ParCNetV2: Oversized Kernel with Enhanced Attention (Supplementary Matetrial)*

Ruihan Xu[1,2], Haokui Zhang[2,3,‡], Wenze Hu[2], Shiliang Zhang[1,‡], Xiaoyu Wang[2]
[1]National Key Laboratory for Multimedia Information Processing,
School of Computer Science, Peking University
[2]Intellifusion [3]Harbin Institute of Technology (Shenzhen)

A. Introduction

In this chapter, we present additional materials and results. First, we show some analysis of the model details. We present the proof that alternating the order of vertical and horizontal convolution does not affect the results of oversized convolution in Sec. B. In Sec. C, we explain how we adjust $\tilde{\alpha}$ to fit the model size close to the original FFN. We also compare ParCNetV2 framework with the ParCNetV1 to show the simplicity of our model in Sec. D.

Then, we provide additional experiments analysis. In Sec. E, we provide more detailed ablation studies of each component in ParCNetV2 and conclude three guidelines. In Sec. F, we present the experiment results on ImageNet-21K dataset to show the generalization ability on large scale dataset. In Sec. G, we evaluate the performance of ParCNetV2 in object detection and semantic segmentation tasks, comparing it to other recently proposed models across various model scales. We show how we accelerate the inference with implicit gemm algorithm in Sec. H.

Finally, we show multiple visualization examples of the proposed ParCNetV2. On the one hand, We provide the corresponding standard convolution kernel of the separated oversized convolution, as well as a more detailed study of the proposed oversized convolution in Sec. I. On the other hand, the comparison of Grad-CAM between the common convolution networks and ParCNetV2 is shown in Sec. J.

B. Proof of the Commutative Property of Oversized Convoluiton

As mentioned in the paper, to compute the output of the oversized convolution $Z_{i,j}$ at location (i, j), we use the following equations:

$$Y_{i,j} = \sum_{s=-(H-1)}^{H-1} k_s^h X_{i+s,j}, \tag{1}$$

$$Z_{i,j} = \sum_{t=-(W-1)}^{W-1} k_t^w Y_{i,j+t}. \tag{2}$$

We combine the two equations and calculate $Z_{i,j}$ with a single function:

$$\begin{aligned} Z_{i,j} &= \sum_{t=-(W-1)}^{W-1} k_t^w Y_{i,j+t} \\ &= \sum_{t=-(W-1)}^{W-1} k_t^w \sum_{s=-(H-1)}^{H-1} k_s^h X_{i+s,j+t} \\ &= \sum_{t=-(W-1)}^{W-1} \sum_{s=-(H-1)}^{H-1} k_t^w k_s^h X_{i+s,j+t}. \end{aligned}$$

Thus the separated oversized convolution can be regarded as a low-rank decomposition of a large convolution kernel ($k^h k^w$). In addition, the commutative law of summation indicates that the order of addition does not influence the result. Thus the order of vertical and horizontal convolution does not affect the results of oversized convolution.

C. Adjusting $\tilde{\alpha}$ of Channel BGU

We adjust $\tilde{\alpha}$ to fit the model size close to the original FFN. The number of parameters in the original FFN is $2\alpha C^2$, and in our FFN with BGU it is $2\tilde{\alpha}C^2 + \tilde{\alpha}C^2 = 3\tilde{\alpha}C^2$. To keep the number of parameters almost unchanged, we get $2\alpha C^2 = 3\tilde{\alpha}C^2$, thus

$$\tilde{\alpha} = 2\alpha/3. \tag{3}$$

The expanded ratio of FFN in most existing models is 4, which indicates that $\tilde{\alpha} = 8/3$. Researchers have shown

*Work was done when R. Xu was an intern at Intellifusion. ‡ denotes corresponding author.

that when the number of channels is a multiple of 32, it is beneficial for hardware optimization [12], so we choose $\tilde{\alpha} = 2.5$ to approximate the original FFN.

D. Comparison ParCNetV2 and ParCNetV1 Framework

We compare the framework of ParCNetV1 and ParCNetV2 in Fig. 1. ParCNetV1 is a complicated model with multi-branch architecture. The fusion modules are necessary to combine local features from MobileNetV2 block and ParC V1 block. While in our ParCNetV2, the whole model utilizes the same ParC V2 blocks. Our method is easy to follow, and consistent to the widely-used 4-stage framework.

