HR-NAS: Searching Efficient High-Resolution Neural Architectures with Lightweight Transformers – Supplemental Material –

Mingyu Ding[1*] Xiaochen Lian[2] Linjie Yang[2] Peng Wang[2] Xiaojie Jin[2] Zhiwu Lu[3] Ping Luo[1]
[1]The University of Hong Kong [2]Bytedance Inc.
[3]Gaoling School of Artificial Intelligence, Renmin University of China
{myding, pluo}@cs.hku.hk luzhiwu@ruc.edu.cn
{xiaochen.lian, linjie.yang, peng.wang, jinxiaojie}@bytedance.com

1. Datasets and Settings

In this section, we provide details of the datasets and settings used. We use the same super network for training and evaluation in each task.

In practice, different hyperparameters are often tuned with a validation set for different tasks according to different datasets and losses. For example, HRNet [14] is trained using two different settings for segmentation and keypoint estimation tasks. In this work, we follow the common training settings of each task, *i.e.*, the setting in HRNet [14] for segmentation and keypoint estimation, AtomNAS [9] for classification, and PointPillar [7] for 3D detection.

As for the choice of λ for each task, we first empirically tuned it so that HR-NAS-A's FLOPs is comparable to the least FLOPs among the baseline models, then we relaxed the restriction so that HR-NAS-B reaches SOTA yet still costs less FLOPs than the best baseline models. Currently, the searched model size cannot be controlled precisely by λ. We will strengthen it by incorporating other techniques as our future work. See below for details.

ImageNet for Image Classification. The ILSVRC 2012 classification dataset [4] consists of 1,000 classes, with a number of 1.2 million training images and 50,000 validation images. Follow the common practice in [12, 16, 9, 13, 11], we adopt a RMSProp optimizer with momentum 0.9 and weight decay 1e-5; exponential moving average (EMA) with decay 0.9999; and exponential learning rate decay. The input size is 224×224. The initial learning rate is set to 0.064 with batch size 1024 on 16 Tesla V100 GPUs for 350 epochs, and decays by 0.97 every 2.4 epochs. By setting the coefficient of the L1 penalty term λ to 1.8e-4 and 1.2e-4, we obtain our HR-NAS-A and HR-NAS-B. Unless specified, we adopt the ReLU activation and the basic data augmentation scheme, i.e., random resizing and cropping, and random horizontal flipping, and use single-crop for evaluation. For experiments of HR-NAS†‡, we also adopt the SE module [6], Swish activation [10], and RandAugment [3] for better performance. We report the top-1 Accuracy as the evaluation metric.

Cityscapes for Semantic Segmentation. The Cityscapes dataset [2] contains high-quality pixel-level annotations of 5000 images with size 1024x2048 (2975, 500, and 1525 for the training, validation, and test sets respectively) and about 20000 coarsely annotated training images. Following the evaluation protocol [2], 19 semantic labels are used for evaluation without considering the void label. In this work, the input size is set to 512×1024. We use an AdamW optimizer with momentum 0.9 and weight decay 1e-5; exponential moving average (EMA) with decay 0.9999. The initial learning rate is set to 0.04 with batch size 32 on 8 Tesla V100 GPUs for 430 epochs. The learning rate and momentum follow the onecycle scheduler with a minimum learning rate of 0.0016. By setting the coefficient of the L1 penalty term λ to 1.6e-4 and 6.0e-5, we obtain our HR-NAS-A and HR-NAS-B. We use a basic data augmentation, *i.e.*, random resizing and cropping, random horizontal flipping, and photometric distortion for training and single-crop testing with a test size of 1024×2048. We report the mean Intersection over Union (mIoU), mean (macro-averaged) Accuracy (mAcc), and overall (micro-averaged) Accuracy (aAcc) as the evaluation metrics.

