Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 1333250, 11 pages
https://doi.org/10.1155/2021/1333250

Research Article

Designing Compact Convolutional Filters for Lightweight Human Pose Estimation

Shili Niu,[1] Weihua Ou,[1] Shihua Feng,[1] Jianping Gou,[2] Fei Long,[3,4] Wenchuan Zhang,[1] and Wu Zeng[5]

[1]*School of Big Data and Computer Science, Guizhou Normal University, Guiyang 550025, China*
[2]*School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212000, China*
[3]*College of Artificial Intelligence and Electrical Engineering, Guizhou Institute of Technology, Guiyang 550003, China*
[4]*Special Key Laboratory of Artificial Intelligence and Intelligent Control of Guizhou Province, Guiyang 550003, China*
[5]*School of Electrical and Electronic Engineering, Wuhan Polytechnic University, Wuhan 430000, China*

Correspondence should be addressed to Weihua Ou; ouweihuahust@gmail.com

Received 19 August 2021; Revised 18 November 2021; Accepted 29 November 2021; Published 17 December 2021

Academic Editor: Chi-Hua Chen

Existing methods for human pose estimation usually use a large intermediate tensor, leading to a high computational load, which is detrimental to resource-limited devices. To solve this problem, we propose a low computational cost pose estimation network, MobilePoseNet, which includes encoder, decoder, and parallel nonmaximum suppression operation. Specifically, we design a lightweight upsampling block instead of transposing the convolution as the decoder and use the lightweight network as our downsampling part. Then, we choose the high-resolution features as the input for upsampling to reduce the number of model parameters. Finally, we propose a parallel OKS-NMS, which significantly outperforms the conventional NMS in terms of accuracy and speed. Experimental results on the benchmark datasets show that MobilePoseNet obtains almost comparable results to state-of-the-art methods with a low compilation load. Compared to SimpleBaseline, the parameter of MobilePoseNet is only 4%, while the estimation accuracy reaches 98%.

1. Introduction

Human pose estimation is also called human key point detection. Its main task is to detect the key points of human body (eyes, nose, shoulders, elbows, etc.) in a given RGB picture. Human pose estimation is one of the basic tasks of computer vision and has many practical applications, such as human-computer interaction [1], human tracking [2], and motion analysis [3]. In recent years, with the quick development of neural networks, human pose estimation based on deep neural networks [4–9] has gained a high accuracy. However, these works have focused only on improving the accuracy of pose estimation through the use of complex and computationally expensive models, while largely ignoring the issue of the cost of model inference. Many methods already require computational resources beyond the capabilities of many mobile and embedded devices. At the same time, information security is a growing concern for people, and it is important to deploy applications directly on edge devices for personal information protection, which leads to high requirements for the computational volume and complexity of human pose estimation models.

Many works have been proposed to solve this problem by building human pose estimation networks with small model size and low computing cost [7, 10, 11]. For example, there is a recent attempt [10] to construct pose estimation models with fewer parameters using quantitative methods, but the performance of the obtained model largely degraded. Also, some researchers [7] try to use knowledge distillation to reduce the parameters of the model, but the model training time and deployment time are increased. On the other hand, some works attempt [12] to find a lightweight pose

