CABiNet: Efficient Context Aggregation Network for Low-Latency Semantic Segmentation

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

Abstract—With the increasing demand of autonomous machines, pixel-wise semantic segmentation for visual scene understanding needs to be not only accurate but also efficient for any potential real-time applications. In this paper, we propose CABiNet (Context Aggregated Bi-lateral Network), a dual branch convolutional neural network (CNN), with significantly lower computational costs as compared to the state-of-the-art, while maintaining a competitive prediction accuracy. Building upon the existing multi-branch architectures for high-speed semantic segmentation, we design a cheap 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. Specifically, we achieve 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.

I. INTRODUCTION

Visual scene understanding has profound implications in modern robotic systems. However, autonomous machines have real-time requirements that require crucial trade-offs, especially in computationally intense designs, such as for pixel-wise image semantic segmentation. Hence, 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. [26], [27], whereas the more accurate models tend to have lower inference speeds and higher computational overheads, e.g. [23], [39]. There is a significant gap between the high-speed and high-accuracy architectures, in terms of computational expenses and execution speeds (See Table I).

There are several challenges that are very commonly associated with real-time segmentation designs. Firstly, high-accuracy designs like [23], [39] rely heavily on dense feature extractors such as ResNet-18 [8]. Secondly, the shallow extractors utilized in the relatively high-speed algorithms such as [26], [27], 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 [45], [15], but while designing low-latency architectures, the trade-offs are sometimes unfavourable. In these regards, inspired by popular dual-branch architectures, we propose 

CABiNet - Context Aggregated Bi-lateral Network, where we design two branches, one for fast and effective spatial detailing and the other for dense context embedding. We further address the issue of local and global inconsistencies by reformulating global aggregation and local distribution (GALD) blocks [15] 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 Cityscapes dataset \(^1\) with an mIOU score of 75.9\% at 76 FPS.

II. RELATED WORK

Real-time semantic segmentation has been addressed using diverse approaches. Romera et al. [29] proposed to use factorized convolutions with residual connections for maintaining a balance between accuracy and execution speed. Poudel et al. [26] suggest a dual-branch network with bottle-necks to effectively capture local and global context for fast segmentation. Later they propose an improved learning-to-dowmsample module in [27] for improved trade-offs between execution speed and accuracy. Accurate dual-branch segmentation networks were suggested by Yu et al. [39], 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 [46]. This ensemble of shallow and deep paths is viewed as a shelf of multiple networks allows for effective feature representation with

\(^1\) Codes and training models will be made publicly available. Experimental results on a challenging UA Vid semantic segmentation dataset [22] are shown in the supplementary video.
shallow backbones like ResNet-34, as compared to [39], [42]. Another approach to real-time segmentation is by using depth-wise asymmetric bottlenecks [14], 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 [17], [19], [36], [31]. The introduction of attention to machine understanding was achieved first in [19], 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 [6], [15], [28], [43], [44], [35], [11], thereby outperforming the previous conventional context embedding methodologies.

Another context-focused work was published by Jiang et al. [12] where they introduced context refinement and context integration modules for efficient scene segmentation. Light-weight feature pyramid encoding models were suggested in [20], which is an adaptation of the regular encoder-decoder architecture with depth-wise dilated convolutions. Multi-scale context aggregation was presented in yet another couple of approaches [33], [40], where [33] uses class boundary supervision to process certain relevant boundary information and [40] use optimized cascaded factorized ASPP [3] module to balance the trade-offs between accuracy and execution speed. Orsic et al. [23] developed an 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.

III. METHOD

We illustrate the architecture of CABI\textit{Net} in Fig. 2 with the 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.

A. Spatial Branch

Conventional real-time designs usually either downsize the image to a smaller resolution [41] or use a lightweight reduction model [2], [25] for speeding up the overall architecture. Downsizing the image however, inures a loss in the spatial information and light-weight reduction models like Xception [4] tend to damage the receptive fields because of the incessant channel pruning, especially in the early stages [1], [25]. This problem was addressed in [39], 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 1. Strides for the first three layers are fixed at 2, whereas the last layer has a unit stride. This branch (Fig. 3), hence generates an output that is \((\frac{1}{2})^{th}\) of the input resolution [39], thereby maintaining the required spatial information with a significant reduction in computations.