E. Additional Ablation Studies

In this section, we present more detailed ablation studies. Tab. 1 highlights that both OC and BGU could enhance the performance of ConvNeXt.

Models	Params	Flops	Top-1 Acc
ConvNeXt	29M	4.5G	82.1%
ConvNeXt + OC	30M	4.8G	82.6%
ConvNeXt + S-BGU + C-BGU	29M	4.5G	82.9%

Table 1: Ablation studies on Oversized Conv and BGU.

Tab. 2 shows that the parallel structure outperforms the cascade structure. The cascade structure leads to a deeper model, which may increase the training difficulty. Additionally, the interaction between global and local features in the cascade structure may lead to interference.

Merge of local & global conv	Params	Flops	Top-1 Acc
Parallel	7.4M	1.6G	79.4%
DWConv only	7.4M	1.6G	78.9%
DWConv-Vertical 1D-Horizontal 1D	7.4M	1.6G	78.4%
Vertical 1D-DWConv-Horizontal 1D	7.4M	1.6G	77.4%
Vertical 1D-Horizontal 1D-DWConv	7.4M	1.6G	78.6%

Table 2: Ablation studies on the combination of local and global convolutions.

Merging of branches	Params	Flops	Top-1 Acc
Multiply	7.4M	1.6G	79.4%
Add	7.4M	1.6G	68.4%

Table 3: Ablation studies on the merging of branches.

Tab. 3 compares different ways of merging BGU branches. Addition loses input adaptability compared with multiplication, therefore it does not perform well.

We conclude three guidelines for building CNNs as: 1) leveraging the global effective receptive field; 2) integrating efficient attention mechanisms; 3) combining global and local features. The first two points let CNNs gain advantages of ViTs. The capability of accessing local features could make CNNs perform better in dense prediction CV tasks.

Models	Mixing Type	Param (M)	MACs (G)	Top-1 (%)
Swin-B [10]	Conv	88	15.4	85.2
ConvNeXt-B [10]	Attn	89	15.4	85.8
ParCNetV2-B [11]	Attn	56	12.5	**86.0**

Table 4: **Comparison with state-of-the-art transformer and hybrid networks on ImageNet-21K classification dataset.** Top-1 accuracy on the validation set is reported.

backbone	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
	Mask R-CNN 3× schedule					
Swin-T	46.0	68.1	50.3	41.6	65.1	44.9
ConvNeXt-T	46.2	67.9	50.8	41.7	65.0	44.9
ParCNetV2-T	**48.9**	70.3	53.9	**43.7**	67.6	47.0
	Cascade Mask R-CNN 3× schedule					
Swin-T	50.4	69.2	54.7	43.7	66.6	47.3
ConvNeXt-T	50.4	69.1	54.8	43.7	66.5	47.3
ParCNetV2-T	**52.6**	71.0	57.3	**45.6**	68.6	49.8
Swin-S	51.9	70.7	56.3	45.0	68.2	48.8
ConvNeXt-S	51.9	70.8	56.5	45.0	68.2	48.8
ParCNetV2-S	**53.4**	72.1	58.4	**46.3**	69.6	50.2
Swin-B	51.9	70.5	56.4	45.0	68.1	48.9
ConvNeXt-B	52.7	71.3	57.2	45.6	68.9	49.5
ParCNetV2-B	**54.0**	72.6	58.6	**46.7**	70.2	51.1

Table 5: Comparisons on **COCO [8] object detection and instance segmentation.** We use Mask R-CNN [6] and Cascade Mask R-CNN [2] as a basic framework. All models are pretrained on ImageNet-1K and trained on COCO for 3× iterations.

F. Experiments on large-scale dataset

We show the image classification results on the ImageNet-21K dataset in Tab. 4. We follow the experiment settings of ConvNeXt [11], which means we first pretrain ParCNetV2 on ImageNet-22K for 90 epochs and finetune on ImageNet-1K for 30 epochs. Compared with Swin Transformer[10] and ConvNeXt [11], ParCNetV2 achieved better Top-1 accuracy with fewer parameters and less computational cost, further demonstrating the generalization ability of our proposed methods.

