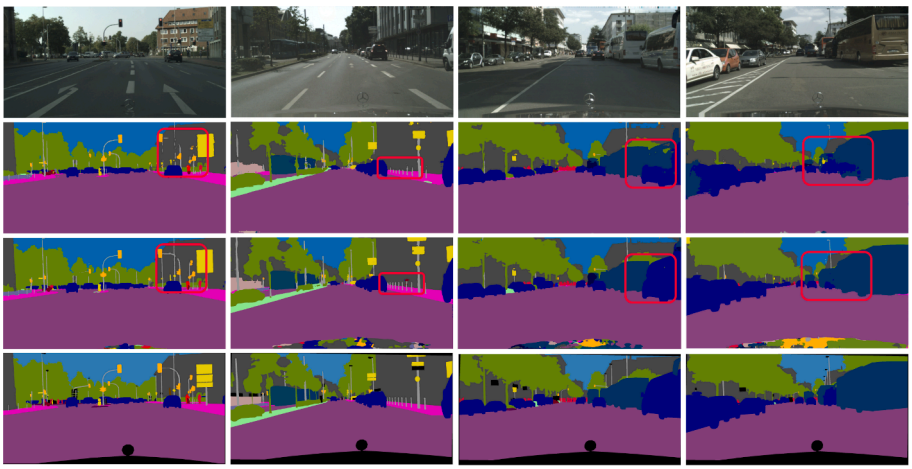

Fig. 5. Semantic segmentation results on the Cityscapes validation set. First row consists of the input RGB images; the second row shows the prediction results of SwiftNet (Orsic et al., 2019); the third row shows the predictions from our model and the red boxes show the regions of improvements, and the last row comprises of the ground truth of t.he input images.

patches of [1024, 1024] from the original input images during training. We use data augmentation techniques like random horizontal flips, random scaling and color jitter. Scales range from (0.75, 1.0, 1.5, 1.75, 2.0). The batch size is set at six for Cityscapes and training iteration is set at 160 K. All the experiments are conducted on a single NVIDIA RTX 2080Ti and Jetson Xavier NX, with PyTorch.

4.1.3. Evaluation metrics

For evaluation, we use the standard mean of class-wise intersection over union (mIOU), memory footprint (in MB), MAdd (Multiply–add operations) count, Flops count (floating point operations), and the overall execution speed (Frames per second). We take the mean over the IOU (Everingham et al., 2015) of each predefined class $IOU = TP/(TP + FP + FN)$, where TP, FP, and FN are the numbers of true positive, false positive, and false negative pixels, respectively. For calculating the computational expenses we adopt the same strategies as mentioned in Orsic et al. (2019), Yu et al. (2018).

4.1.4. Comparison with state-of-the-art

A detailed comparison between our Context Aggregation Network and other architectures has been provided in Table 1, based upon the GPU memory footprint, MAdd/Flops count, execution speed (RTX 2080Ti) and the overall mIOU score on the Cityscapes validation and test sets. As it can be observed from the table, our model outperforms the previous methods for real-time scene understanding and achieves the highest mIOU scores of 76.6% and 75.9% on validation and test sets respectively. In comparison with the most memory efficient model SINet (Park et al., 2020), our model has 7.7% higher mIOU and 7.8 FPS faster speed. In comparison with fastest model Fast-SCNN (Poudel et al., 2019), our model has 7.5% higher mIOU. In comparison with the most recent work GAS (Lin et al., 2020), our model has 4.1% higher mIOU, while being competitive regarding the speed assuming full resolution image input. Note that the FPS number of GAS was computed on a smaller image input of 1537 $\times$ 769.

Furthermore, comparing our proposed method with the previously established state-of-the-art algorithms (Orsic et al., 2019; Poudel et al., 2019; Yu et al., 2018), our improvements favour both accuracy and speed simultaneously. Computational overheads, such as parameter count, Flops etc., in our architecture are significantly lower than the existing accurate real-time architectures, with increased accuracy. Optimized GALD-blocks coupled with efficient spatial detail and light-weight dense extractors, allow our approach to outperform the conventional real-time semantic segmentation architectures in multiple aspects.