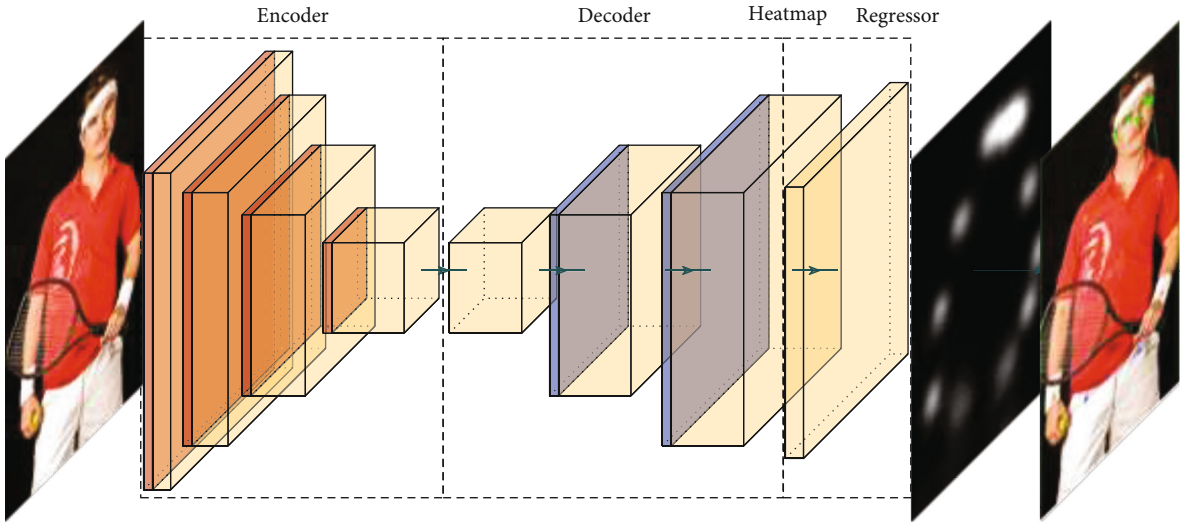

Figure 1: The architecture of the presented MobilePoseNet.

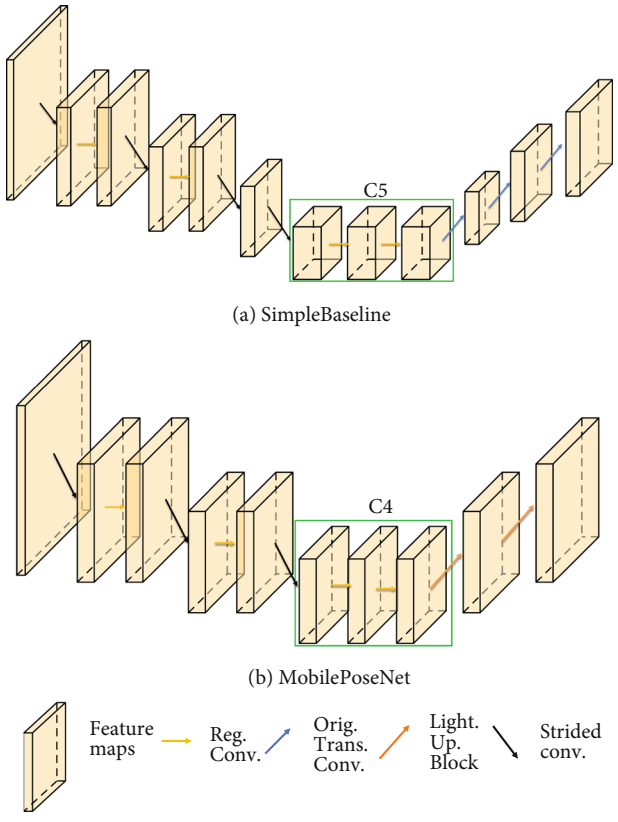

Figure 2: The differences between our proposed network and the SimpleBaseline structure. We can see from the figure that we choose C4 as the input for upsampling, and the implement time of our network is less compared to SimpleBaseline.

estimation model using neural structure search methods. However, the obtained model has a complex structure and slow inference speed. The key problem is how to balance the model accuracy and the inference efficiency.

To address this problem, in this paper, we propose a lightweight human pose estimation network specifically for mobile and resource-constrained environments by designing compact convolutional filters. As shown in Figure 1, our model contains three main parts: an encoder, a decoder, and a heat map regressor that estimates each key point. To keep the model lightweight, we use the first 13 layers of MobileNetV3 [13] as our encoder. Intuitively, high resolution is beneficial for human pose estimation, so we design the encoder with less downsampling, and the structure of the specific model and the comparison of SimpleBaseline can be seen in Figure 2. In the decoder part, inspired by the bottleneck block, we propose a lightweight upsampling module, whose concrete structure is shown in Figure 1. The detailed structure of the overall model is shown in Table 1. Finally, we also propose a parallel OKS-based NMS to further improve the operation speed of pose estimation. Experimental results show that our method can achieve 69.0 AP with only 1.5M model parameters and 1.23 GFLOP calculation amount under the condition of less cost. The contributions of the proposed method are summarized as follows:

(i) We design a lightweight upsampling block that integrates separable transpose convolution and channel-based attention. This is achieved by extensively examining the upsampling modules in existing state-of-the-art deep convolutional networks

(ii) We reduce the number of upsampling and use lightweight upsampling blocks to achieve a lightweight pose estimation network. In particular, we balance the accuracy of the model and the inference speed of the model, which is a key issue to be addressed in extending existing depth-pose estimation methods to practical applications

(iii) We propose a parallel OKS-NMS by combining Matrix-NMS [14] and OKS-NMS [15] to further improve the efficiency of the human pose estimation system

The rest of this paper is organized as follows. We briefly review the related work in the second section and followed

Table 1: Specification for MobilePoseNet. SE denotes whether there is a squeeze-and-excite in that block. NL denotes the type of nonlinearity used. Here, HS denotes h-swish and RE denotes relu. LPB is our proposed lightweight upsampling block. bneck is the bottleneck block in MobileNetV3. k is the number of key points.

Input channel	Input size	Operator	Exp size	#out	Attention	NL	s
3	256×192	Conv2d	—	16	—	HS	2
16	128×96	bneck, 3×3	16	16	—	RE	1
16	128×96	bneck, 3×3	64	24	—	RE	2
24	64×48	bneck, 3×3	72	24	—	RE	1
24	64×48	bneck, 5×5	72	40	SE	RE	2
40	32×24	bneck, 5×5	120	40	SE	RE	1
40	32×24	bneck, 5×5	120	40	SE	RE	1
40	32×24	bneck, 3×3	240	80	—	HS	2
80	16×12	bneck, 3×3	200	80	—	HS	1
80	16×12	bneck, 3×3	184	80	—	HS	1
80	16×12	bneck, 3×3	184	80	—	HS	1
80	16×12	bneck, 3×3	480	112	SE	HS	1
112	16×12	bneck, 3×3	672	112	SE	HS	1
112	16×12	bneck, 5×5	672	160	SE	HS	1
160	16×12	bneck, 5×5	960	160	SE	HS	1
160	16×12	bneck, 5×5	960	160	SE	HS	1
160	16×12	LPB, 4×4	320	160	SE	RE	2
160	32×24	LPB, 4×4	320	160	SE	RE	2
160	64×48	Conv2d 1×1	—	k	—	RE	1

by description of the proposed method. Then, we conduct experiments on the MSCOCO and MPII datasets and conclude this work.

2. Related Works

2.1. Human Posture Estimation. In recent years, deep learning-based pose estimation methods [16] have made great progress. Despite significant performance improvements, these prior works focused only on improving the accuracy of pose estimation by using complex networks and large tensors, while largely ignoring the cost issues of model inference. This state of affairs significantly limits their deploy ability in real-world applications, especially when the available computational budget is very limited.

In the literature, there are some recent works aimed at improving model efficiency. Bulat and Tzimiropoulos [10] designed a binary hourglass network using quantitative methods, but the restricted binary network has weak information representation and low accuracy of the model. Zhang et al. [7] proposed a new fast pose distillation (FPD) model learning strategy. A pretrained teacher network can be used to obtain a computationally fast and computationally inexpensive student network. However, it requires too much time to train. Yu et al. [17] proposed conditional channel weighting blocks and constructed the HR-Lite network, which achieves a great advantage in model accuracy and scale. However, the network structure is too complex, resulting in slow model inference. Zhang and Tang [11] proposed lightweight bottleneck block with depthwise convolution and attention mechanism, while the model size is still up to 2.7M parameters.

Compared with previous methods, comprehensively considering the accuracy of the model, the speed of inference, and the complexity of the model, we directly designed a model with simple structure and low complexity, which makes the model more practical and reliable in practical application scenarios.

