BINARY DENSE PREDICTORS FOR HUMAN POSE ESTIMATION BASED ON DYNAMIC THRESHOLDS AND FILTERING

Xingrun Xing[⋆]*, Yalong Jiang*[⋆]*, Baochang Zhang*[⋆]*, Wenrui Ding*[⋆]*, Yangguang Li*[‡]*, Hongguang Li*[⋆]*, Huan Peng*[†]

[⋆] Beihang University
[†] Huazhong University of Science and Technology
[‡] SenseTime Group

ABSTRACT

Binary neural networks (BNNs) contribute a lot to the efficiency of image classification models. However, in dense predication tasks such as human pose estimation, predictions in different locations are coupled and rely on the extraction of features across entire images. As a result, more robust and adaptive binarization is required to bridge the performance gap between binarized and full precision models. We propose two approaches to conduct image-aware and pixel-aware dynamic binarization in a model for human pose estimation. Firstly, a simplified dynamic thresholding is leveraged in the backbone to determine unique binarization thresholds for each image. Secondly, in the decoder, we decouple binarization for each pixel according to the activations surrounding the pixel. Dynamic filtering modules are proposed to determine a different binarization strategy for each pixel. Compared with the strong baselines, the proposed framework improves 5.2% and 3.6% mAP on the COCO test-dev benchmark for ResNet-18/34 architectures respectively.

***Index Terms*—** Human pose estimation, binary neural network, dense predication

1. INTRODUCTION

Human pose estimation [1, 2], as a dense prediction task, lays the foundation for human-centric image understanding tasks in computer vision. Given an image, a pose estimator determines the coordinates of human anatomical parts [3] [4]. Boosted by large-scale human-centric datasets and increasing model capacity, state-of-the-art convolutional models have shown extraordinary results in COCO [5] and MPII [6] datasets in recent years. However, under practical scenarios such as mobile and real-time applications, there are limitations in computational cost and latency. Full precision convolutional models require massive operations and memory resources, which makes it hard for implementation on embedded devices.

To tackle this problem, model compression, including knowledge distillation [7], low-rank decomposition, pruning [8], quantitation [9], are widely used. Binary neural networks (BNNs) [10, 11, 12, 13, 14] are usually used for classification and detection [15] tasks. In this work, we adopt BNNs for pose estimation, which saves up to $32\times$ memory and $64\times$ operations in a binary layer [16]. However, one of the major problems that prevents BNNs from practical applications is that the variance of large-scale datasets causes severe quantization error and performance degradation. The degradation becomes more serious for dense prediction tasks than for image classification tasks, because output heatmaps are coupled in different locations. The correlation requires more adaptive binarization strategies to retain contextual cues.

Specifically, two major problems arise in binarizing pose estimators: variations of human actions leading to instance-specific binarization and uniform binarization in each location ignoring the spatial interactions in heatmaps. **(a)** Previous studies [12] indicate that learnable binarization thresholds help to reduce quantitation error. Thresholds keep fixed for all image once the training stage finished. However, the optimal value fitted to the whole training set may be sub-optimal for each specific human sample. **(b)** For the decoder, different locations in one heatmap couple to one keypoint. By leveraging a uniform threshold throughout all spatial locations may break the contextual cues, disabling BNNs from jointly extracting the features in each spatial location and modeling the relations between different locations. By adapting to pixels, the capability in performing the above-mentioned operations can be improved.

Based on Single Baseline [17] networks, we binarize the backbone using dynamic thresholds, and decoder features with leveraged binary dynamic filtering modules. Firstly, for the backbone, we infer image-aware thresholds instead of learning fixed binarization thresholds. A lightweight transpose matrix is applied to generate dynamic thresholds in inference. Secondly, for the decoder, we decouple uniform binarization for both weights and activations in different pixels based on dynamic filtering. Overall, we propose a unified binary pose estimator named BiPose which exceeds the state-of-the-arts [18] on MPII single person dataset. Furthermore, on challenging COCO multi-person dataset, we first employ a BNN method and express classification BNNs a lot.

