HRT: High-Resolution Transformer for Dense Prediction

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present a High-Resolution Transformer (HRT) that learns high-resolution representations for dense prediction tasks, in contrast to the original Vision Transformer that produces low-resolution representations and has high memory and computational cost. We take advantage of the multi-resolution parallel design introduced in high-resolution convolutional networks (HRNet), along with local-window self-attention that performs self-attention over small non-overlapping image windows, for improving the memory and computation efficiency. In addition, we introduce a convolution into the FFN to exchange information across the disconnected image windows. We demonstrate the effectiveness of the High-Resolution Transformer on human pose estimation and semantic segmentation tasks.

1 Introduction

Vision Transformer (ViT) [12] shows promising performance on ImageNet classification tasks. Many follow-up works boost the classification accuracy through knowledge distillation [40], adopting deeper architecture [41], directly introducing convolution operations [15, 46], redesigning input image tokens [51], and etc. Besides, some studies attempt to extend the transformer to address broader vision tasks such as object detection [4], semantic segmentation [58, 35], pose estimation [48, 22], video understanding [56, 2, 29], and so on. This work focuses on the transformer for dense prediction tasks, including pose estimation and semantic segmentation.

Vision Transformer splits an image into a sequence of image patches of size 16×16, and extracts the feature representation of each image patch. Thus, the output representations of Vision Transformer lose the fine-grained spatial details that are essential for accurate dense predictions. The Vision Transformer only outputs a single-scale feature representation, and thus lacks the capability to handle multi-scale variation. To mitigate the loss of feature granularity and model the multi-scale variation, we present High-Resolution Transformer (HRT) that contains richer spatial information and constructs multi-resolution representations for dense predictions.

The High-Resolution Transformer is built by following the multi-resolution parallel design that is adopted in HRNet [44]. HRT maintains a high-resolution stream through the entire process with parallel medium- and low-resolution streams helping boost high-resolution representations. With feature maps of different resolutions, HRT is capable to model the multi-scale variation.

At each resolution, the local-window self-attention mechanism is adopted to reduce the memory and computation complexity. We partition the representation maps into a set of non-overlapping small image windows and perform self-attention in each image window separately. This reduces the memory and computation complexity from quadratic to linear with respect to spatial size. We further introduce a 3×3 depth-wise convolution into the feed-forward network (FFN) that follows the local-window self-attention, to exchange information between the image windows which are

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

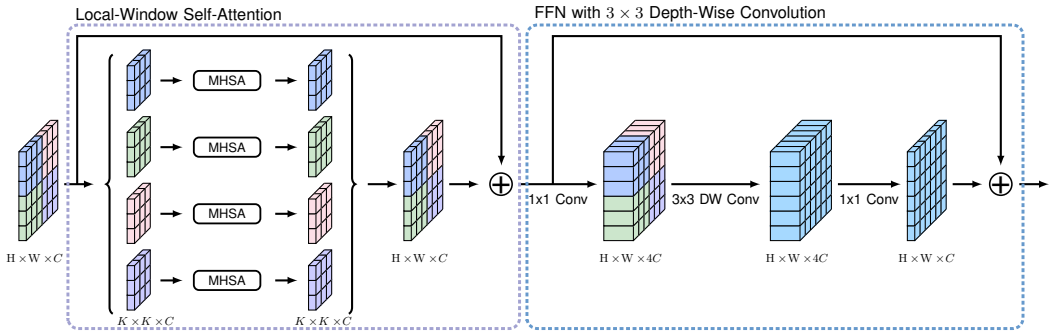


Figure 1: **Illustrating the HRT transformer block.** The HRT transformer block is composed of a local-window self-attention followed by a feed-forward network (FFN) with 3×3 depth-wise convolution. The local-window self-attention applies multi-head self-attention (MHSA) on each window and models the interactions between pixels within the same window. Each window is marked with a different color. The FFN with 3×3 depth-wise convolution exchanges the information across different windows and models the neighboring relations between windows. We mark the feature map after 3×3 depth-wise convolution with a new color to represent that all pixels from four windows are connected together. We use $\oplus$ to represent element-wise addition. H, W, K, and C represent the input height, width, window size, and channel dimension respectively. The above example is based on $\mathrm{H} = \mathrm{W} = 6$ and $K = 3$.

disconnected in the local-window self-attention process. This helps to expand the receptive field and is essential for dense prediction tasks. Figure 1 shows the details of an HRT transformer block.

We conduct experiments on image classification, pose estimation, and semantic segmentation tasks, and achieve competitive performance on various benchmarks. For example, HRT-B gains $+1.0\%$ top-1 accuracy on ImageNet classification over DeiT-B [40] with 40% fewer parameters and 20% fewer FLOPs. HRT-B gains 0.9% AP over HRNet-W48 [39] on COCO `val` set with with 32% fewer parameters and 19% fewer FLOPs. HRT-B + OCR gains $+1.2\%$ and $+2.0\%$ mIoU over HRNet-W48 + OCR [52] with 25% fewer parameters and slightly more FLOPs on PASCAL-Context `test` and COCO-Stuff `test`, respectively.