2.2. Efficient Upsampling Module. Recent work [13, 18–20] has shown that deep convolutional neural networks have reached state-of-the-art performance. For advanced vision problems such as semantic segmentation [21], pose estimation [16], and object detection [22], existing approaches pass inputs through a network, usually consisting of high- to low-resolution subnetworks and a main network of raised resolutions. Many approaches have been designed to improve the resolution of the main network in different ways. For example, networks such as hourglass [6] reduce the input high-resolution features to low-resolution features and then use interpolation upsampling to scale the low-resolution features to the original input features, fuse the information with the previously input high-resolution features, and finally expect to generate fused high semantic and high resolution. Although it achieved very good results, the large tensor is used in the process of feature fusion. Zhou et al. [23, 24] constructed an attention-driven feature fusion upsampling network in an attempt to reduce the complexity of the model and reduce the use of large tensor using heterogeneous convolution. However, the network structure is complex and does not solve the problem of slow model inference fundamentally. SimpleBaseline [25] uses several transposed convolutional layers to generate high-resolution representations and achieves very good results. Although the model structure is simple, however, transposed convolution introduces a large number of parameters and computational effort, which is not friendly to small devices.

Therefore, we propose an efficient upsampling module that achieves a significant reduction in the number of

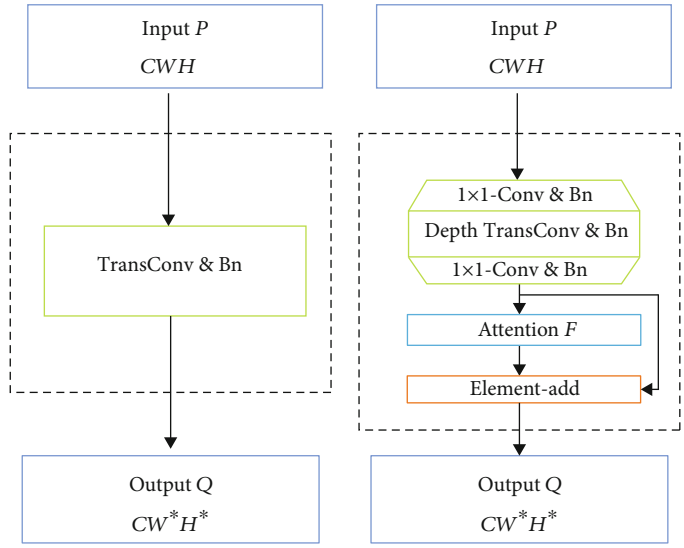

FIGURE 3: The comparison of two different upsampling methods. (a) The traditional transposed convolution, which has a large computational overhead. (b) The proposed lightweight upsampling block, which includes depthwise transposed convolution operation, pointwise convolution operation, and attention F. In (a), the features are amplified directly by transposed convolution. In (b), we first use 1×1 point convolution to expand the number of channels of the feature so that the number of channels goes from C to C^* and then use the depthwise transposed convolution to generate high-resolution feature maps. Finally, we use a 1×1 point convolution to change the number of channels to C and the attention mechanism to make feature map stronger.

parameters and computation during upsampling while ensuring the simplicity of the model structure and inference accuracy.

3. Proposed Method

In this section, we detail a simple and low computational cost human pose estimation network (MobilePoseNet), which designs a lightweight upsampling block (LPB) and directly uses high-resolution features to achieve high-resolution representation while maintaining lightweight features.

3.1. Lightweight Upsampling Block. Transposed convolution was first introduced into pose estimation by SimpleBaseline and achieved excellent performance. However, this operation brings a model with nearly a third of the parameters and calculations. Specifically, given the input of feature maps $C_{\text{in}} \times W_{\text{in}} \times H_{\text{in}}$ and the output of feature maps $C_{\text{out}} \times W_{\text{out}} \times H_{\text{out}}$, the amount of computation for conventional transposed convolution is

$$C_{\text{in}} \times C_{\text{out}} \times W_{\text{out}} \times H_{\text{out}} \times K \times K. \tag{1}$$

The number of parameters of traditional transpose convolution is

$$C_{\text{in}} \times C_{\text{out}} \times K \times K, \tag{2}$$

where K is the kernel size of traditional transposed convolution.

To reduce the burden of calculation and the number of parameters, while maintaining the effect of transpose convolution, we designed a lightweight upsampling block inspired by the intuition that the bottlenecks actually contain all the necessary information, as shown in Figure 3(b), which composed of three parts: depthwise transposed convolution, 1×1 point convolution, and attention module. Specifically, we first expand the low-latitude information to high-latitude information by 1×1 point convolution and use depth transpose convolution on each channel of the feature map for the spatial transformation. Finally, we use 1×1 point convolution to fuse the information between each channel and compress the high-latitude information to the original input latitude.