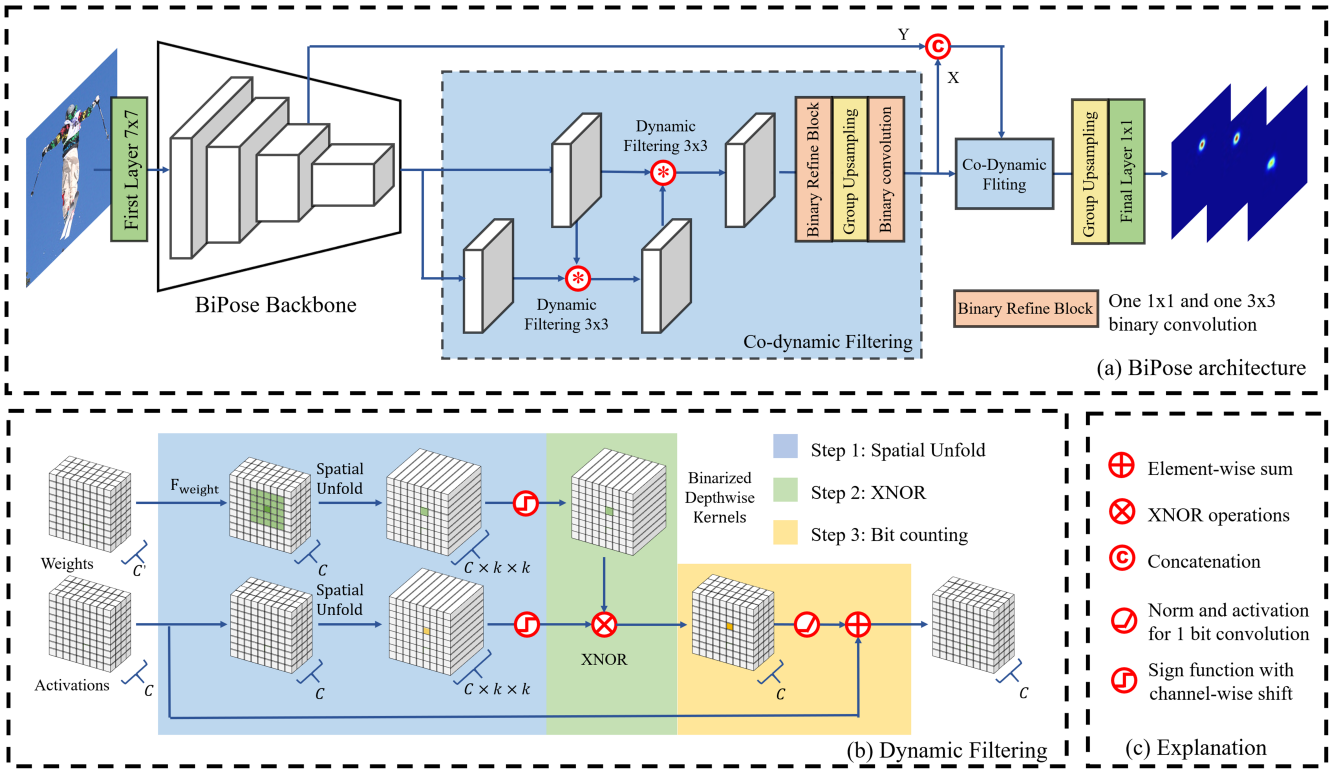

Fig. 1. (a) The overall architecture of BiPose; (b) binary dynamic filtering; (c) explanation. For the decoder, we introduce dynamic filtering modules to decouple binarization in each pixel.