B. Context Branch

As already established previously, detailed spatial information coupled with adequate receptive field, significantly affects semantic segmentation accuracy [39]. While the shallow branch takes care of the spatial details, we design a new attention-based context branch, with light-weight global aggregation [15] and local attention [15] blocks, for providing a sufficient receptive field and capturing both global and local context. We use a pre-trained MobileNetV3-Small

**Fig. 2: CABiNet Architecture.** 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.

Authorized licensed use limited to: UNIVERSITY OF TWENTE.. Downloaded on November 25,2021 at 09:32:49 UTC from IEEE Xplore. Restrictions apply.
In our experiments, the number of sparse representations can be reduced by pooling at four scales and the pooling results are flattened before entering the next multiplication process. Following [45], we use four adaptive maximum pooling operations to select the most representative points and feed only certain representative points to the subsequent layers. Instead of feeding all the spatial points to the pyramid pooling (SPP) modules [42] in the global attention (GA) module, we can effectively alleviate some of the computations. Although, changes to the input feature vector, $F$, are performed on relatively large matrices [45], the key idea is to begin with a smaller size for execution speed, but a neural design with lesser parameters and better memory usage is desirable.

2) Context Aggregation Block. The long-range and local dependencies in the representational outputs of feature extractors are crucial for accurate semantic segmentation [45], [15], [37]. The proposed context aggregation block is to capture such inter-channel and intra-channel mappings, effectively and efficiently. Several previous works have suggested that accurate feature selection requires either accurate channel selection or accurate feature selection. For our work, we adopt the global attention (GA) block from global aggregation and local distribution (GALD) [15]. 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. 3 shows the flow of the global attention (GA) module. A careful observation of the pipeline indicates that there could be two possible limitations to the global attention module suggested in [15]. Firstly, the original design proposes to extract the contextual information directly from the outputs of the backbone using three parallel combination layers, thereby increasing the required number of parameters. Secondly, the matrix multiplications of the key and value convolutions followed by the next multiplication process after the softmax activation stage, increase the time complexity, as these computations are performed on relatively large matrices [45].

The matrix multiplications are large because of the size of the input feature vector, $A$, and if it was changed to a smaller value $M$, (where $M << A$), 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 [42] in the global attention module to effectively reduce the size of the feature vectors (Fig. 3). 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 [45], 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 $O(\hat{N}A^2)$, which is much lower than $O(NA^2)$. Specifically, for the input to the GA block of $64 \times 32 = 2048$, this asymmetric multiplication saves $64 \times 32 \times 2048 \approx 18$ times the computation cost. Furthermore, the feature statistics captured by the pooling module are sufficient to provide cues about the global scene semantics.

Next, this block employs three parallel $1 \times 1$ convolution layers, which results in a relatively larger number of parameters. This might not have a direct influence on the overall execution speed, but a neural design with lesser parameters and better memory usage is preferable. Regular convolution layers have multiple learnable filters that convolve on the input feature vector. It was suggested in [7] 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. Then to simply stack the representations from both the stages to create a set of features that correspond to a full convolution operation. These linear transformations significantly reduce the parameters and computations [7] as compared to regular convolutions.

Local Attention. Previously calculated the global statistics for every group in the global attention 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 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 [15] 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 and hence requires no additional improvements. Fundamentally, 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) Bottleneck: Inspired from previous works [9], [8], 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.

C. 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 [26], [27] to save computations, is unlikely to produce desirable accuracy. Therefore in this work, we implement a feature fusion technique as suggested in [39], 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 (Table VI). The weighted attention section inspired from [39], [10] 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 [39], 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.

D. Loss Functions

For training the model, we use three cross entropy loss functions with online hard example mining [32], 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:

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

(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, $W$ 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 [45], [39].

IV. Experiments

We benchmark our proposed approach on the Cityscapes dataset [5]. Cityscapes is an urban-scene understanding dataset which contains a total of fully annotated 5000 images out of which, 2975 are for training, 500 for validation and the remaining 1525 for testing. This dataset contains 35 classes, out of which 19 are used for urban scene understanding and the image size is $1024 \times 2048$.

A. Training Setting

For optimizing the network, we use Stochastic Gradient Descent (SGD) [13] 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 $1 - \left( \frac{t}{\text{iter}_{\text{max}} - \text{iter}} \right)^{\text{power}}$, with power being equal to 0.9. For Cityscapes image, we randomly crop patches of [1024, 1024] from the original input images during training. We use data augmentation techniques like random horizontal flips,
Fig. 5: Semantic segmentation results on the Cityscapes validation set. From left, the first column consists of the input images; the second column indicates the prediction results of SwiftNet [23]; the third column shows the predictions from our architecture and the red boxes show the regions of improvements and the last column comprises of the ground truths.