Qualitative results are shown in Fig. 5. Compared with Orsic et al. (2019), our model has better performance in terms of detecting under-represented objects like poles, traffic signs, etc. Due to the efficient global and local semantic aggregation, our model does not suffer from such local or global inconsistencies. More qualitative results on Cityscapes validation set are shown in Fig. 6. Qualitative results on Cityscapes test set are shown in Fig. 7.

4.1.5. Ablation Studies

Baseline is defined as a simple dual-branch network with two convolution layers in the spatial branch and untrained feature extractor in the second branch. The baseline is devoid of attention and bottleneck modules and is similar in structure with Poudel et al. (2018). For fusing the features from both branches, we simply add them which are later discriminated by a small classifier block into the respective number of classes. Both branches are fed images at the same resolution, unlike

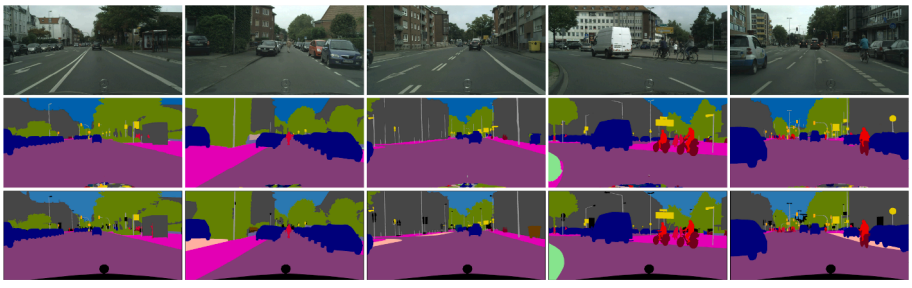

Fig. 6. More semantic segmentation results on the Cityscapes validation set. First row consists of the input RGB images. Second row contains the predictions from our model and the third row shows the ground truth of the input images.

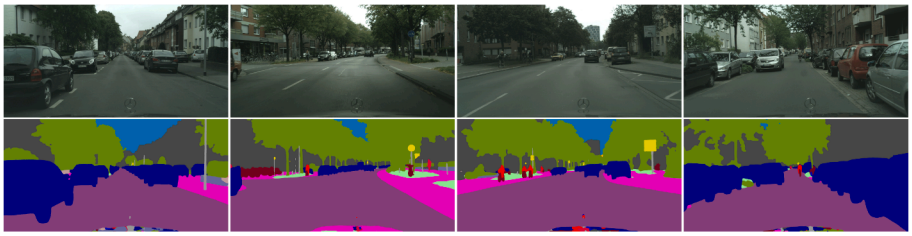

Fig. 7. Semantic segmentation results on the Cityscapes test set. Top row shows the images, and the bottom row shows the predictions.

Table 2
Basic ablation study. SB and CB stand for spatial and context branches, whereas FFM stands for feature fusion module respectively.

Model	mIOU	FPS
Baseline	68.4	110.65
Baseline + SB + CB	72.3	86.76
Baseline + SB + CB + CAB	74.7	81.20
Baseline + SB + CB + CAB + FFM	76.6	76.50

Table 3
CAB implemented in other algorithms. Straightforward addition to Poudel et al. (2018, 2019) results in significant improvements over the baseline models. In Yu et al. (2018), the attention refinement modules were replaced with CAB.

Model	mIOU w/o CAB	mIOU w CAB
ContextNet Poudel et al. (2018)	66.1	69.2
Fast-SCNN Poudel et al. (2019)	68.4	71.2
BiSeNet Yu et al. (2018)	74.7	75.3

Table 4
Comparative study of different attention modules. Flops, Params and Runtime correspond to the attention modules and not the overall architecture.

Module	Flops	Params	Runtime	mIOU
BiSeNet Yu et al. (2018)	3.63G	311 K	**3.24 ms**	74.8
DANet Fu et al. (2019)	1.01G	82.24 K	17.62 ms	76.3
GALDNet Li et al. (2019)	1.01G	65.34 K	14.28 ms	76.1
ANNNet Zhu et al. (2019)	0.82G	42.24 K	8.35 ms	76.4
GALD + SPP + CLO	**0.024**	**12.29 K**	3.48 ms	**76.6**

Poudel et al. (2018) and all the ablation experiments are performed on this baseline. In order to clarify how different modules contribute to the performance, we ablate different components and present the results in Table 2.