2. METHOD

2.1. Dynamic binary neural network backbone

In the backbone, we acquire binarization functions and activation functions dynamically. Different with using reinforcement learning [19], we directly train a routing function. For each backbone block, we firstly obtain the mean values of each channel using global average pooling (GAP). As shown in Eq. (1), matrix products then transform mean values of each channel to latent embeddings $\alpha(\mathbf{X})$ using weight matrix $\mathbf{W_1W_2}$, where $\mathbf{W_1} \in \mathbf{R^{C\times 0.25C}}$ and $\mathbf{W_2} \in \mathbf{R^{0.25C\times C}}$:

$$\alpha(X) = \left(\frac{1}{H \times W}\sum_{i=0}^{H}\sum_{j=0}^{W} X_{ij}\right) W_1 W_2. \tag{1}$$

Note that, the matrix products increase negligible computational cost even compared with binary convolutions. For example, for a $64 \times 64 \times 64$ sized input, cost is $1/(64 \times 2 \times 9)$ overhead compared with a 3×3 binary convolution layer with 64 filters. The latent embeddings $\alpha(X)$ can then interact with binarization and activation functions as followings.

To binarize activations precisely, instead of fixed thresholds, we adopt simplified dynamic thresholds based on the dynamic sign (DSign) function. After adding a trainable $bias_\beta$, a channel-wise binarization threshold $\beta(\mathbf{X})$ is generated:

$$\beta(X) = \alpha(X) + bias_\beta. \tag{2}$$

To adopt channel-wise dynamic thresholds, we directly add $\beta(X)$ before the sign function to determine values more or less than a threshold as -1 or $+1$ respectly:

$$x_b = Q_b(x) = DSign(x) = \begin{cases} -1, if x > \alpha(x) \\ +1, otherwise \end{cases} \tag{3}$$

At the same time, we also share the latent embeddings $\alpha(X)$ to adjust activation function. A dynamic PReLU (DPReLU) activation function is proposed after adding another bias, $bias_\gamma$:

$$\gamma(X) = \alpha(X) + bias_\gamma, \tag{4}$$

$$DPReLU\,(X) = PReLU\,(X + \gamma(X))\,. \tag{5}$$

As a result, given image-aware latent embeddings $\alpha(X)$, we can determine binarization thresholds using DSign functions and adjust binary convolutions outputs using DPReLU function for each layer.

Method	backbone	w/a	1/64BOPs	FLOPs	OPs	Head	Shld	Elbow	Wrist	Hip	Knee	Ankle	PKCh0.5
Hourglass[1]	Hourglass x8	32/32	–	–	–	97.3*	96.0*	90.2*	85.2*	89.1*	85.1*	82.0*	89.3*
Hourglass[18]	Hourglass x1	32/32	–	–	–	96.8	93.8	86.4	80.3	87.0	80.4	75.7	85.5
Hourglass[18]		1/1	–	–	–	94.7	89.6	78.8	71.5	79.1	70.5	64.0	78.1
Single baseline[17]	ResNet18	32/32	0	7.72G	7.72G	96.3	94.5	86.0	79.8	86.6	81.1	77.2	86.5
ReActNet [12]		1/1	167M	200M	367M	95.2	92.2	81.5	74.6	83.8	74.4	68.0	82.3
BiPose†		1/1	167M	200M	367M	95.4	93.1	82.6	75.8	84.1	75.9	71.2	83.4
BiPose		**1/1**	**200M**	**200M**	**400M**	**95.9**	**93.4**	**83.9**	**77.1**	**84.8**	**77.5**	**72.7**	**84.4**
Single baseline[17]	ResNet34	32/32	0	12.56G	12.56G	96.6	95.0	88.1	82.4	88.0	83.3	79.5	88.1
ReActNet[12]		1/1	243M	200M	443M	95.4	92.6	83.5	76.4	84.7	76.7	71.7	83.8
BiPose†		1/1	243M	200M	443M	95.7	93.6	84.7	77.7	85.9	78.1	73.9	85.0
BiPose		**1/1**	**275M**	**200M**	**475M**	**96.1**	**93.7**	**85.1**	**78.3**	**85.9**	**79.6**	**74.5**	**85.4**
ReActNet-A	MobileNetV1	1/1	291M	74M	365M	95.1	92.6	82.7	75.3	84.6	75.6	69.3	83.1
BiPose		**1/1**	**323M**	**74M**	**397M**	**95.5**	**93.5**	**84.2**	**77.1**	**85.3**	**77.8**	**73.3**	**84.5**

Table 1. Results on MPII validation set. Note that we only calculate operations for convolutional layers. '†' means just using dynamic backbone. Also note that OPs of BiPose(R18) is smaller than ReActNet(R34), but has better performance.