2 Related work

Vision Transformers. With the success of Vision Transformer (ViT) [12] and the data-efficient image transformer (DeiT) [40], various techniques are proposed to improve the ImageNet classification accuracy of Vision Transformer [11, 41, 46, 15, 51, 16, 5, 26, 21, 38]. Among the very recent advancements, the community has verified several effective improvements such as multi-scale feature hierarchies and incorporating convolutions.

For example, the concurrent works MViT [13], PVT [45], and Swin [26] introduce the multi-scale feature hierarchies into transformer following the spatial configuration of a typical convolutional architecture such as ResNet-50. Different from them, our HRT incorporates the multi-scale feature hierarchies through exploiting the multi-resolution parallel design inspired by HRNet. CvT [46], CeiT [50], and LocalViT [24] propose to enhance the locality of transformer via inserting depth-wise convolutions into either the self-attention or the FFN. The purpose of the inserted convolution within our HRT is different, apart from enhancing the locality, it also ensures information exchange across the non-overlapping windows.

Several previous studies [34, 18] have proposed similar local self-attention schemes for image classification. They construct the overlapped local windows following the strided convolution, resulting in heavy computation cost. Similar to [20, 42, 26], we propose to apply the local-window self-attention scheme to divide the input feature map into non-overlapping windows. Then we apply the self-attention within each window independently so as to improve the efficiency significantly.

There are several concurrently-developed works [58, 35] use the Vision Transformer to address the dense predict tasks such as semantic segmentation. They have shown that increasing the spatial resolution of the representations output by the Vision Transformer is important for semantic segmentation.

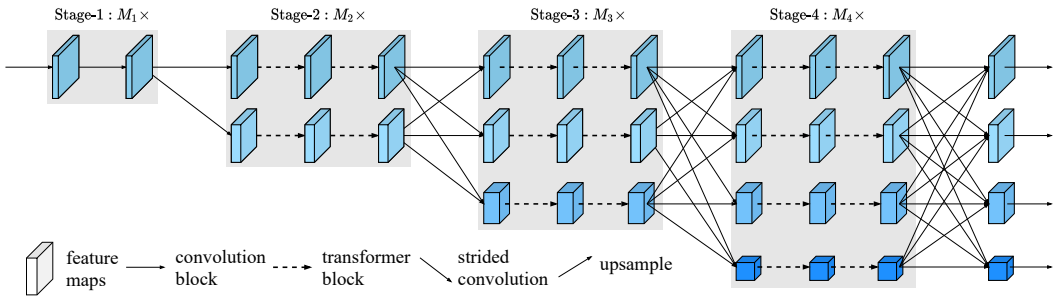


Figure 2: **Illustrating the High-Resolution Transformer architecture.** The multi-resolution parallel transformer modules are marked in gray. $M_1\times$, $M_2\times$, $M_3\times$, and $M_4\times$ indicate the number of times the module is repeated in each stage. Each module consists of multiple successive multi-resolution parallel transformer blocks.

Our HRT provides a different path to address the low-resolution problem of the Vision Transformer via exploiting the multi-resolution parallel transformer scheme.

High-Resolution CNN for Dense Prediction. The high-resolution convolutional schemes have achieved great success on both pose estimation and semantic segmentation tasks. In the development of high-resolution convolutional neural networks, the community has developed three main paths including: (i) applying dilated convolutions to remove some down-sample layers [6, 49], (ii) recovering high-resolution representations from low-resolution representations with decoders [36, 1, 30, 31], and (iii) maintaining high-resolution representations throughout the network [44, 14, 37, 59, 43, 54, 19]. Our HRT belongs to the third path, and retains the advantages of both vision transformer and HRNet [44].

3 High-Resolution Transformer

Multi-resolution parallel transformer. We follow the HRNet [44] design and start from a high-resolution convolution stem as the first stage, gradually adding high-to-low resolution streams one by one as new stages. The multi-resolution streams are connected in parallel. The main body consists of a sequence of stages. In each stage, the feature representation of each resolution stream is updated with multiple transformer blocks independently and the information across resolutions is exchanged repeatedly with the convolutional multi-scale fusion modules.

Figure 2 illustrates the overall HRT architecture. The design of convolutional multi-scale fusion modules exactly follows HRNet. We illustrate the details of the transformer block in the following discussion and more details are presented in Figure 1.