ADE20K for Semantic Segmentation. The ADE20K dataset [17] contains 150 classes and diverse scenes with 1,038 image-level labels. The dataset is divided into 20K/2K/3K images for training, validation, and testing respectively. In this work, the input size and testing size is set to 512×512 and 512×2048, respectively. The model is trained with a batch size of 64 on 8 Tesla V100 GPUs for 200 epochs. We use the same optimizer, learning rate scheduler, data augmentation, and penalty weight λ as in the

*This work was done when Mingyu was a research intern at Bytedance.

Cityscapes dataset. We report the mean Intersection over Union (mIoU) as the evaluation metric.

COCO Keypoint for Human Pose Estimation. The COCO dataset [8] contains over $200,000$ images and $250,000$ person instances labeled with 17 keypoints. We train our model on the COCO `train2017` set, including $57K$ images and $150K$ person instances. We evaluate our approach on the `val2017`, containing 5000 images. In this work, we train the model using input sizes of 256×192 and 384×288 with batch size 384 and 192 on 8 Tesla V100 GPUs for 210 epochs, respectively. Following HRNet [14], the initial learning rate is set to 1e-3 with a multistep scheduler (decayed by a factor of 0.1 in 170 and 200 epochs). We use an Adam optimizer with momentum 0.9 and weight decay 1e-8; exponential moving average (EMA) with decay 0.9999. By setting the coefficient of the L1 penalty term λ to 1e-6 and 1e-8, we obtain our HR-NAS-A and HR-NAS-B. We use random scaling and rotation as only data augmentation for training and single-crop testing. We report average precision (AP), recall scores (AR), AP^M for medium objects, and AP^L for large objects as evaluation metrics.

KITTI for 3D Object Detection. The KITTI 3D object detection dataset [5] is widely used for monocular and LiDAR-based 3D detection. It consists of 7,481 training images and 7,518 test images as well as the corresponding point clouds and the calibration parameters, comprising a total of 80,256 2D-3D labeled objects with three object classes: Car, Pedestrian, and Cyclist. Each 3D ground truth box is assigned to one out of three difficulty classes (easy, moderate, hard) according to the occlusion and truncation levels of objects. In this work, we follow the train-val split [1], which contains 3,712 training and 3,769 validation images. The overall framework is based on Pointpillars [7]. The input point points are projected into bird's-eye view (BEV) feature maps by a voxel feature encoder (VFE). The projected BEV feature maps (496×432) are then used as input of our 2D network for 3D/BEV detection. Following [7], we set, pillar resolution: 0.16m, max number of pillars: 12000, and max number of points per pillar: 100. We use the onecycle scheduler with an initial learning rate of 2e-3, a minimum learning rate of 2e-4, and batch size 16 on 8 Tesla V100 GPUs for 80 epochs. We use an AdamW optimizer with momentum 0.9 and weight decay 1e-2. We apply the same data augmentation, *i.e.*, random mirroring and flipping, global rotation and scaling, and global translation for 3D point clouds as in Pointpillar [7]. At inference time, we apply axis-aligned nonmaximum suppression (NMS) with an overlap threshold of 0.5 IoU. We report standard average precision (AP) as the evaluation metric.

2. Network Architecture

As shown in Fig. 1, we visualize our entire super network used in all experiments. It begins with two 3×3 convolutions with stride 2 and number of channels 24, which are followed by five parallel modules (respectively with 1, 2, 3, 4, 4 branches); a fusion module is inserted between every two adjacent parallel modules, to obtain multi-scale features. The numbers of channels for the four branches in parallel modules are 18, 36, 72, 144, respectively.

Table 1. Comparisons of different projection size s of Transformer on the CityScapes validation set. The query number n is set to 8.

Input size	Params	FLOPs	**mIoU**(%)	mACC(%)	aACC(%)
Baseline	1.120M	1.863G	71.99	80.33	95.40
2×2	1.180M	1.863G	72.27	80.74	95.40
4×4	1.246M	1.864G	73.32	81.76	95.45
8×8	2.273M	1.872G	**74.22**	**82.36**	**95.52**
16×16	18.543M	1.969G	74.18	82.07	95.50

Table 2. Ablation study of our lightweight Transformer with $n = 8$ and $s = 8$ on the CityScapes validation set. Notations: 'Enc' – only the encoder of Transformer is used, 'Enc + Dec' – both the encoder and decoder are used in Transformer, 'channel' – use each channel as a token, 'spatial' – use each spatial position as a token.