2.2. Dynamic filtering module

We first introduce a binary dynamic filtering block, based on which, a co-dynamic filtering architecture is proposed to execute pixel-aware adaptive binarization in the decoder. As shown in Fig. 1 (b), there are two paths in dynamic filtering which serve as weights and activations respectively. Instead of training fixed binarized kernels directly, we train a binary function F_{weight} to generate convolution kernels for each pixel. Binary convolutional operations are subsequently executed. In our experiments, we use one 1×1 and one 3×3 BNN block as the kernel generation function F_{weight}. In practice, we also concatenate original activation X and the same sized feature Y from the BNN backbone as shown in Fig. 1 (a). Next, a binary kernel bank $\hat{X}$ for each pixel is generated as followings and $\mathbf{W}$ is binary parameters in F_{weight}:

$$\hat{X} = F_{weight}\left(Concate\left(\mathbf{X}, \mathbf{Y}\right); \mathbf{W}\right) \tag{6}$$

For both paths, the binarization function and their gradient approximations are illustrated in Eq. (7), where x is activations or weights to be quantized.

$$\frac{\partial Q_b\left(x\right)}{\partial x} \approx \frac{\partial \widetilde{sign}\left(x\right)}{\partial x} = \begin{cases} 2x+2, & -1 < x < 0 \\ -2x+2, & 0 < x < 1 \\ 0, & otherwise \end{cases} \tag{7}$$

$$w.r.t. \quad sign(x) \approx \widetilde{sign}\left(x\right) = \begin{cases} x^2+2x, & -1 < x < 0 \\ -x^2+2x, & 0 < x < 1 \\ 0, & otherwise \end{cases}$$

The core operations in binarized dynamic filtering can be divided into three steps in Fig. 1 (b). In the first step, we unfold both paths within a $k \times k$ local patch for each pixel. At the same time, pixel-wise binary convolutional kernels are generated according to local features. So that, the uniform binary convolutional kernels are decoupled in different locations. In the second step, we match activations and corresponding weights using XNOR; in the third step, we aggregate unfolded patches using bit-counting. The dynamic filtering operations can be lightweight because XNOR and bit-counting form binary depthwise convolutional operations. The overall calculations are illustrated in Eq. (8), where $U_{k\times k}$ means to unfold $k \times k$ patches.

$$Z = bitcount\left(XNOR\left(U_{k\times k}\left(\widehat{\mathbf{X_b}}\right), U_{k\times k}\left(\mathbf{X_b}\right)\right)\right) \tag{8}$$

3. EXPERIMENTS

Implementation details: In BiPose, we apply dynamic approaches in binarizing ResNet-18, ResNet-34 and MobileNet-V1 which are then leveraged as backbones. In upsampling layers, we apply a floating-point group convolutional layer which is followed by a 3×3 BNN block. We don't binarize this layer because it influences the results a lot. To reduce computational cost, we set groups in this layer to be the highest. A refine block including a 1×1 and a 3×3 binary convolutions also attached. In our experiments, we use the state-of-the-art ReActNet as a strong BNN baseline for comparison. To save operations, we modify the stem layer as $\{3 \times 3$conv (output=32), ReLU, 3×3conv (groups=8, output=64)$\}$ in all binary models.