Local-window self-attention. We divide the feature maps $\mathbf{X} \in \mathbb{R}^{N\times D}$ into a set of non-overlapping small windows: $\mathbf{X} \rightarrow \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_P\}$, where each window is of size $K \times K$. We perform multi-head self-attention (MHSA) within each window independently. The formulation of multi-head self-attention on the p-th window is given as:

$$\mathrm{MultiHead}(\mathbf{X}_p) = \mathrm{Concat}[\mathrm{head}(\mathbf{X}_p)_1, \cdots, \mathrm{head}(\mathbf{X}_p)_H] \in \mathbb{R}^{K^2\times D}, \tag{1}$$

$$\mathrm{head}(\mathbf{X}_p)_h = \mathrm{Softmax}\left[\frac{(\mathbf{X}_p\mathbf{W}_q^h)(\mathbf{X}_p\mathbf{W}_k^h)^T}{\sqrt{D/H}}\right]\mathbf{X}_p\mathbf{W}_v^h \in \mathbb{R}^{K^2\times\frac{D}{H}}, \tag{2}$$

$$\widehat{\mathbf{X}}_p = \mathbf{X}_p + \mathrm{MultiHead}(\mathbf{X}_p)\mathbf{W}_o \in \mathbb{R}^{K^2\times\frac{D}{H}}, \tag{3}$$

where $\mathbf{W}_o \in \mathbb{R}^{D\times D}$, $\mathbf{W}_q^h \in \mathbb{R}^{\frac{D}{H}\times D}$, $\mathbf{W}_k^h \in \mathbb{R}^{\frac{D}{H}\times D}$, and $\mathbf{W}_v^h \in \mathbb{R}^{\frac{D}{H}\times D}$ for $h \in \{1, \cdots, H\}$. H represents the number of heads, D represents the number of channels, N represents the input resolutions, and $\widehat{\mathbf{X}}_p$ represents the output representation of MHSA. We also apply the relative position embedding scheme introduced in the T5 model [33] to incorporate the relative position information into the local-window self-attention.

Table 1: **The architecture configuration of HRT.** LSA: local-window self-attention, FFN-DW: feed-forward network with a 3×3 depth-wise convolution, $\{M_1, M_2, M_3, M_4\}$: the number of modules, $\{B_1, B_2, B_3, B_4\}$: the number of blocks, $\{H_1, H_2, H_3, H_4\}$: the number of heads, $\{R_1, R_2, R_3, R_4\}$: the MLP expansion ratios.

Res.	Stage 1	Stage 2	Stage 3	Stage 4
$4\times$	$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, 256 \end{bmatrix} \times B_1 \times M_1$	$\begin{bmatrix} \text{LSA}, H_1 \\ \text{FFN-DW}, R_1 \end{bmatrix} \times B_2 \times M_2$	$\begin{bmatrix} \text{LSA}, H_1 \\ \text{FFN-DW}, R_1 \end{bmatrix} \times B_3 \times M_3$	$\begin{bmatrix} \text{LSA}, H_1 \\ \text{FFN-DW}, R_1 \end{bmatrix} \times B_4 \times M_4$
$8\times$		$\begin{bmatrix} \text{LSA}, H_2 \\ \text{FFN-DW}, R_2 \end{bmatrix} \times B_2 \times M_2$	$\begin{bmatrix} \text{LSA}, H_2 \\ \text{FFN-DW}, R_2 \end{bmatrix} \times B_3 \times M_3$	$\begin{bmatrix} \text{LSA}, H_2 \\ \text{FFN-DW}, R_2 \end{bmatrix} \times B_4 \times M_4$
$16\times$			$\begin{bmatrix} \text{LSA}, H_3 \\ \text{FFN-DW}, R_3 \end{bmatrix} \times B_3 \times M_3$	$\begin{bmatrix} \text{LSA}, H_3 \\ \text{FFN-DW}, R_3 \end{bmatrix} \times B_4 \times M_4$
$32\times$				$\begin{bmatrix} \text{LSA}, H_4 \\ \text{FFN-DW}, R_4 \end{bmatrix} \times B_4 \times M_4$

Table 2: **HRT instances.** HRT-T, HRT-S, and HRT-B represents tiny, small, and base HRT model, respectively.

Model	#modules $\{M_1, M_2, M_3, M_4\}$	#blocks $\{B_1, B_2, B_3, B_4\}$	#channels $\{C_1, C_2, C_3, C_4\}$	#heads $\{H_1, H_2, H_3, H_4\}$
HRT-T	$\{1, 1, 3, 2\}$	$\{2, 2, 2, 2\}$	$\{18, 36, 72, 144\}$	$\{1, 2, 4, 8\}$
HRT-S	$\{1, 1, 4, 2\}$	$\{2, 2, 2, 2\}$	$\{32, 64, 128, 256\}$	$\{1, 2, 4, 8\}$
HRT-B	$\{1, 1, 4, 2\}$	$\{2, 2, 2, 2\}$	$\{78, 156, 312, 624\}$	$\{2, 4, 8, 16\}$

With MHSA aggregates information within each window, we merge them to compute the output $\mathbf{X}^{\text{MHSA}}$:

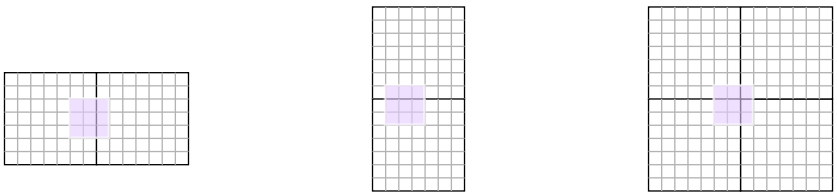
$$\{\widehat{\mathbf{X}}_1, \widehat{\mathbf{X}}_2, \cdots, \widehat{\mathbf{X}}_P\} \xrightarrow{\text{Merge}} \mathbf{X}^{\text{MHSA}}. \tag{4}$$

The left part of Figure 1 illustrates how local-window self-attention updates the 2D input representations, where the multi-head self-attention operates within each window independently.