Input size	Params	FLOPs	**mIoU**(%)	mACC(%)	aACC(%)
Baseline	1.120M	1.863G	71.99	80.33	95.40
SE [6]	2.101M	1.864G	72.81	81.33	95.35
Non-local [15]	1.317M	2.951G	72.50	81.32	95.34
Enc (spatial)	1.184M	1.866G	72.61	80.97	95.26
Enc (channel)	1.723M	1.869G	73.66	82.10	95.50
Enc + Dec (spatial)	1.204M	1.867G	73.54	81.87	95.44
Enc + Dec (channel)	2.273M	1.872G	**74.22**	**82.36**	**95.52**

3. Ablative Results for Transformer

In this section, we conduct two ablative experiments to study the impact of the projection size s, the encoder-decoder structure, and the attention mechanism on the performance of our lightweight Transformer. For both experiments, we take the searched network on Multi-branch + MixConv space (without Transformer) in Tab.6 of the main paper as a strong baseline.

Projection Sizes. We evaluate our Transformers with different projected spatial sizes s. From Tab. 1 we can see that when s goes from 0 to 8, the mIoU keeps increasing at the expense of small extra cost (*i.e.*, FLOPs). Further increasing s brings no gain in performance but drastically increasing FLOPs. We therefore choose $s = 8$ throughout the experiments.

Attention Structures and Mechanisms. We also conduct ablative experiments to validate the effectiveness of our Transformer. We discuss (1) encoder-decoder structures and (2) two kinds of attention mechanisms by transposing the feature, *i.e.*, 'channel' – use each channel of the flattened feature map as a token, 'spatial' – use each spatial position as a token. As shown in Tab. 2, our Transformer obtains the best performance when both encoder and decoder are used on channel-wise tokens. Our Transformer also significantly outperforms its counterparts such as SE [6] and Non-local [15] on dense prediction tasks. Since the channel-wise