4.1.5.1. Context Aggregation Block. The context aggregation block (CAB) is designed specifically to capture local and global context effectively and efficiently. If we remove CAB from the design keeping all other modules and training/inference parameters intact, we observe a drop of 2.1% in the overall mIOU score, along with a drop in inference time by almost 3 ms. This implies that the addition of the context block enhances the feature representations, while having minimal impact on the overall execution speed and complexity. Table 3 further proves the efficacy of the context aggregation block, which can be used as a plug&play module with other dual-branch architectures for semantic segmentation.

Table 5
Complexity comparison between our approach and the current state-of-the-art with different backbones. R18 and MV3 stand for ResNet-18 and MobileNetV3-Small (1. ×) respectively.

Model	mIOU	Flops	Params	FPS
BiseNet-R18 Yu et al. (2018)	74.8	103.72G	12.89 M	47.20
SwiftNet-R18 Orsic et al. (2019)	75.4	103.37G	11.80 M	45.40
Ours-R18	76.7	66.41G	9.19 M	54.50
Ours-MV3	76.6	12.03G	2.64 M	66.50

Table 6
Comparative study of different fusion modules. AW stands for attention weight based fusion. Addition of bottlenecks indicates the relative decrease in computations without significant impacts on mIOU.

Fusion Style	mIOU	Flops
Feature Addition Poudel et al. (2018)	73.2	0.5G
Feature Concatenation w/o AW Poudel et al. (2019)	74.5	0.8G
Feature Concatenation w AW Yu et al. (2018)	76.7	1.8G
Feature Concatenation w AW + Bottlenecks	76.6	0.9G

Table 4 shows a comparative study of different attention modules. Note that, adding SPP modules (Zhao et al., 2017) for attention modules was suggested in Zhu et al. (2019). We observe that adding cheap linear operations (CLO) (Han et al., 2020) not only reduces the required computations, but also provides a slightly better accuracy. This could be attributed to the fact within these linear transformations, there can be multiple kernel sizes (Han et al., 2020), thereby allowing for multi-scale feature aggregation.

4.1.5.2. Backbone Choice. A lot of previous real-time semantic segmentation architectures (Orsic et al., 2019; Yu et al., 2018; Zhao et al., 2018; Zhuang et al., 2019) employ powerful feature extractors like ResNet-18. Even though this choice is justified for accurate semantic segmentation, the implications on execution speed and computational complexity are profound. Hence, for effective comparison, we replace our MobileNetV3 backbone with ResNet-18 and study the outcomes (Table 5). From the table we confirm that our segmentation head is still lighter, faster and more accurate as compared to both SwiftNet and BiSeNet, even if we use an expensive feature extractor like ResNet-18. Furthermore, the comparison between Ours-R18 and Ours-MV3 from Table 5 reveals that the computational overheads added by ResNet-18 are larger as compared to MobileNetV3-Small even though they both provide similar mIOU scores.

4.1.5.3. Feature Fusion Module. Several fusion techniques have been suggested in the literature and designing the right one has significant impacts on the final outcome. Consider Table 6 for a quantitative comparison between the various fusion techniques. Feature concatenation with weighted attention and bottlenecks provides the best mIOU-Flops balance out of all the variants.

Table 7
Comparison with state-of-the-art on the Cityscapes dataset (Cordts et al., 2016), with respect to the system execution speed, memory footprints, and Flops, counting on one fourth and one sixteenth of the full resolution Cityscapes images, respectively. Note that FPS in this table indicates the average model run-times on a single Jetson Xavier NX.