FFN with 3×3 depth-wise convolution. Local-window self-attention performs self-attention over the non-overlapping windows separately. There is no information exchange across the windows. To handle this issue, we add a 3×3 depth-wise convolution in between the two point-wise MLPs that form the FFN in Vision transformer: $\text{MLP}(\text{DW-Conv.}(\text{MLP}()))$. The right part of Figure 1 shows an example of how FFN with 3×3 depth-wise convolution updates the 2D input representations.

Representation head designs. As shown in Figure 2, the output of HRT consists of four feature maps of different resolutions. We illustrate the details of the representation head designs for different tasks as following: (i) ImageNet classification, we send the four-resolution feature maps into a bottleneck and the output channels are changed to 128, 256, 512, and 1024 respectively. Then, we apply the strided convolutions to fuse them and output a feature map of the lowest resolution with 2048 channels. Last, we apply a global average pooling operation followed by the final classifier. (ii) pose estimation, we only apply the regression head over the highest resolution feature map. (iii) semantic segmentation, we apply the semantic segmentation head over the concatenated representations, which are computed by first upsampling all the low-resolution representations to the highest resolution and then concatenate them together.

Instantiation. We illustrate the overall architecture configuration of HRT in Table 1. We use $\{M_1, M_2, M_3, M_4\}$ and $\{B_1, B_2, B_3, B_4\}$ to represent the number of modules and the number of blocks of {state1, stage2, stage3, stage4}, respectively. We use $\{C_1, C_2, C_3, C_4\}$, $\{H_1, H_2, H_3, H_4\}$, and $\{R_1, R_2, R_3, R_4\}$ to represent the number of channels, the number of heads and the MLP expansion ratios in transformer block associated with different resolutions. We keep the first stage unchanged following the original HRNet and use the bottleneck as the basic building block. We apply the transformer blocks in the other stages and each transformer block consists of a local-window self-attention followed by an FFN with 3×3 depth-wise convolution. We have not included the convolutional multi-scale fusion modules in Table 1 for simplicity. In our implementation, we set the size of the window on four resolution streams as 7×7 by default. Table 2 illustrates the configuration details of three different HRT instances with increasing complexities, where the MLP expansion ratios $\{R_1, R_2, R_3, R_4\}$ are set as $\{4, 4, 4, 4\}$ for all models and are not shown.



a) Horizontal Neighboring Windows b) Vertical Neighboring Windows c) Diagonal Neighboring Windows

Figure 3: **Illustrating how FFN with 3×3 depth-wise convolution connects the non-overlapping windows.** The 3×3 depth-wise convolution (marked with) is capable to propagate the information across the adjacent windows under the 3 typical scenarios, e.g, horizontal, vertical, and diagonal neighboring windows.

Table 3: **Comparisons on ImageNet-1K `val`.**

Method	image size	#param.	FLOPs	Top-1 acc.
ResNet-18 [17]	224×224	11M	1.8G	69.8
ResNet-50 [17]	224×224	26M	4.1G	78.5
ResNet-101 [17]	224×224	45M	7.9G	79.8
HRNet-W18 [44]	224×224	21.3M	4.0G	76.8
HRNet-W32 [44]	224×224	41.2M	8.3G	78.5
HRNet-W48 [44]	224×224	77.5M	16.1G	79.3
RegNetY-4G [32]	224×224	21M	4.0G	80.0
RegNetY-8G [32]	224×224	39M	8.0G	81.7
RegNetY-16G [32]	224×224	84M	16.0G	82.9
ViT-B/16 [12]	224×224	86M	55.4G	77.9
ViT-L/16 [12]	224×224	307M	190.7G	76.5
DeiT-T [40]	224×224	5M	1.3G	72.2
DeiT-S [40]	224×224	22M	4.6G	79.8
DeiT-B [40]	224×224	86M	17.5G	81.8
HRT-T	224×224	8.0M	1.8G	78.5
HRT-S	224×224	13.5M	3.6G	81.2
HRT-B	224×224	50.3M	13.7G	82.8

Analysis. We illustrate how the FFN with 3×3 depth-wise convolution is capable to expand the interactions beyond the non-overlapping local windows and model the relations between them in Figure 3. The benefits of 3×3 depth-wise convolution are twofold: one is enhancing the locality and the other one is enabling the interactions across windows. Therefore, based on the combination of the local-window self-attention and the FFN with 3×3 depth-wise convolution, we can build the HRT transformer block that improves the memory and computation efficiency significantly.