As shown in Figure 3(b), the computation of the lightweight upsampling block is the sum of the depth transpose convolution and the two point convolution computations:

$$\begin{aligned} C^* \times (&C_{\text{in}} \times W_{\text{in}} \times H_{\text{in}} \times 1 \times 1 + W_{\text{out}} \times H_{\text{out}} \\ &\times K \times K + C_{\text{out}} \times W_{\text{out}} \times H_{\text{out}} \times 1 \times 1). \end{aligned} \tag{3}$$

The number of parameters of the lightweight upsampling block is

$$C^* \times (K \times K + C_{\text{in}} + C_{\text{out}}), \tag{4}$$

where C^* is the number of channels for high-latitude features. Compared to the traditional transposed convolution, our method reduces the calculation amount to 83.2% and the number of parameters reduces to 74%.

Since LPB separates space operation and channel operation into two independent steps, the decoding effect of transpose convolution will be weakened. To solve this problem,

we enhance the feature responses through channel attention mechanism. Here, we directly use SENet [26] as our channel attention mechanism to dynamically adjust the weight of each channel, as shown in Figure 3(b). To sum up, we assumed the input feature map $X \in R^{C_{\text{in}} \times W_{\text{in}} \times H_{\text{in}}}$, the feature output through the LPB is $X' \in R^{C_{\text{out}} \times W_{\text{out}} \times H_{\text{out}}}$ as input to the channel attention mechanism.

The feature output through the channel attention mechanism is $X_{\text{att}} \in R^{C_{\text{out}} \times 1 \times 1}$. Then, the feature output of LPB and the feature output of the channel attention mechanism are multiplied and summed to obtain the final fusion information Y, i.e.,

$$Y = X' + X' X_{\text{att}}. \tag{5}$$

3.2. Lightweight Human Pose Estimation. Usually, the pipeline [5, 6, 27, 28] for poses estimation consists of three parts: the upsampling, the downsampling, and estimation of the heat map. In this work, we focus on the design of a lightweight upsampling and downsampling.

Different from SimpleBaseline which uses a ResNet backbone as the downsampling and three traditional deconvolutional layers as the upsampling, we use MobileNetV3 as our downsampling, which reduces the size of the parameters up to 96% and reduces the computation load up to 79%. For the upsampling, we replace each traditional deconvolution layer with a lightweight upsampling block. The details of the model are shown in Table 1.

As shown in Figure 2, different with SimpleBaseline, we use a higher resolution feature map as the input for upsampling. The rationale behind this that it is beneficial to maintain high-resolution representations before upsampling.

3.3. Parallel Pose NMS. In pose estimation, human body detectors inevitably generate redundant detection, and pose estimation also generates redundant poses. Therefore, nonmaximum suppression (NMS) is required to eliminate redundant postures.

Given pose P_i with m joints $\{<k_1^i, s_1^i>, <k_2^i, s_2^i>, \cdots, <k_m^i, s_m^i>\}$ where k_j^i and s_j^i are the location and confidence score of the j^{th} joint, respectively. Corresponding detection boxes b^i with b_s^i confidence score. The general pose NMS is as follows: firstly, the pose with the highest confidences was chosen as the reference, and the poses similar to it were suppressed or discarded. This process is repeated for the rest of the pose set until only one pose is left.

However, the main problems for this process are sequential and cannot be implemented in parallel, resulting in slower speeds. Inspired by Matrix-NMS, we proposed parallel nonmaximum suppression considering following two key factors:

(1) The confidence of pose: the higher the confidence of pose, the lower the probability of joints being suppressed, i.e., if the pose P_i and P_j with confidence $(p_i > p_j)$, P_j will have a high probability of being suppressed

(2) Similarity between the pose and other poses: the lower the similarity between one pose and other poses, the lower the suppression ratio of the poses

For the pose confidence, we set the product of the average of the confidence of the key points and the confidence of the human detector as the final pose confidence below

$$p_i = \left(\frac{\sum_m s_m^i \delta(s_m^i, \text{threshold})}{\sum_m \delta(s_m^i, \text{threshold})} \right) \cdot b_s^i, \tag{6}$$

where b_s^i is the confidence of the detection box and δ is defined as follows:

$$\delta(s, \text{threshold}) = \begin{cases} 1, s > \text{threshold}, \\ 0, s \le \text{threshold}. \end{cases} \tag{7}$$

We consider the key point prediction to be true if s_m^i is bigger than threshold and otherwise to be false.