Model	Input Size							
	512 × 256				1024 × 512			
	Memory	MAdd	Flops	FPS	Memory	MAdd	Flops	FPS
CGNet Wu et al. (2018)	195.96 MB	3.44G	1.69G	25.77	786.75 MB	13.75G	6.76G	11.43
ContextNet Poudel et al. (2018)	88.65 MB	869.38 M	419.61 M	60.97	356.44 MB	3.49G	1.68G	36.77
SINet Park et al. (2020)	42.00 MB	187.02 M	74.98 M	28.28	168.00 MB	784.04 M	299.90 M	24.78
Fast-SCNN Poudel et al. (2019)	77.37 MB	865.74 M	419.78 M	69.64	309.71 MB	3.46G	1.68G	42.34
DFANet Li et al. (2019)	111.16 MB	1.92G	955.16 M	15.25	444.55 MB	7.67G	3.82G	13.82
ShelfNet Zhuang et al. (2019)	72.39 MB	11.74G	5.86G	26.97	289.53 MB	46.97G	23.42G	7.30
SwiftNet Orsic et al. (2019)	104.66 MB	12.99G	6.47G	26.19	418.06 MB	51.95G	25.89G	9.14
BiseNet Yu et al. (2018)	121.35 MB	13.01G	6.48G	28.07	418.36 MB	52.05G	25.93G	8.84
Ours	61.23 MB	1.03G	502.22 M	45.55	244.82 MB	4.10G	2.01G	35.72

Table 8
Quantitative comparisons between other methods and our model on the UAVid test set (Lyu et al., 2020). The first eight columns show IOU scores for eight classes. The last two columns list mean IOU score and the execution speed. The boldface text means the highest value in the column.

Model	Building	Tree	Clutter	Road	Vegetation	Static Car	Moving Car	Human	mIOU	FPS
MSD Lyu et al. (2020)	79.8	74.5	57.0	**74.0**	55.9	32.1	**62.9**	19.7	57.0	1.00
Fast-SCNN Poudel et al. (2019)	75.7	71.5	44.2	61.6	43.4	19.5	51.6	0.0	45.9	**33.84**
ShelfNet Zhuang et al. (2019)	76.9	73.2	44.1	61.4	43.4	21.0	52.6	3.6	47.0	9.65
SwiftNet Orsic et al. (2019)	85.3	78.2	64.1	61.5	76.4	62.1	51.1	15.7	61.1	11.84
BiSeNet Yu et al. (2018)	85.7	78.3	64.7	61.1	77.3	63.4	48.6	17.5	61.5	11.08
Ours	**86.6**	**79.3**	**66.0**	62.1	**78.1**	**68.3**	47.8	**19.9**	**63.5**	15.14

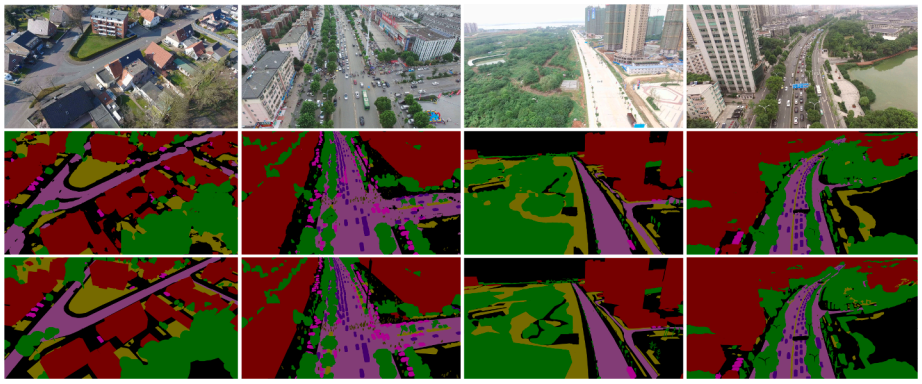

Fig. 8. Semantic segmentation results on the UAVid (Lyu et al., 2020) validation set. First row consists of the input RGB images. Second row contains the predictions from our model and the third row shows the ground truth of t.he input images.

4.1.6. Results on Embedded Device

Inference on full scale GPUs (Titan X or RTX20 series) is unlikely to provide a real-world analysis, as autonomous vehicles, UAVs and UGVs are more likely to have low-power consumption modules with limited memory. In order to create effective comparisons, we further benchmark recent real-time semantic segmentation models on an embedded device Jetson Xavier NX, a small form factor system-on-module. The device has a 384-core NVIDIA Volta GPU with 48 Tensor Cores and a 6-core NVIDIA Carmel ARM 64-bit CPU. Results on the full resolution Cityscapes images are shown in the last column of Table 1. Our model has the highest mIOU, and meantime remains relatively fast at 8.21 FPS. Compared with Fast-SCNN (Poudel et al., 2019), our method decreases at the speed of 3.28 FPS, but gains 7.7% mIOU on the Cityscapes test set.