4 Experiments

4.1 ImageNet Classification

Training setting. We conduct the comparisons on ImageNet-1K, which consists of 1.28M `train` images and 50K `val` images with 1000 classes. We train all models with batch size 1024 for 300 epochs with AdamW [27] optimizer, cosine decay learning rate schedule, weight decay as 0.05, and a bag of augmentation policies, including rand augmentation [10], mixup [55], cutmix [53], and so on. HRT-T and HRT-S require $8 \times$ 32G-V100 GPUs and HRT-B requires $32 \times$ 32G-V100 GPUs.

Results. We compare HRT to some representative CNN methods and vision transformer methods in Table 3, where all methods are trained on ImageNet-1K only. The results of ViT-Large with larger dataset such as ImageNet-21K not included for fairness. According to Table 3, HRT achieves

Table 4: **Comparison on the COCO pose estimation `val` set.** The number of parameters and FLOPs for the pose estimation network are measured w/o considering neither human detection nor keypoint grouping. All results are based on ImageNet pretraining. $-$ means the numbers are not provided in the original paper.

Method	input size	#param.	FLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
HRNet-W32 [39]	256×192	28.5M	7.1G	74.4	90.5	81.9	70.8	81.0	78.9
HRNet-W32 [39]	384×288	28.5M	16.0G	75.8	90.6	82.7	71.9	82.8	81.0
HRNet-W48 [39]	256×192	63.6M	14.6G	75.1	90.6	82.2	71.5	81.8	80.4
HRNet-W48 [39]	384×288	63.6M	32.9G	76.3	90.8	82.9	72.3	83.4	81.2
PRTR [22]	512×384	57.2M	37.8G	73.3	89.2	79.9	69.0	80.9	80.2
TransPose-H-A6 [48]	256×192	17.5M	21.8G	75.8	$-$	$-$	$-$	$-$	80.8
TokenPose-L/D24 [23]	256×192	27.5M	11.0G	75.8	90.3	82.5	72.3	82.7	80.9
HRT-T	256×192	2.5M	1.3G	70.9	89.0	78.4	67.2	77.8	76.6
HRT-T	384×288	2.5M	1.8G	72.4	89.3	79.0	68.2	79.7	77.9
HRT-S	256×192	7.8M	2.8G	74.0	90.2	81.2	70.4	80.7	79.4
HRT-S	384×288	7.8M	6.2G	75.6	90.3	82.2	71.6	82.5	80.7
HRT-B	256×192	43.2M	12.2G	75.6	90.8	82.8	71.7	82.6	80.8
HRT-B	384×288	43.2M	26.8G	77.2	91.0	83.6	73.2	84.2	82.0

Table 5: **Comparison on the COCO pose estimation `test-dev` set.** The number of parameters and FLOPs for the pose estimation network are measured w/o considering neither human detection nor keypoint grouping. All results are based on ImageNet pretraining.

Method	input size	#param.	FLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
HRNet-W32 [39]	384×288	28.5M	16.0G	74.9	92.5	82.8	71.3	80.9	80.1
HRNet-W48 [39]	384×288	63.6M	32.9G	75.5	92.5	83.3	71.9	81.5	80.5
PRTR [22]	512×384	57.2M	37.8G	72.1	90.4	79.6	68.1	79.0	79.4
TransPose-H-A6 [48]	256×192	17.5M	21.8G	75.0	92.2	82.3	71.3	81.1	$-$
TokenPose-L/D24 [23]	384×288	29.8M	22.1G	75.9	92.3	83.4	72.2	82.1	80.8
HRT-S	384×288	7.8M	6.2G	74.5	92.3	82.1	70.7	80.6	79.8
HRT-B	384×288	43.2M	26.8G	76.2	92.7	83.8	72.5	82.3	81.2

competitive performance. For example, HRT-B gains 1.0% over DeiT-B while saving nearly 40% parameters and 20% FLOPs.

4.2 Human Pose Estimation

Training setting. We study the performance of HRT on the COCO [25] human pose estimation benchmark, which contains more than 200K images and 250K person instances labeled with 17 keypoints. We train our model on COCO `train` 2017 dataset, including 57K images and 150K person instances. We evaluate our approach on the `val` 2017 set and `test-dev` 2017, containing 5K images and 20K images, respectively.

We follow most of the default training and evaluation settings of `mmpose` [8][1], and change the optimizer from Adam to AdamW. For the training batch size, we choose 256 for HRT-T and HRT-S and 128 for HRT-B due to limited GPU memory. Each HRT experiment on COCO pose estimation task takes $8\times$ 32G-V100 GPUs.