G. Additional Experiments on Downstream Tasks

Object detection and instance segmentaion on COCO. Following previous works [10, 11], we finetune Cascade Mask R-CNN [2] on COCO dataset [8] with ParCNetV2

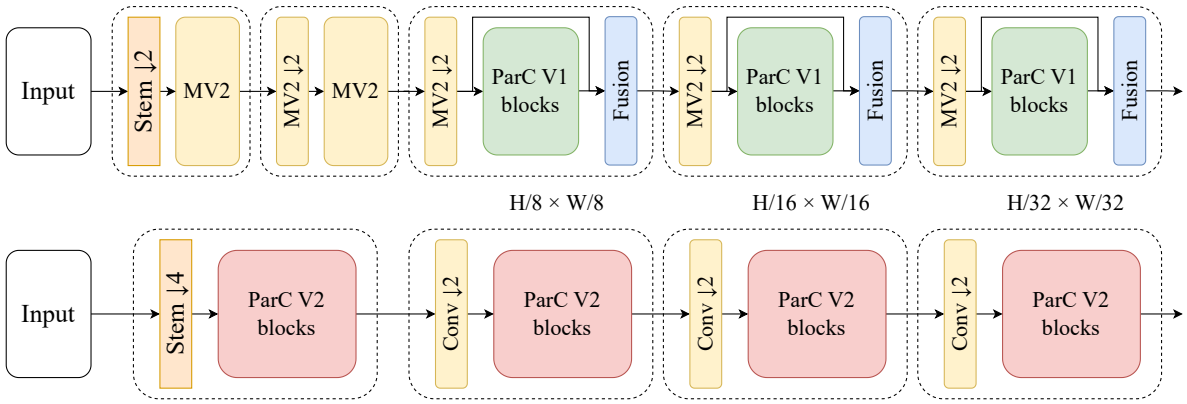


Figure 1: **Framework comparison between ParCNetV1 and ParCNetV2.** Downsampling modules with downsampling ratio 2 and 4 are represented by $\downarrow 2$ and $\downarrow 4$, respectively. **MV2**: MobileNetV2 block.

backbone	Param (M)	FLOPs (G)	mIoU ss (%)	mIoU ms (%)
Swin-T	60	945	-	45.8
ConvNeXt-T	60	939	46.0	46.7
SLaK-T	65	936	47.6	-
ParCNetV2-T	55	932	**48.5**	**49.4**
Swin-S	81	1038	-	49.5
ConvNeXt-S	82	1027	48.7	49.6
SLaK-S	91	1028	49.4	-
ParCNetV2-S	69	1005	**50.0**	**51.0**
Swin-B	121	1188	48.1	49.7
ConvNeXt-B	122	1170	49.1	49.9
RepLKNet-31B	112	1170	49.9	50.6
SLaK-B	135	1172	50.2	-
ParCNetV2-B	87	1105	**50.2**	**51.1**

Table 6: Comparisons on **ADE20K [16] semantic segmentation.** We use UperNet as a basic framework. All models are pretrained on ImageNet-1K and trained on ADE20K for 160K iterations. FLOPs are measured with the input size of (2048, 512). **ss** and **ms** indicates single-scale and multi-scale testing, respectively.

backbones. MMDetection [3] is used as the base framework. All models use pre-trained weights from ImageNet1K and are trained with $3\times$ schedule with multi-scale training. The experiment settings follow [11]. We follow all the experiment settings of ConvNeXt [11] except that the number of layers in layerwise learning rate decay [1] are adjusted to $\{7, 13, 13\}$ to fit with our model. Tab. 5 shows object detection and instance segmentation results comparing our ParCNetV2 with Swin [10] and ConvNeXt [11]. ParCNetV2 outperforms both the transformer network and convolution network by a large margin across different model complexities.

Semantic segmentation on ADE20K. ADE20K [16] is a widely-used semantic segmentation dataset, covering a broad range of 150 semantic categories. It has 25K images in total, with 20K for training, 2K for validation, and another 3K for testing. In this paper, we trained our ParCNetV2 on the training set, and report mIoUs on the validation set with both single-scale testing and multi-scale testing.

We finetune UperNet [15] in mmsegmentation as our base framework. Following Swin [10] and ConvNeXt [11] settings in training, we employ the AdamW [7] optimizer with an initial learning rate of 1×10^{-4}. We use stage-wise learning rate decay [1] as ConvNeXt. We also employ a linear warmup of 1500 iterations with initial learning rate 1×10^{-6}. We adjust the weight decay to 0.02. All models use pre-trained weights from ImageNet1K and are trained on 8 GPUs with 2 images per GPU for 160K iterations. For augmentations, we adopt the default setting in mmsegmentation of random horizontal flipping, random re-scaling within ratio range [0.5, 2.0] and random photometric distortion. Stochastic depth with ratio for ParCNetV2-T, ParCNetV2-S, ParCNetV2-B are set to 0.3, 0.3, and 0.5, respectively. All the models are trained on the standard setting as the previous approaches with an input of 512×512. Tab. 6 lists the model size, FLOPs, and mIoU of single-scale and multi-scale testing for different backbones.