We further experiment with our algorithm and others on multiple resolutions to demonstrate the efficiency. Table 7 provides the system execution speed, memory footprints and MAdd/Flops, counting on one fourth and one sixteenth of the full resolution Cityscapes images (2048 × 1024), respectively. Note that, for full resolution image inference (2048 × 1024) in Table 1, we employ the maximum power mode (15 W, all six cores). However, for the smaller resolution, we use a lower mode (10 W, only four cores) because implementing semantic segmentation on lower resolutions is likely to imply that there could be more processes running, and hence considering the usage of other cores for other threads, we utilize only four. The execution speed is affected by the number of processors involved in computations.

4.2. Experiments on UAVid dataset

4.2.1. Dataset

We also evaluate our Context Aggregation Network by another semantic segmentation dataset UAVid (Lyu et al., 2020). UAVid dataset is a high-resolution UAV semantic segmentation dataset focusing on street scenes. The dataset contains a total of 420 images out of which 200 are for training, 70 for validation and the remaining 150 for testing. The original images have been resized to the resolution of 3840 × 2160. UAVid is a challenging benchmark due to the large resolution of images, large scale variation, and complexities in the scenes. For UAVid dataset, 8 classes are selected for the semantic segmentation, i.e., building, road, tree, low vegetation, static car, moving car, human, and clutter. Densely labeled fine annotations are acquired with the authors' own labeler tool

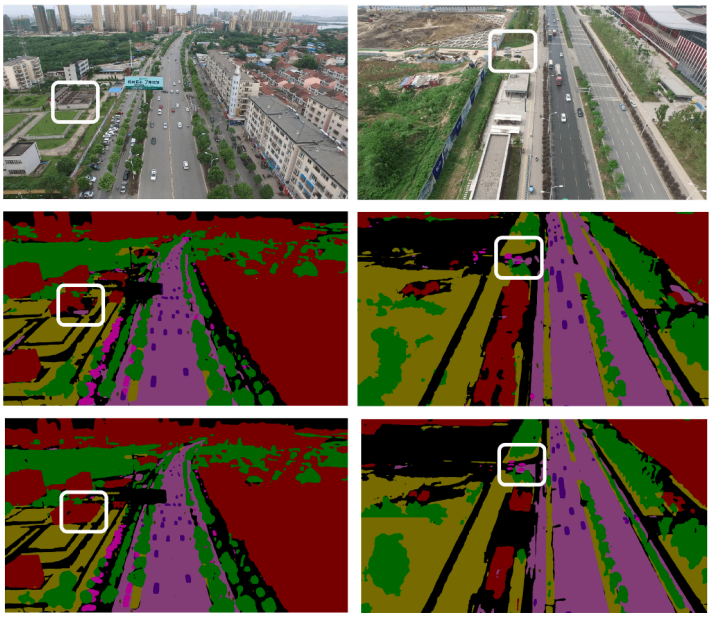

Fig. 9. Comparative segmentation results from the UAVid (Lyu et al., 2020) test dataset. First row shows the input RGB images, second row depicts the outputs of MSD (Lyu et al., 2020) and the third row shows the predictions of our model. White boxes highlight the regions. of improvement.

(Lyu et al., 2020) with pixel level, super-pixel level, and polygon level annotation methods. It takes approximately 2 h to label all pixels in one image. We use the same hyperparameters and data augmentation as those for experiments on Cityscapes dataset, except the initial learning rate as $5e^{-5}$, batch size as three, and training iterations as 240 K. We split the image into four equal quarters of 1920 $\times$ 1080 during training.