Results. Table 4 reports the comparisons on COCO `val` set. We compare HRT to the representative convolutional method such as HRNet [39] and several recent transformer methods, including PRTR [22], TransPose-H-A6 [48], and TokenPose-L/D24 [23]. HRT-B gains 0.9% with 32% fewer parameters and 19% fewer FLOPs when compared to HRNet-W48 with an input size of 384×288.

[1] https://github.com/open-mmlab/mmpose, Apache License 2.0

Figure 4: Example results of HRT-B on COCO pose estimation `val`: containing occlusion, multiple persons, viewpoint and appearance change.

Table 6: **Comparison on semantic segmentation tasks.** We report the mIoUs on Cityscapes `val`, PASCAL-Context `test`, and COCO-Stuff `test`. ‡ marks the results are obtained with extra pre-training on ADE20K. The number of parameters and FLOPs are measured on the image size, 1024×1024 and the output label map size, $19 \times 1024 \times 1024$. All results are evaluated with multi-scale testing.

Method	#params.	FLOPs	Cityscapes	PASCAL-Context	COCO-Stuff
SETR-PUP [58]	317.8M	2326.7G	82.2	55.3	–
SETR-MLA [58]	309.5M	2138.6G	–	55.8	–
DPT-Hybrid [35]	124.0M	1231.5G	–	60.5‡	–
Deeplabv3 [7]	87.1M	1394.0G	80.7	54.1	–
PSPNet [57]	68.0M	1028.8G	80.0	54.0	–
HRNet-W48 + OCR [52]	74.5M	924.7G	–	56.2	40.5
HRT-T + OCR	4.1M	129.9G	78.1	50.4	34.9
HRT-S + OCR	13.5M	418.3G	81.0	54.6	38.9
HRT-B + OCR	56.0M	1051.6G	82.0	57.3	42.5

Table 7: **Study of larger window size.** The results are evaluated with single-scale testing.

Method	window size	#param.	FLOPs	PASCAL-Context
HRT-T + OCR	7×7	4.1M	129.9G	49.7
	11×11	4.2M	134.3G	50.7
HRT-S + OCR	7×7	13.5M	418.3G	53.8
	11×11	13.6M	426.5G	54.3
HRT-B + OCR	7×7	56.0M	1051.6G	56.3
	11×11	56.1M	1069.6G	57.0

We also report the comparisons on COCO `test-dev` set in Table 5. HRT-B also outperforms HRNet-W48 by around 0.7% with fewer parameters and FLOPs. Figure 4 shows some example results of human pose estimation on COCO `val` set.

4.3 Semantic Segmentation

Cityscapes. The Cityscapes dataset [9] is for urban scene understanding. There are a total of 30 classes and only 19 classes are used for parsing evaluation. The dataset contains 5K high-quality pixel-level finely annotated images and 20K coarsely annotated images. The finely annotated 5K images are divided into $2,975$ `train` images, 500 `val` images and $1,525$ `test` images. We set the initial learning rate as 0.0001, weight decay as 0.01, crop size as 1024×512, batch size as 8, and training iterations as 80K by default. Each HRT + OCR experiment on Cityscapes takes $8\times$ 32G-V100 GPUs.

Table 6 reports the results on Cityscapes `val`. We choose to use HRT + OCR as our semantic segmentation architecture. We compare our method with several well-known Vision Transformer based methods [58, 35] and CNN based methods [6, 57, 52]. Specifically, SETR-PUP and SETR-MLA use the ViT-Large [12] as the backbone. DPT-Hybrid uses the ViT-Hybrid [12] that consists of

Figure 5: Example results of HRT-B + OCR on Cityscapes `val` (left one), COCO-Stuff `test` (middle two), and PASCAL-Context `test` (right two).

Table 8: **Study of the 3×3 depth-wise convolution in FFN.** We report the top1 acc., mIoU, and AP on ImageNet `val`, PASCAL-Context `test`, and COCO pose estimation `val`, respectively. Results on PASCAL-Context are evaluated with single-scale testing. The number of parameters and FLOPs are measured on ImageNet.

Method	#param.	FLOPs	ImageNet	PASCAL-Context	COCO
FFN w/o 3×3 DW-Conv.	7.9M	1.76G	77.83	46.84	66.88
FFN w/ 3× 3 DW-Conv.	8.0M	1.83G	78.48	49.74	70.92

a ResNet-50 followed by 12 transformer layers. Both ViT-Large and ViT-Hybrid are initialized with the weights pre-trained on ImageNet-21K, where both of them achieve around 85.1% top1 accuracy on ImageNet. DeepLabv3 [6] and PSPNet [57] are based on dilated ResNet-101 with output stride 8. According to the fourth column of Table 6, HRT + OCR achieves competitive performance overall. For example, HRT-B + OCR achieves comparable performance with SETR-PUP while saving 70% parameters and 50% FLOPs.