Data sets and training: we use MPII datasets for single person pose estimation and use COCO2017 datasets for multi-person pose estimation. For both datasets, we use [17] as the full precision baseline and keep the same hyper-parameters, and ReActNets as the binary baseline. Adam optimizers and heatmap decoding methods [3] are used in all binary models. For multi-person estimation, we keep the same settings and detecting results following other top-down methods [24] for fair comparison.

Method	Backbone	Input size	w/a	OPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
CMU-Pose[20]	–	368×654	32/32	9.0G*	61.8	84.9	67.5	57.1	68.2	–
Asso. Emb.[21]	Hourglass	512×512	32/32	206.9G*	65.0	86.7	71.3	59.7	72.5	–
Mask-RCNN[22]	ResNet50	600×800	32/32	–	62.7	87.0	68.4	57.4	71.1	–
G-RMI[23]	ResNet101	353×257	32/32	57.0G	64.9	85.5	71.3	62.3	70.0	69.7
Single baseline[17]	ResNet18	192×256	32/32	5.79G	66.5	89.4	73.9	63.5	72.2	72.3
ReActNet[12]			1/1	275M	53.5	82.9	57.5	50.6	58.6	60.1
BiPose (ours)			1/1	300M	58.7	85.8	64.5	55.8	63.9	65.2
Single baseline[17]	ResNet34	192×256	32/32	9.42G	69.8	90.6	77.7	66.8	75.5	75.4
ReActNet[12]			1/1	332M	58.2	85.4	63.9	55.3	63.4	64.5
BiPose (ours)			1/1	352M	61.8	87.2	68.4	59.0	67.0	67.9
ReActNet-A[12]	MobileNetV1	192×256	1/1	275M	56.9	84.9	62.1	53.9	62.3	63.3
BiPose (ours)			1/1	299M	60.8	87.0	67.3	58.0	66.1	67.1

Table 2. Experiments on COCO test-dev set. Note that we only calculate operations for convolutional layers.

3.1. Single person pose estimation on MPII

We count the numbers of binary operations (BOPs), float operations (FLOPs) and combined operations (OPs) according to $OPs = FLOPs + 1/64BOPs$. We evaluate BiPose models both with and without binary dynamic filtering modules in the decoder. Compared with previous binary networks for pose estimation [18], BiPose exceeds Bulat et al. by 6.3% and 7.3% using ResNet 18 and 34 backbones respectively; on the other hand, compared with ReActNet backbone, dynamic binary methods are able to improve expression ability dramatically while keeping operations slightly overhead. When we compare BiPose with their full precision counterparts, we observe there is still a $2 \sim 3\%$ gap, but at the same time 94.8% operations are saved.

3.2. Multi-person pose estimation on COCO

We also evaluate BiPose in multi-person pose estimation tasks. In Table 2, BiPose is able to boost performance in large scale datasets by a large margin compared with ReActNets. In ResNet-18, 34 and MobileNet-V1 architectures, BiPose make 5.2%, 3.6% and 3.9% mAP progress on COCO test-dev set. Moreover, when we compare BiPose (ResNet34 backbone) with some classic models such as CMU-pose and mask-RCNN, we achieve comparable performance. Note that this is the first time that BNNs approximates the performance of full precision models in practical applications for pose estimation, which is harder than simple classification tasks.

3.3. Ablation studies on ImageNet-1K

To further explore the effects of the dynamic binarization backbone, we compare BiPose backbone with state-of-the-art binary backbone ReActNets in ResNet18 architecture. We train all binary models on the same condition. As shown in Table 3, although just adopting DSign functions decrease classification accuracy, we find that adding both DSign and DPReLU functions makes out the best performance, which indicates the effectiveness of BiPose backbone.

Method	DSign	DPReLU	Top1 (%)
ReActNet	✗	✗	59.83
BiPose backbone	✓	✗	58.73
BiPose backbone	✗	✓	61.15
BiPose backbone	✓	✓	**61.76**

Table 3. Classification results on ImageNet-1K for the backbone part. We fast train 125000 iterations from scratch with batchsize 512.