4.2.2. Results

Table 8 provides a detailed quantitative results of our model and the previous state-of-the-art method - MSD (Lyu et al., 2020). Our model outperforms MSD by a large margin of 6.5%, while maintaining an execution speed of 15 FPS with the full resolution image on a single GPU RTX 2080Ti. Furthermore, we also train several recent real-time semantic segmentation models on UAVid dataset and report the results on UAVid test set from the official server [2] in Table 8. However, we do not benchmark with this dataset on the Jetson Xavier NX, since it is not powerful enough. Our model achieves the best IOU score on six out of eight classes, and the best mean IOU with 2% gain over BiSeNet (Yu et al., 2018) while being 4 FPS faster. Although Fast-SCNN (Poudel et al., 2019) is 18.7 FPS faster than ours, their mIOU is worse by a large margin of 17.6%.

Qualitative results on UAVid validation set and test set are shown in Figs. 8 and 9, respectively. As it can be seen from Fig. 9, compared with MSD, our model does not suffer from local or global inconsistencies, thereby effectively capturing the cues to scene semantics.

5. Conclusions

In this paper, we have developed a light-weight Context Aggregation Network to address the challenge of real-time semantic segmentation with improved inference speeds and reduced computational expenses. Building upon the dual-branch architectures for high-speed semantic segmentation, we design a high resolution branch for effective spatial detailing and a context branch with light-weight versions of global aggregation and local distribution blocks, potent to capture both long-range and local contextual dependencies required for accurate semantic segmentation, with low computational overheads. Our proposed approach is trained end-to-end. We verify the advantages of our proposed method by comprehensive ablation experiments on Cityscapes dataset. When comparing to other state-of-the-art models, our model achieves 76.6% and 75.9% mIOU on Cityscapes validation and test sets respectively, at 76 FPS on an NVIDIA RTX 2080Ti and 8 FPS on a Jetson Xavier NX. Furthermore, we conduct experiments on the UAVid dataset and our model produces more accurate pixelwise semantic predictions in terms of mIOU (63.5%) with high execution speed (15 FPS), compared to the baseline and other leading models. This further demonstrates the advantage of our proposed method. For future work, we will extend the current approach to address low-latency instance segmentation.

[2] https://uavid.nl/

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Agrawal, P., Girshick, R.B., Malik, J., 2014. Analyzing the performance of multilayer neural networks for object recognition. In: European Conference on Computer Vision (ECCV), pp. 329–344.

Badrinarayanan, V., Kendall, A., Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE transactions on pattern analysis and machine intelligence 39 (12), 2481–2495.

Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A., 2015. Semantic image segmentation with deep convolutional nets and fully connected crfs. In: International Conference on Learning Representations (ICLR).

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence 40 (4), 834–848.

Chen, L.-C., Papandreou, G., Schroff, F., Adam, H., 2017. Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv: 1706.05587.

Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1251–1258.