Table I: Comparison with state-of-the-art on the Cityscapes dataset. For all the network models, mIOU are taken directly from the original publications. FPS and FPS* indicate the average model run-times on a single RTX 2080Ti and Jetson Xavier NX respectively, on an input resolution of 1024×2048. We recompute the MAdd and FLOPS on the full resolution image from the official implementations, except GAS [18].

| Model               | mIOU | Memory   | MAdd  | Flops  | Params | FPS   | FPS*  |
|---------------------|------|----------|-------|--------|--------|-------|-------|
| ContextNet [26]     | –    | 66.1     | 1429.43MB | 13.98G | 6.74G  | 0.88M | 118.65 | 10.49 |
| SINet [24]          | 69.4 | 68.2     | 672.00MB | 2.99G  | 1.24G  | 0.12M | 68.61  | 12.02 |
| Fast-SCNN [27]      | –    | 68.4     | 1239.33MB | 13.85G | 6.72G  | 1.14M | 128.97 | 11.49 |
| LedNet [38]         | 71.5 | 70.6     | 3031.75MB | 90.71G | 45.84G | 0.93M | 24.72  | 0.7   |
| ESNet [21]          | –    | 70.7     | 1176.29MB | 66.81G | 33.81G | 1.81M | 55.65  | 4.65  |
| ShelfNet [46]       | 75.2 | 74.8     | 1158.12MB | 187.37G | 93.69G | 14.6M | 44.37  | 2.59  |
| SwiftNet [23]       | 75.4 | 75.5     | 1671.66MB | 207.64G | 103.37G | 11.80M | 45.40  | 2.61  |
| BiSeNet [39]        | 74.8 | 74.7     | 1941.39MB | 208.18G | 103.72G | 12.89M | 47.20  | 2.42  |
| GAS [18]            | 72.4 | 71.8     | –       | –      | –      | 108.40 | –      | –     |
| CABiNet (Ours)      | 76.6 | 75.9     | 1256.18MB | 24.37G | 12.03G | 2.64M | 76.50  | 8.21  |

Table II: 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 + CAB + FFM      | 76.6 | 76.50 |

A detailed comparison between our method and other architectures has been provided in Table I, based upon the GPU memory footprint, MAdd/GLOPs 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 [24], CABiNet has 7.7% higher mIOU and 7.8 FPS faster speed. In comparison with fastest model Fast-SCNN [27], CABiNet has 7.5% higher mIOU. In comparison with the most recent work GAS [18], our model has 4.1% higher mIOU, while being competitive regarding the speed assuming full resolution image input. Qualitative results are shown in Fig. 5. Compared with [23], our model has better performance in terms of detecting under-represented objects like poles, traffic signs, etc. Thanks to the efficient global and local semantic aggregation, our model does not suffer from such local or global inconsistencies. Furthermore, comparing our proposed method with the previously established state-of-the-art algorithms [27], [23], [39], our improvements favour both accuracy and speed simultaneously. Computational overheads such as parameter count, GFLOPs 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. More qualitative results on Cityscapes test set are shown in Fig. 6.

C. 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 [26]. 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 [26] and all the ablation experiments are performed on this baseline.

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 3ms. 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 III 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.

Interestingly, using SPP modules [42] for attention modules was suggested in [45], but adding cheap linear operations (CLO) [7] not only reduces the required computations, but also provides a slightly better accuracy (Table IV). This could be attributed to the fact within these linear transformations, there can be multiple kernel sizes [7], thereby allowing for multi-scale feature aggregation.

TABLE III: CAB implemented in other algorithms. Straightforward addition to [26], [27] results in significant improvements over the baseline models. In [39], the attention refinement modules were replaced with CAB.

| Model          | mIoU w/o CAB | mIoU w CAB |
|----------------|--------------|------------|
| ContextNet [26] | 66.1         | 69.2       |
| Fast-SCNN [27]  | 68.4         | 71.2       |
| BiSeNet [39]    | 74.7         | 75.3       |

TABLE IV: 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 [39]| 3.63G  | 311K   | 3.24 ms | 74.8  |
| DANet [6]  | 1.01G  | 82.24 K| 17.62 ms| 76.3  |
| GALDNNet [15]| 1.01G | 65.34 K| 14.28 ms| 76.1  |
| ANNNet [45]| 0.82G  | 42.24 K| 8.35 ms | 76.4  |
| GALD+SPP+CLO | 0.024 | 12.29K | 3.48 ms | 76.6  |