For the similarity between two poses, we use the object key point similarity (OKS) [29] to measure the pose distance function as follows:

$$f(P_i, P_j) = 1 - O_{i,j}, \tag{8}$$

where $O_{i,j}$ is given by

$$O_{i,j} = \frac{\sum_m \exp\left\{ -\left(k_m^i - k_m^j\right)^2 / \left(2 b_a^i b_a^j\right) \right\}}{\sum_m \delta(s_m^i, \text{threshold}) \bullet \delta\left(s_m^j, \text{threshold}\right)}, \tag{9}$$

where b_a^i is the area of the detection box.

We define a new decay factor for pose NMS. For $f(P_i, P_j) = 1 - O_{i,j}$, we can get a new decay_j, where

$$\text{decay}_j = \min_{\forall p_i > p_j} \frac{f(P_i, P_j)}{f(:, P_i)}, \tag{10}$$

where $f(:, P_i) = \min_{\forall p_l > p_i} f(p_l, p_i)$.

Finally, we get a new pose confidence $p_j \longleftarrow \text{decay}_j \cdot p_j$. For usage, we just need threshold and selecting top-k scoring masks as the final predictions.

Like Matrix-NMS, all the operations in pose NMS could be implemented in one shot without recurrence. We first get a N pose confidence and then compute a $N \times N$ pairwise OKS matrix for the N pose sorted descending by pose confidence score. The decay factors of each pose can be obtained by looking up the table of the OKS matrix. Finally, the pose scores are updated by the decay factors. For usage, we just need threshold and select top-k pose scoring as the final predictions. The whole procedure is summarized in Algorithm 1.

Input: the area of the detection boxer $b_a = \{b_a^i\}$, the confidence of the detection boxer $b_s = \{b_s^i\}$, the location of the key point $K = \{k_j^i\}$, the confidence of the key point $S = \{s_j^i\}$, and parameter threshold. Here, i is the i-th person, $i \in \{1, 2, 3, \cdots, n\}$, j is the j-th key point, and $j \in \{1, 2, 3, \cdots, m\}$.
Output: the confidence of the key point $p = \{p^i | i = 1, 2, 3, \cdots n\}$.
1: Initialize threshold = 0.1
2: Calculate p by equation (6) and parameter b_s, S, threshold
3: Sort b_a, b_s, and K in descending order by $p = \{p^1, p^2, \cdots p^n\}$
4: Calculate B_+ using $b_s \times (b_s)^T$
5: Calculate K_+ using $(K + K^T)^2 - 4 \times (K + K^T)$
6: Calculate OKS matrix O by equation (9) and parameter K_+, B_+, threshold
7: Update $O = \{O_{i,j} = 1 | i \geq j, i, j = 1, 2, 3, \cdots n\}$
8: Set $o_m = \{\max_j(O_{:,j}) | j = 1, 2, 3, \cdots n\}$
9: Set O_m by repeating $o_m n$ times
10: Calculate decay matrix D using $(I - O)/(I - O_m)$
11: Set decay $o_d = \{\min(D_{:,j}) | j = 1, 2, 3, \cdots n\}$
12: Update the confidence of the key point p by $o_d \odot p$

Algorithm 1: Parallel pose NMS.

Table 2: Comparisons of results on the MSCOCO validation set.