Cong, Y., Ackermann, H., Liao, W., Yang, M.Y., Rosenhahn, B., 2020. NODIS: neural ordinary differential scene understanding. In: European Conference on Computer Vision. Springer, pp. 636–653.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Everingham, M., Eslami, S.M.A., Gool, L.V., Williams, C.K.I., Winn, J.M., Zisserman, A., 2015. The pascal visual object classes challenge: A retrospective. Int. J. Comput. Vision 111 (1), 98–136.
Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., Lu, H., 2019. Dual attention network for scene segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3146–3154.
Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition.
Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C., 2020. Ghostnet: More features from cheap operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1580–1589.
He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.
Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al., 2019. Searching for mobilenetv3. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1314–1324.
Huang, L., Yuan, Y., Guo, J., Zhang, C., Chen, X., Wang, J., 2019. Interlaced sparse self-attention for semantic segmentation. arXiv preprint arXiv: 1907.12273.
Hu, J., Shen, L., Sun, G., 2018. Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7132–7141.
Jafari, O.H., Groth, O., Kirillov, A., Yang, M.Y., Rother, C., 2017. Analyzing modular CNN architectures for joint depth prediction and semantic segmentation. In: IEEE International Conference on Robotics and Automation. ICRA, pp. 4620–4627.
Jiang, B., Tu, W., Yang, C., Yuan, J., 2020. Context-integrated and feature-refined network for lightweight object parsing. IEEE Trans. Image Process. 29, 5079–5093.
Kiefer, J., Wolfowitz, J., 1952. Stochastic estimation of the maximum of a regression function. Ann. Math. Stat. 23 (3), 462–466.
Krähenbühl, P., Koltun, V., 2011. Efficient inference in fully connected crfs with gaussian edge potentials. In: Neural Information Processing Systems (NIPS), pp. 109–117.
Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Neural Information Processing Systems (NIPS), pp. 1097–1105.
Krull, A., Brachmann, E., Michel, F., Yang, M.Y., Gumhold, S., Rother, C., 2015. Learning analysis-by-synthesis for 6d pose estimation in RGB-D images. In: IEEE International Conference on Computer Vision. ICCV, pp. 954–962.
Lafferty, J., McCallum, A., Pereira, F., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning (ICML), pp. 282–289.
Li, X., Zhang, L., You, A., Yang, M., Yang, K., Tong, Y., 2019. Global aggregation then local distribution in fully convolutional networks. In: British Machine Vision Conference.
Li, G., Yun, I., Kim, J., Kim, J., 2019. Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. In: British Machine Vision Conference.
Li, Y., Chen, X., Zhu, Z., Xie, L., Huang, G., Du, D., Wang, X., 2019. Attention-guided unified network for panoptic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7026–7035.
Li, H., Xiong, P., Fan, H., Sun, J., 2019. Dfanet: Deep feature aggregation for real-time semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9522–9531.
Lin, G., Shen, C., Van Den Hengel, A., Reid, I., 2016. Efficient piecewise training of deep structured models for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3194–3203.
Kumaar, S., Lyu, Y., Nex, F., Yang, M.Y., 2021. Cabinet: efficient context aggregation network for low-latency semantic segmentation. In: IEEE International Conference on Robotics and Automation (ICRA).
Lin, Z., Feng, M., Santos, C.N.d., Yu, M., Xiang, B., Zhou, B., Bengio, Y., 2017. A structured self-attentive sentence embedding. arXiv preprint arXiv: 1703.03130.
Lin, P., Sun, P., Cheng, G., Xie, S., Li, X., Shi, J., 2020. Graph-guided architecture search for real-time semantic segmentation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4203–4212.
Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2021. Local and global encoder network for semantic segmentation of airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 176, 151–168.
Liu, M., Yin, H., 2019. Feature pyramid encoding network for real-time semantic segmentation. In: British Machine Vision Conference, BMVC, p. 260.
Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation, in. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
Lyu, H., Fu, H., Hu, X., Liu, L., 2019. Esnet: Edge-based segmentation network for real-time semantic segmentation in traffic scenes. In: 2019 IEEE International Conference on Image Processing (ICIP), pp. 1855–1859.
Lyu, Y., Vosselman, G., Xia, G.-S., Yilmaz, A., Yang, M.Y., 2020. Uavid: A semantic segmentation dataset for uav imagery. ISPRS Journal of Photogrammetry and Remote Sensing 165, 108–119.
Oquab, M., Bottou, L., Laptev, I., Sivic, J., 2014. Learning and transferring mid-level image representations using convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1717–1724.
Orsic, M., Kreso, I., Bevandic, P., Segvic, S., 2019. In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 12607–12616.
Park, H., Sjosund, L., Yoo, Y., Monet, N., Bang, J., Kwak, N., 2020. Sinet: Extreme lightweight portrait segmentation networks with spatial squeeze module and information blocking decoder. In: The IEEE Winter Conference on Applications of Computer Vision, pp. 2066–2074.