TABLE V: 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   | Runtime |
|------------------------|-------|--------|----------|---------|
| BiscNet-R18 [39]       | 74.8  | 103.72G| 12.89M   | 47.20   |
| SwiftNet-R18 [23]      | 75.4  | 103.37G| 11.80M   | 45.40   |
| CABiNet-R18 (Ours)     | 76.7  | 66.41G | 9.19M    | 54.30   |
| CABiNet-MV3 (Ours)     | 76.6  | 12.03G | 2.64M    | 66.50   |

TABLE VI: Comparative study of different fusion modules. AW stands for attention weight based fusion.

| Fusion Style                      | mIoU  | FLOPs  |
|-----------------------------------|-------|--------|
| Feature Addition [26]             | 75.2  | 0.5G   |
| Feature Concatenation w/o AW [27] | 74.5  | 0.8G   |
| Feature Concatenation w AW [39]   | 76.7  | 1.8G   |
| Feature Concatenation w AW + BOTTENNECKS | 76.6  | 0.9G   |

2) Backbone Choice: A lot of previous real-time semantic segmentation architectures [39], [23], [46], [41] employ powerful feature extractors like ResNet-18 [8]. 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 V). From the table we confirm that our segmentation head is still lighter, faster and more accurate as compared to both SwiftNet [23] and BiSeNet [39], even if we use an expensive feature extractor like ResNet-18. Furthermore, the comparison between CABiNet-R18 and CABiNet-MV3 from Table V reveals that the computational overheads added by ResNet-18 are larger as compared to MobileNetV3-Small even though they both provide similar mIoU scores.

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 VI for a quantitative comparison between the various fusion techniques. Feature concatenation with weighted attention and bottlenecks provides the most optimal mIoU-FLOPs balance out of all the variants.

4) 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. Hence, we further benchmark our algorithm and others on Jetson Xavier NX, a small form factor system-on-module, on the full resolution Cityscapes images. Results are shown in the last column of Table I.

V. CONCLUSIONS

In this paper, we have developed a light-weight approach to address the challenge of real-time semantic segmentation with improved inference speeds and reduced computational expenses. Our proposed approach is end-to-end trainable on Cityscapes, and computes an accurate prediction within 13 ms. For future work, we will extend the current approach to address real-time panoptic segmentation.
REFERENCES

[1] Vijay Badrinarayanan, Ankur Handa, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. arXiv preprint arXiv:1505.07293, 2015.

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

[3] 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 transactions on pattern analysis and machine intelligence, 40(4):834–848, 2017.

[4] François Chollet. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251–1258, 2017.

[5] 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.

[6] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3146–3154, 2019.

[7] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1580–1589, 2020.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. corr abs/1512.03385 (2015), 2015.

[9] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In Proceedings of the IEEE International Conference on Computer Vision, pages 1314–1324, 2019.

[10] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7132–7141, 2018.

[11] Lang Huang, Yuhui Yuan, Jianyuan Guo, Chao Zhang, Xilin Chen, and Jingdong Wang. Interlaced sparse self-attention for semantic segmentation. arXiv preprint arXiv:1907.12273, 2019.

[12] Bin Jiang, Wenxuan Tu, Chao Yang, and Junsong Yuan. Context-interactive and feature-refined network for lightweight urban scene parsing. arXiv preprint arXiv:1907.11474, 2019.

[13] Jack Kiefer, Jacob Wolfowitz, et al. Stochastic estimation of the maximum of a regression function. The Annals of Mathematical Statistics, 23(3):469–472, 1952.

[14] Gen Li, Inyoung Yun, Jonghyun Kim, and Jongkyu Kim. Dahnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. arXiv preprint arXiv:1907.11357, 2019.

[15] Xiangtai Li, Li Zhang, Ansheng You, Maoke Yang, Kuiyuan Yang, and Yunhai Tang. Global aggregation then local distribution in fully convolutional networks. arXiv preprint arXiv:1909.07229, 2019.

[16] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7026–7035, 2019.

[17] Guosheng Lin, Chunhua Shen, Anton Van Den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3194–3203, 2016.

[18] Peiwen Lin, Feng Sun, Guangliang Cheng, Sirui Xie, Xi Li, and Jianping Shi. Graph-guided architecture search for real-time semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4203–4212, 2020.

[19] Zhouran Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence encoder. arXiv preprint arXiv:1703.03130, 2017.

[20] Mengyu Liu and Hujun Yin. Feature pyramid encoding network for real-time semantic segmentation. arXiv preprint arXiv:1909.08599, 2019.