H. Inference Acceleration

We implement the implicit gemm algorithm as [4]. To speed up ParCNetV2, we first reconstruct the standard convolution kernel with reparameterization, including separative oversized convolution and local 7×7 convolution. Then we use implicit gemm algorithm to implement the depth-wise convolution. It is worth noting that this transform brings a bit more computational complexity, and only the convolutions of the last three stages run faster under these operations.

Tab. 2 show the original and accelerated inference time

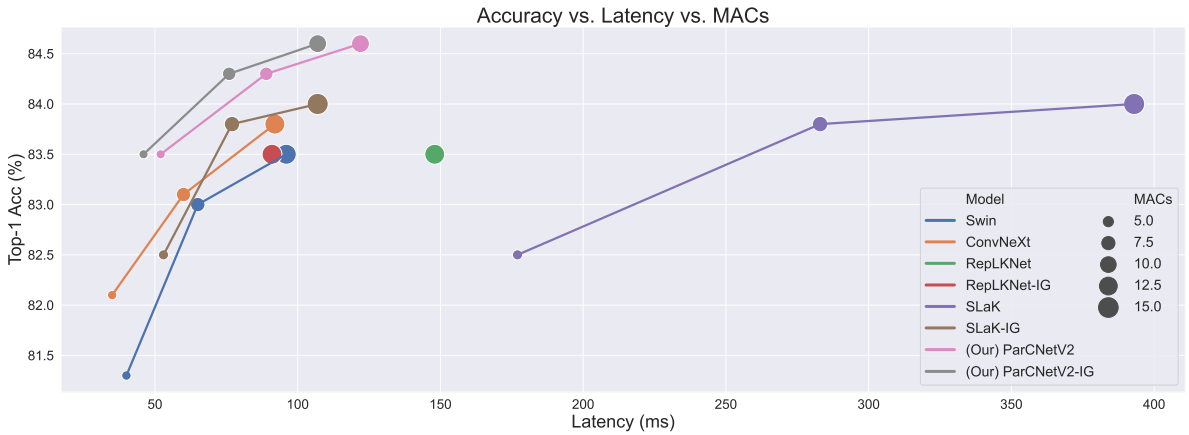
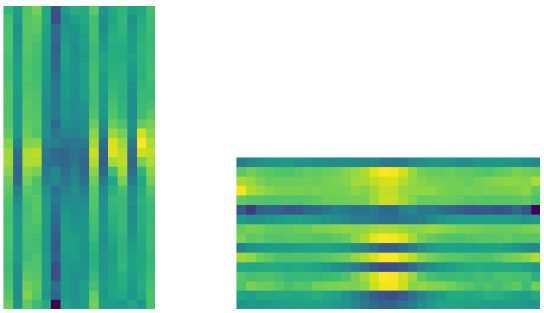


Figure 2: **Inference time and model accuracy. IG**: implicit gemm acceleration.



(a) Vertical convolution kernels. Each column is a channel of the vertical kernel.

(b) Horizontal convolution kernels. Each row is a channel of the horizontal kernel.

Figure 3: The vertical and horizontal oversized convolution kernel of the last uniform block of the third stage. We randomly selected 16 channels as examples.

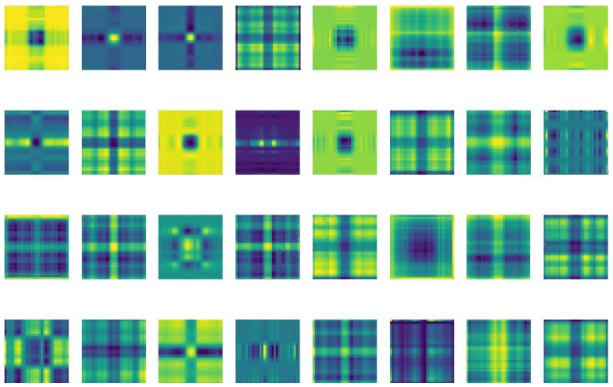


Figure 4: The corresponding oversized convolution kernel of the last uniform block of the third stage. We randomly selected 32 channels as examples.

of ParCNetV2. As illustrated in Figure 2, our proposed ParCNetV2 benefits from optimized algorithms. However, it does not heavily rely on optimization. Even without optimization, ParCNetV2 achieves a better balance between accuracy and speed compared to other large kernel models that have been optimized, such as RepLKNet [4] and SLaK [9]. However, dropping specific optimization for other large kernel models, especially SLaK, significantly affects their speed (as shown by the transition from the earth-colored line to the purple line). After optimization, ParCNetV2 exhibits clear advantages.