Paszke, A., Chaurasia, A., Kim, S., Culurciello, E., 2016. Enet: a deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv: 1606.02147.
Peng, C., Zhang, X., Yu, G., Luo, G., Sun, J., 2017. Large kernel matters–improve semantic segmentation by global convolutional network, in. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4353–4361.
Pinheiro, P., Collobert, R., 2014. Recurrent convolutional neural networks for scene labeling. In: International Conference on Machine Learning (ICML).
Poudel, R.P., Bonde, U., Liwicki, S., Zach, C., 2018. Contextnet: Exploring context and detail for semantic segmentation in real-time. In: British Machine Vision Conference.
Poudel, R.P., Liwicki, S., Cipolla, R., 2019. Fast-scnn: fast semantic segmentation network. In: British Machine Vision Conference.
Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., Shlens, J., 2019. Stand-alone self-attention in vision models. arXiv preprint arXiv: 1906.05909.
Romera, E., Alvarez, J.M., Bergasa, L.M., Arroyo, R., 2017. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Trans. Intell. Transp. Syst. 19 (1), 263–272.
Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C., 2018. Mobilenetv 2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510–4520.
Schroff, F., Criminisi, A., Zisserman, A., 2008. Object class segmentation using random forests. In: British Machine Vision Conference (BMVC).
Shen, T., Zhou, T., Long, G., Jiang, J., Pan, S., Zhang, C., 2018. Disan: directional self-attention network for rnn/cnn-free language understanding. In: Thirty-Second AAAI Conference on Artificial Intelligence.
Shrivastava, A., Gupta, A., Girshick, R., 2016. Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 761–769.
Si, H., Zhang, Z., Lu, F., 2020. Real-time semantic segmentation via multiply spatial fusion network. In: British Machine Vision Conference, BMVC.
Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., Le, Q.V., 2019. Mnasnet: Platform-aware neural architecture search for mobile, in. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2820–2828.
Tao, A., Sapra, K., Catanzaro, B., 2020. Hierarchical multi-scale attention for semantic segmentation. arXiv preprint arXiv: 2005.10821.
Toshev, A., Szegedy, C., 2014. Deeppose: Human pose estimation via deep neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1653–1660.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. In: Advances in Neural Information Processing Systems.
Wang, P., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X., Cottrell, G., 2018. Understanding convolution for semantic segmentation. In: 2018 IEEE winter conference on applications of computer vision (WACV). IEEE, pp. 1451–1460.
Wang, X., Girshick, R., Gupta, A., He, K., 2018. Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7794–7803.
Wang, Y., Zhou, Q., Liu, J., Xiong, J., Gao, G., Wu, X., Latecki, L.J., 2019. Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In: 2019 IEEE International Conference on Image Processing (ICIP). IEEE, pp. 1860–1864.
Wu, T., Tang, S., Zhang, R., Zhang, Y., 2018. Cgnet: a light-weight context guided network for semantic segmentation. arXiv preprint arXiv: 1811.08201.
Yang, M.Y., Liao, W., Ackermann, H., Rosenhahn, B., 2017. On support relations and semantic scene graphs. ISPRS Journal of Photogrammetry and Remote Sensing 131, 15–25.
Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N., 2018. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: Proceedings of the European conference on computer vision (ECCV), pp. 325–341.
Zhang, Z., Zhang, K., 2020. Farsee-net: Real-time semantic segmentation by efficient multi-scale context aggregation and feature space super-resolution. In: in: IEEE International Conference on Robotics and Automation (ICRA), pp. 8411–8417.
Zhang, N., Donahue, J., Girshick, R.B., Darrell, T., 2014. Part-based r-cnns for fine-grained category detection. In: European Conference on Computer Vision (ECCV), pp. 834–849.
Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J., 2017. Pyramid scene parsing network, in. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2881–2890.
Zhao, H., Qi, X., Shen, X., Shi, J., Jia, J., 2018. Icnet for real-time semantic segmentation on high-resolution images. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 405–420.
Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P., 2015. Conditional random fields as recurrent neural networks. In: International Conference on Computer Vision (ICCV).
Zhong, Z., Lin, Z.Q., Bidart, R., Hu, X., Daya, I.B., Li, Z., Zheng, W.-S., Li, J., Wong, A., 2020. Squeeze-and-attention networks for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13065–13074.
Zhuang, J., Yang, J., Gu, L., Dvornek, N., 2019. Shelfnet for fast semantic segmentation, in. In: Proceedings of the IEEE International Conference on Computer Vision Workshops.
Zhu, Z., Xu, M., Bai, S., Huang, T., Bai, X., 2019. Asymmetric non-local neural networks for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 593–602.
Zhu, L., Wang, T., Aksu, E., Kamarainen, J.-K., 2019. Cross-granularity attention network for semantic segmentation, in. In: Proceedings of the IEEE International Conference on Computer Vision Workshops.