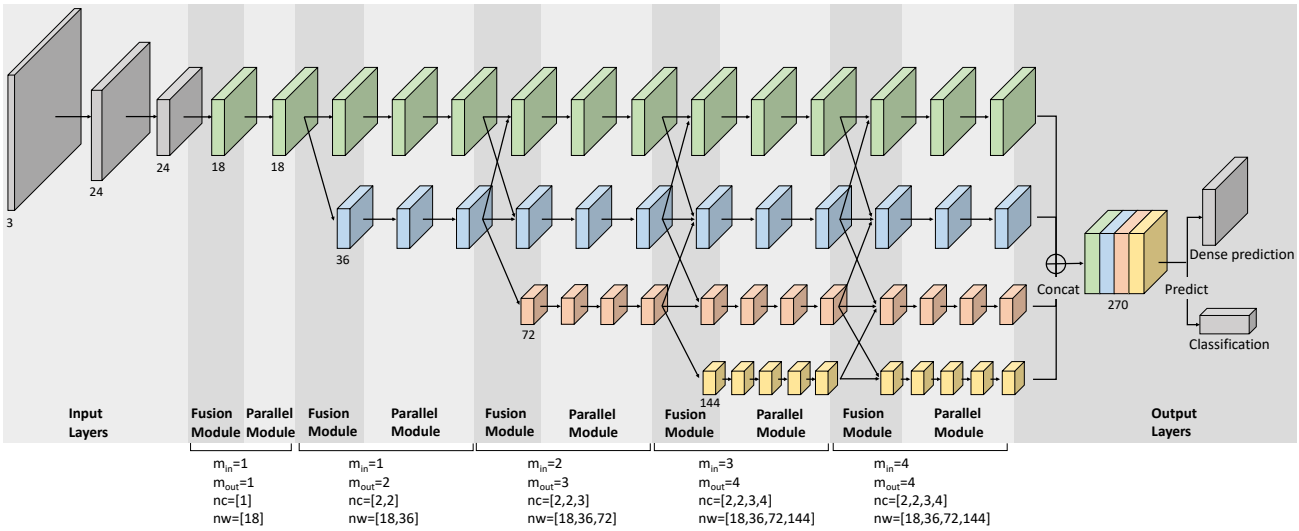

Figure 1. Visualization of our super network architecture. m_{in} and m_{out} denote the input and output numbers of branches in the fusion module. nc and nw denote the number of searching blocks and the number of channels in the parallel module, respectively. The arrows represent the searching blocks and the cubes represent the feature maps. The number under the cube represents the number of channels.

lightweight transformer shows better performance, we set it as the default in this work.

4. Visualization of Visual Recognition Results

We visualize the results of HR-NAS-A on segmentation, human pose estimation, and 3D detection (Fig. 2, 3, 4).

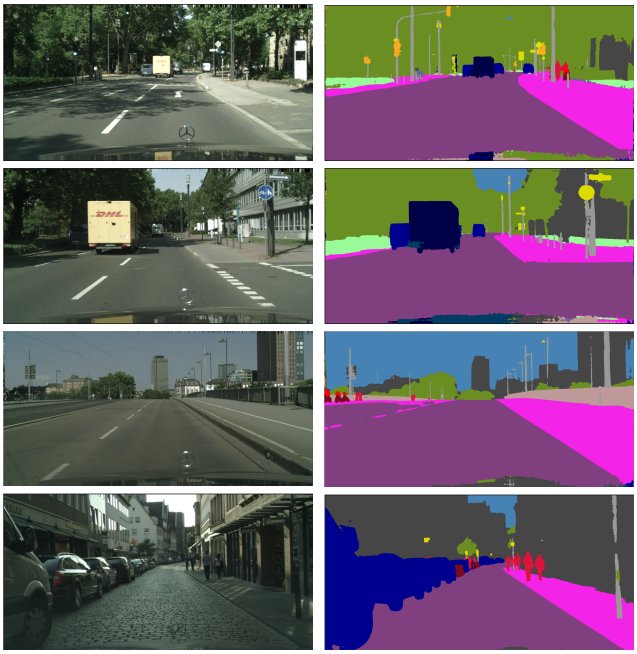

Figure 2. Visualization of semantic segmentation results (left: original images; right: segmentation maps) on Cityscapes.

Figure 3. Visualization of human pose estimation on COCO.

Figure 4. Visualization of 3D object detection results on KITTI.

References

[1] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NeurIPS*, pages 424–432, 2015. 2

[2] 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 *CVPR*, pages 3213–3223, 2016. 1

[3] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, pages 702–703, 2020. 1

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 1

[5] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361, 2012. 2

[6] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, pages 7132–7141, 2018. 1, 2

[7] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, pages 12697–12705, 2019. 1, 2

[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 *ECCV*, pages 740–755. Springer, 2014. 2

[9] Jieru Mei, Yingwei Li, Xiaochen Lian, Xiaojie Jin, Linjie Yang, Alan Yuille, and Jianchao Yang. Atomnas: Fine-grained end-to-end neural architecture search. In *ICLR*, 2020. 1

[10] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. 1

[11] Dimitrios Stamoulis, Ruizhou Ding, Di Wang, Dimitrios Lymberopoulos, Bodhi Priyantha, Jie Liu, and Diana Marculescu. Single-path nas: Designing hardware-efficient convnets in less than 4 hours. In *ECML-PKDD*, pages 481–497. Springer, 2019. 1

[12] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *CVPR*, pages 2820–2828, 2019. 1

[13] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114, 2019. 1

[14] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE TPAMI*, 2020. 1, 2

[15] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803, 2018. 2

[16] Shan You, Tao Huang, Mingmin Yang, Fei Wang, Chen Qian, and Changshui Zhang. Greedynas: Towards fast one-shot nas with greedy supernet. In *CVPR*, pages 1999–2008, 2020. 1

[17] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, pages 633–641, 2017. 1