I. Visualization of Local and Oversized Convolutions

Our proposed ParCNet V2 involves using an oversized convolution kernel with dimensions $C \times (2H-1) \times 1$ and $C \times 1 \times (2W-1)$, as illustrated in Fig.3. This oversized kernel is effective in capturing global context with a smoother kernel. For further analysis, we reconstruct a sequence of vertical and horizontal convolution kernels into 2D convolution kernels, as shown in Figure4. We observe that different kernels have distinct characteristics, with some focusing on local features and others on longer-range features. This behavior is similar to the attention maps used in vision transformers [5, 13]. Viewed in 2D, the oversized convolution kernels exhibit a wide range of diversity, which makes them well-suited for handling complex global contexts.

J. Visualization of Grad-CAM

We compare the Grad-CAM [14] of our ParCNetV2 against the strong baseline ConvNeXt [11]. ParCNetV2 utilizes global oversized convolutions and an attention mechanism of bifurcate gate units. As shown in Fig. 5, ParCNetV2 either focuses on larger areas of the objects or produces a more smooth activation map, which indicates that

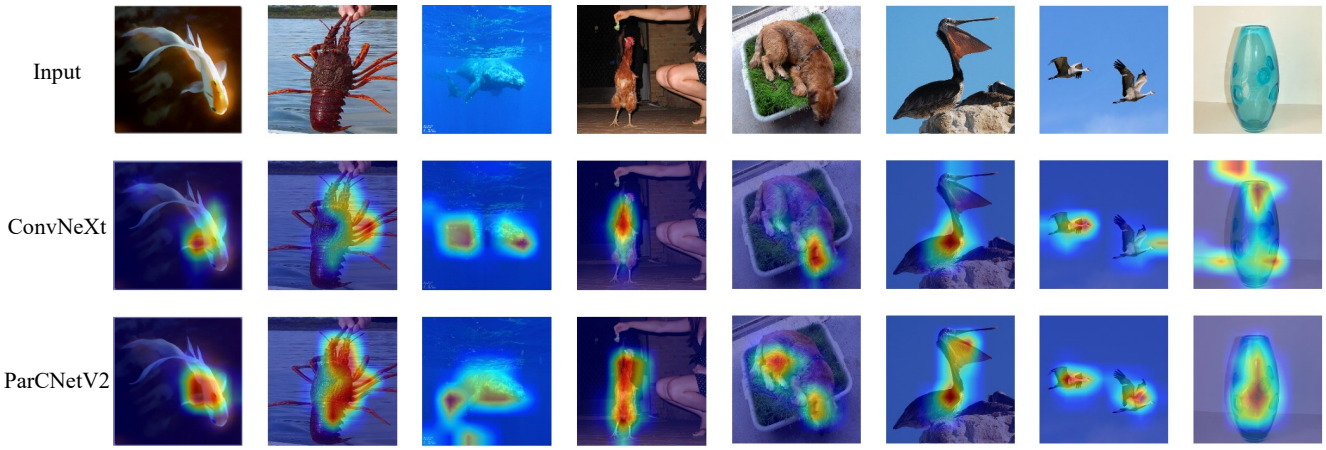


Figure 5: The Grad-CAM of ConvNeXt and our proposed ParCNetV2. The first line is the original image, the second line is the Grad-CAM for ConvNeXt, and the third line is our ParCNetV2.

our model has a stronger ability to capture large objects and texture features.

References

[1] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 3

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 2

[3] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 3

[4] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11963–11975, 2022. 3, 4

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2

[9] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Mykola Pechenizkiy, Decebal Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. *arXiv preprint arXiv:2207.03620*, 2022. 4

[10] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2, 3

[11] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. 2, 3, 4

[12] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020. 2

[13] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128, 2021. 4

[14] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 4

[15] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 418–434, 2018. 3

[16] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 3