PASCAL-Context. The PASCAL-Context dataset [28] is a challenging scene parsing dataset that contains 59 semantic classes and 1 background class. The `train` set and `test` set consist of $4,998$ and $5,105$ images respectively. We set the initial learning rate as 0.0001, weight decay as 0.01, crop size as 520×520, batch size as 16, and training iterations as 60K by default. We report the comparisons on the fifth column of Table 6. Accordingly, HRT-B + OCR gains 1.1%, 1.5% over HRNet-W48 + OCR, SETR-MLA with fewer parameters and FLOPs, respectively. Notably, DPT-Hybrid achieves the best performance through extra pre-training the models on ADE20K in advance. Each HRT + OCR experiment on PASCAL-Context takes $8\times$ 32G-V100 GPUs.

COCO-Stuff. The COCO-Stuff dataset [3] is a challenging scene parsing dataset that contains 171 semantic classes. The `train` set and `test` set consist of 9K and 1K images respectively. We set the initial learning rate as 0.0001, weight decay as 0.01, crop size as 520×520, batch size as 16, and training iterations as 60K by default. We report the comparisons on the last column of Table 6 and HRT-B + OCR outperforms the previous best-performing HRNet-W48 + OCR by nearly 2%. Each HRT + OCR experiment on COCO-Stuff takes $8\times$ 32G-V100 GPUs. Figure 5 shows some example results on Cityscapes, PASCAL-Context, and COCO-Stuff.

Larger window size. Considering that semantic segmentation models need to process much larger input resolution, e.g., 520×520, compared to ImageNet classification models that typically process smaller resolution, e.g., 224×224, we empirically study the influence of larger window size on semantic segmentation. The ImageNet pre-trained weights of HRT models can support different window sizes w/o retraining. We report the comparisons of different window sizes based on HRT + OCR in Table 7. The results show that increasing the window size from 7×7 to 11×11 consistently improves the performance across all model complexities, e.g., based on HRT-T + OCR, window size 11×11 outperforms window size 7×7 by nearly 1.0% on PASCAL-Context `test`.

4.4 Ablation Experiments

Influence of 3×3 depth-wise convolution within FFN We study the influence of the 3×3 depth-wise convolution within FFN based on HRT-T in Table 8. We observe that applying 3×3 depth-wise convolution in FFN significantly improves the performance on multiple tasks, including ImageNet classification, PASCAL-Context segmentation, and COCO pose estimation. For example, HRT-T + FFN w/ $3\times$ 3 depth-wise convolution outperforms HRT-T + FFN w/o $3\times$ 3 depth-wise convolution by 0.65%, 2.9% and 4.04% on ImageNet, PASCAL-Context and COCO, respectively.

Table 9: **Comparisons to ViT & DeiT on COCO pose estimation** `val`. † marks the method can not fit into 32G-V100 GPUs. OOM: out-of-memory.

Method	feature resolution	image size	#param.	FLOPs	COCO
ViT-Large	$1/16$	256×192	308.5M	60.1G	69.2
ViT-Large†	$1/8$	256×192	307.5M	235.7G	OOM
DeiT-B⚗ ↑ 384	$1/16$	256×192	90.0M	17.9G	69.0
DeiT-B⚗ ↑ 384	$1/8$	256×192	88.9M	67.9G	71.8
HRT-B	$1/4$	256×192	43.2M	12.2G	75.6

Table 10: **Comparisons to HRNet.** We report the top1 acc., mIoU, and AP on ImageNet `val`, PASCAL-Context `test`, and COCO pose estimation `val`, respectively. Results on PASCAL-Context are based on single-scale testing. The number of parameters and FLOPs are measured on ImageNet.

Method	#param.	FLOPs	ImageNet	PASCAL-Context	COCO
HRNet-T	15.6M	2.7G	76.5	47.8	69.3
HRT-T	8.0M	1.8G	78.5	49.3	70.9
HRNet-S	24.5M	5.0G	78.7	52.3	73.1
HRT-S	13.5M	3.6G	81.2	53.8	74.0
HRNet-B	85.3M	20.3G	81.4	55.2	75.1
HRT-B	50.3M	13.7G	82.8	56.3	75.6

Comparison to ViT & DeiT on pose estimation. We report the COCO pose estimation results based on the two well-known transformer models, including ViT-Large [12] and DeiT-B⚗ ↑ 384 [40] in Table 9. The output feature maps of both models are $1/16$ of the input resolution. We also increase the output resolution of both models to $1/8$ and find that only DeiT-B⚗ ↑ 384 is acceptable and ViT-Large suffers from OOM. Notably, ViT-Large is pre-trained on ImageNet21K in advance and then finetuned on ImageNet1K and achieves 85.1% top-1 accuracy. DeiT-B⚗ ↑ 384 is trained on ImageNet1K for 1000 epochs and achieves 85.2% top-1 accuracy. We compare the number of parameters and FLOPs on the fourth and fifth columns of Table 9.