[21] Haoran Lyu, Huiyu Sun, Xiaojun Hu, and Liang Liu. Esnet: Edge-based segmentation network for real-time semantic segmentation in traffic scenes. In 2019 IEEE International Conference on Image Processing (ICIP), pages 1855–1859. IEEE, 2019.

[22] Ye Lyu, George Vosselman, Gui-Song Xia, Alper Yilmaz, and Michael Ying Yang. Uavid: A semantic segmentation dataset for uav imagery. ISPRS Journal of Photogrammetry and Remote Sensing, 165:108–119, 2020.

[23] Marin Orsic, Ivan Krenso, Petra Bevandie, and Sinisa Segvic. 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, pages 12607–12616, 2019.

[24] Hyojin Park, Lars Sjostedt, Youngjoo Yoo, Nicolas Monet, Jiwhan Bang, and Noju Kwak. Sines: Extreme lightweight portrait segmentation networks with spatial squeeze module and information blocking decoder. In The IEEE Winter Conference on Applications of Computer Vision, pages 2066–2074, 2020.

[25] Adam Paszke, Abhishek Chaurasia, Sangpill Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv:1606.02147, 2016.

[26] Budhika PK Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. Contextnet: Exploring context and detail for semantic segmentation in real-time. arXiv preprint arXiv:1805.04554, 2018.

[27] Rudra PK Poudel, Stephan Liwicki, and Roberto Cipolla. Fast-scnn: fast semantic segmentation network. arXiv preprint arXiv:1902.04502, 2019.

[28] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Iwan Bello, Anshul Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. arXiv preprint arXiv:1906.05909, 2019.

[29] Eduardo Romero, José M Alvarez, Luis M Bergasa, and Roberto Arroyo. Efinet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Transactions on Intelligent Transportation Systems, 19(1):263–272, 2017.

[30] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Ivan Hendrycks. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4510–4520, 2018.

[31] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. Disan: Directional self-attention network for mm/cnn-free language understanding. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[32] Paramvir Shivbastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 761–769, 2016.

[33] Haiyang Si, Zhiqiang Zhang, Feifan Lv, Gang Yu, and Feng Lu. Real-time semantic segmentation via multiply spatial fusion network. arXiv preprint arXiv:1911.07217, 2019.

[34] Yixing Tan, Bo Chen, Mingxing Tan, Yuqing Wang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2820–2828, 2019.

[35] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. arXiv preprint arXiv:2005.10821, 2020.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017.

[37] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7794–7803, 2018.

[38] Yu Wang, Quan Zhou, Jia Liu, Jian Xiong, Guangwei Gao, Xiaofu Wu, and Longin Jan Latecki. Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In 2019 IEEE International Conference on Image Processing (ICIP), pages 1860–1864. IEEE, 2019.

[39] Changqian Yu, Jingbo Wang, Changxin Gao, Gang Yu, and Xingjun Xiong. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In Proceedings of the European conference on computer vision (ECCV), pages 325–341, 2018.

[40] Zhanpeng Zhang and Kaipeng Zhang. Farsee-net: Real-time semantic segmentation networks with spatial squeeze module and information blocking decoder. In The IEEE Winter Conference on Applications of Computer Vision, pages 2066–2074, 2020.
segmentation by efficient multi-scale context aggregation and feature space super-resolution. \textit{arXiv preprint arXiv:2003.03913}, 2020.

[41] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In \textit{Proceedings of the European Conference on Computer Vision (ECCV)}, pages 405–420, 2018.

[42] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In \textit{Proceedings of the IEEE conference on computer vision and pattern recognition}, pages 2881–2890, 2017.

[43] Zilong Zhong, Zhong Qiu Lin, Rene Bidart, Xiaodan Hu, Ibrahim Ben Daya, Zhiheng Li, Wei-Shi Zheng, Jonathan Li, and Alexander Wong. Squeeze-and-attention networks for semantic segmentation. In \textit{Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition}, pages 13065–13074, 2020.

[44] Lingyu Zhu, Tinghuai Wang, Emre Aksu, and Joni-Kristian Kamarainen. Cross-granularity attention network for semantic segmentation. In \textit{Proceedings of the IEEE International Conference on Computer Vision Workshops}, pages 0–0, 2019.

[45] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In \textit{Proceedings of the IEEE International Conference on Computer Vision}, pages 593–602, 2019.

[46] Juntang Zhuang, Junlin Yang, Lin Gu, and Nicha Dvornek. Shelfnet for fast semantic segmentation. In \textit{Proceedings of the IEEE International Conference on Computer Vision Workshops}, pages 0–0, 2019.