4. CONCLUSION

In this work, we explore adaptive binarization methods including image-aware binarization thresholds and pixel-aware co-dynamic filtering. As is shown by experiments, adaptive binarization can largely narrow the gap between binary and full precision models while BiPose saves more than 94% operations in bitwidth of 1. Compared with currently binarization frameworks, dynamic thresholds and dynamic filtering modules retain image specific and contextual cues during binarization and are able to exceed state-of-the-arts.

5. ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (U20B2042) and the National Natural Science Foundation of China (62076019).

6. REFERENCES

[1] Alejandro Newell, Kaiyu Yang, and Jia Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*. Springer, 2016, pp. 483–499.

[2] Ke Li, Shijie Wang, Xiang Zhang, Yifan Xu, Weijian Xu, and Zhuowen Tu, "Pose recognition with cascade transformers," in *CVPR*, 2021, pp. 1944–1953.

[3] Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu, "Distribution-aware coordinate representation for human pose estimation," in *CVPR*, 2020, pp. 7093–7102.

[4] 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 *CVPR*, 2020, pp. 5700–5709.

[5] 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*. Springer, 2014, pp. 740–755.

[6] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele, "2d human pose estimation: New benchmark and state of the art analysis," in *CVPR*, 2014, pp. 3686–3693.

[7] Feng Zhang, Xiatian Zhu, and Mao Ye, "Fast human pose estimation," in *CVPR*, 2019, pp. 3517–3526.

[8] Xingchao Liu, Mao Ye, Dengyong Zhou, and Qiang Liu, "Post-training quantization with multiple points: Mixed precision without mixed precision," in *AAAI*, 2021, vol. 35, pp. 8697–8705.

[9] Feng Zhu, Ruihao Gong, Fengwei Yu, Xianglong Liu, Yanfei Wang, Zhelong Li, Xiuqi Yang, and Junjie Yan, "Towards unified int8 training for convolutional neural network," in *CVPR*, 2020, pp. 1969–1979.

[10] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[11] Mingbao Lin, Rongrong Ji, Zihan Xu, Baochang Zhang, Yan Wang, Yongjian Wu, Feiyue Huang, and Chia-Wen Lin, "Rotated binary neural network," *NeurIPS*, vol. 33, 2020.

[12] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng, "Reactnet: Towards precise binary neural network with generalized activation functions," *arXiv preprint arXiv:2003.03488*, 2020.

[13] Xinrui Jiang, Nannan Wang, Jingwei Xin, Keyu Li, Xi Yang, and Xinbo Gao, "Training binary neural network without batch normalization for image super-resolution," in *AAAI*, 2021, vol. 35, pp. 1700–1707.

[14] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *ECCV*. Springer, 2016, pp. 525–542.

[15] Ziwei Wang, Jiwen Lu, Ziyi Wu, and Jie Zhou, "Learning efficient binarized object detectors with information compression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021.

[16] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng, "Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm," in *ECCV*, 2018, pp. 722–737.

[17] Bin Xiao, Haiping Wu, and Yichen Wei, "Simple baselines for human pose estimation and tracking," in *ECCV*, 2018, pp. 466–481.

[18] Adrian Bulat and Georgios Tzimiropoulos, "Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources," in *ICCV*, 2017, pp. 3706–3714.

[19] Ziwei Wang, Jiwen Lu, and Jie Zhou, "Learning channel-wise interactions for binary convolutional neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3432–3445, 2021.

[20] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[21] Alejandro Newell, Zhiao Huang, and Jia Deng, "Associative embedding: End-to-end learning for joint detection and grouping," in *NeurIPS*, 2017, pp. 2277–2287.

[22] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *ICCV*, 2017, pp. 2961–2969.

[23] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy, "Towards accurate multi-person pose estimation in the wild," in *CVPR*, 2017, pp. 4903–4911.

[24] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang, "Deep high-resolution representation learning for human pose estimation," in *CVPR*, 2019, pp. 5693–5703.