We use ViT-Large/DeiT-B⚗ ↑ 384 as the encoder and add deconvolution modules to upsample the output representations of the encoder from $1/16$ or $1/8$ to $1/4$ following the SimpleBaseline [47]. According to the results reported in Table 9, we observe that the performance of both ViT-Large and DeiT-B⚗ ↑ 384 with feature resolution $1/16$ significantly drop compared to our method due to overfitting. DeiT-B⚗ ↑ 384 with feature resolution $1/8$ achieves competitive performance, which serves as the evidence that proves the importance of high-resolution representation for transformers architecture. In summary, HRT-B achieves better performance with fewer parameters and FLOPs.

Comparison to HRNet. We compare our HRT to the convolutional HRNet with almost the same architecture configurations via replacing all the transformer blocks with the conventional basic block consisting of two 3×3 convolutions. Table 10 shows the comparison results on ImageNet, PASCAL-Context, and COCO. We observe that HRT significantly outperforms HRNet under various configurations with much less model and computation complexity. For example, HRT-T outperforms HRNet-T by 2.0%, 1.5%, and 1.6% on three tasks while requiring only around 50% parameters and FLOPs, respectively. In summary, HRT achieves better performance via exploiting the benefits of transformers such as content-dependent dynamic interactions.

5 Conclusion

In this work, we present the High-Resolution Transformer (HRT), a simple yet effective transformer architecture, for dense prediction tasks, including pose estimation and semantic segmentation. The key insight is to integrate the HRT transformer block, which combines local-window self-attention and FFN with depth-wise convolution to improve the memory and computation efficiency, with the multi-resolution parallel design of the convolutional HRNet. We empirically verify the effectiveness of our HRT on both pose estimation and semantic segmentation tasks.

References

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

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding?, 2021.

[3] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[5] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *arXiv preprint arXiv:2103.14899*, 2021.

[6] 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.

[7] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[8] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. `https://github.com/open-mmlab/mmpose`, 2020.

[9] 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*, 2016.

[10] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.

[11] Stéphane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases, 2021.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[13] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers, 2021.

[14] Damien Fourure, Rémi Emonet, Élisa Fromont, Damien Muselet, Alain Trémeau, and Christian Wolf. Residual conv-deconv grid network for semantic segmentation. In *BMVC*, 2017.

[15] Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference, 2021.

[16] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer, 2021.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3464–3473, 2019.

[19] Junjie Huang, Zheng Zhu, Feng Guo, and Guan Huang. The devil is in the details: Delving into unbiased data processing for human pose estimation. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[20] 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.

[21] Zihang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Xiaojie Jin, Anran Wang, and Jiashi Feng. Token labeling: Training a 85.5% top-1 accuracy vision transformer with 56m parameters on imagenet. *arXiv preprint arXiv:2104.10858*, 2021.

[22] Ke Li, Shijie Wang, Xiang Zhang, Yifan Xu, Weijian Xu, and Zhuowen Tu. Pose recognition with cascade transformers, 2021.

[23] Yanjie Li, Shoukui Zhang, Zhicheng Wang, Sen Yang, Wankou Yang, Shu-Tao Xia, and Erjin Zhou. Tokenpose: Learning keypoint tokens for human pose estimation, 2021.

[24] Yawei Li, Kai Zhang, Jiezhang Cao, Radu Timofte, and Luc Van Gool. Localvit: Bringing locality to vision transformers, 2021.

[25] 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*, 2014.

[26] 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. *arXiv preprint arXiv:2103.14030*, 2021.

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[28] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[29] Daniel Neimark, Omri Bar, Maya Zohar, and Dotan Asselmann. Video transformer network. *arXiv preprint arXiv:2102.00719*, 2021.

[30] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[31] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499, 2016.

[32] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10428–10436, 2020.

[33] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

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

[35] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ArXiv preprint*, 2021.

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241, 2015.

[37] Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In *NIPS*, pages 4053–4061, 2016.

[38] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition, 2021.

[39] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5693–5703, 2019.

[40] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[41] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers, 2021.

[42] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones, 2021.

[43] 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 transactions on pattern analysis and machine intelligence*, 2020.

[44] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2019.

[45] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, 2021.

[46] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers, 2021.

[47] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *European Conference on Computer Vision (ECCV)*, 2018.

[48] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Towards explainable human pose estimation by transformer. *arXiv preprint arXiv:2012.14214*, 2020.

[49] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[50] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers, 2021.

[51] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *CoRR*, abs/2101.11986, 2021.

[52] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. *arXiv preprint arXiv:1909.11065*, 2019.

[53] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019.

[54] Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu. Distribution-aware coordinate representation for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7093–7102, 2020.

[55] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[56] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding, 2021.

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

[58] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021.

[59] Yisu Zhou, Xiaolin Hu, and Bo Zhang. Interlinked convolutional neural networks for face parsing. In *ISNN*, pages 222–231, 2015.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [N/A]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[TODO]**We will release the code soon.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? **[TODO]**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We conduct the pose estimation experiments with mmpose.
   (b) Did you mention the license of the assets? [Yes] We mark the license of mmpose as "Apache 2.0 license".
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We add the URL of mmpose in the footnote.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]