Method	Backbone	Input	#Params	GFLOPs	AP	AP^{50}	AP^{75}	AP^M	AP^L	AR
8-stage hourglass [6]	Hourglass	256 × 192	25.6M	26.2	66.9	—	—	—	—	—
CPN [31]	ResNet-50	256 × 192	27.0M	6.2	68.4	—	—	—	—	—
SimpleBaseline [25]	ResNet-50	256 × 192	34.0M	8.9	70.4	88.6	78.3	67.1	77.2	76.3
HRNet-W32 [5]	ResNet-50	256 × 192	28.5M	12.4	**73.4**	**89.5**	**80.7**	**70.2**	**80.1**	**79.8**
DARK [32]	HRNetV1-W48	128 × 96	63.6M	3.6	71.9	89.1	79.6	69.2	78	77.9
MobileNetV2 [19]	MobileNetV2	256 × 192	9.6M	1.48	64.6	87.4	72.3	61.1	71.2	70.7
MobileNetV2 1×	MobileNetV2	384 × 288	9.6M	3.33	67.3	87.9	74.3	62.8	74.7	72.9
ShuffleNetV2 [33]	ShuffleNetV2	256 × 192	7.6M	1.28	59.9	85.4	66.3	56.6	66.2	66.4
ShuffleNetV2 1×	ShuffleNetV2	384 × 288	7.6M	2.87	63.6	86.5	70.5	59.5	70.7	69.7
Small HRNet	HRNet-W16	256 × 192	1.3M	0.54	55.2	83.7	62.4	52.3	61	62.1
Small HRNet	HRNet-W16	384 × 288	1.3M	1.21	56	83.8	63	52.4	62.6	62.6
Lite-HRNet	Lite-HRNet-18	256 × 192	1.1M	0.20	64.8	86.7	73	62.1	70.5	71.2
Lite-HRNet	Lite-HRNet-18	384 × 288	1.1M	0.45	67.6	87.8	75	64.5	73.7	73.7
MobilePoseNet	MobilNetV3	256 × 192	1.5M	0.55	66.2	87.3	74.2	63.1	72.5	72.4
MobilePoseNet	MobilNetV3	384 × 288	1.5M	1.23	69	88.2	75.9	65.5	75.5	74.9

4. Experiments

We conduct experiments on the MSCOCO and MPII datasets to evaluate the performance of our method in multiperson pose estimation.

4.1. Datasets

(i) The MSCOCO dataset contains over 200K images, 250K human body instances, and 17 key points. We trained our model on the MSCOCO train2017 dataset, including 57K images and 150K person instances and evaluated our approach on val2017 and test-dev2017, which contained 5000 images and 20K images, respectively

(ii) The MPII Human Pose dataset contains about 25K images of more than 40,000 people with annotated human joints, which are taken from a wide range of real-world activities with full-body pose annotations

We selected the object key point similarity (OKS) as an evaluation metric for the MSCOCO dataset. The standard metric [30], the PCK (probability of correct key point normalized by head) score, was used to evaluate the MPII dataset.

4.2. Implement Details. In MSCOCO, we extend the human detection box into a fixed aspect ratio with 4 : 3, and crop the box from the image with fixed size, 256 × 192 or 384 × 288. In MPII, the input size is cropped to 256 × 256 for fair comparison with other methods. In addition, the same data

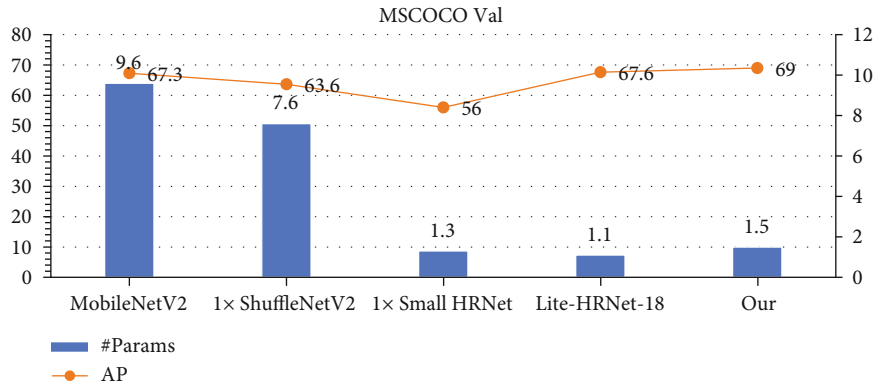

Figure 4: The complexity and accuracy comparison of MSCOCO val for 384 × 288 input size.

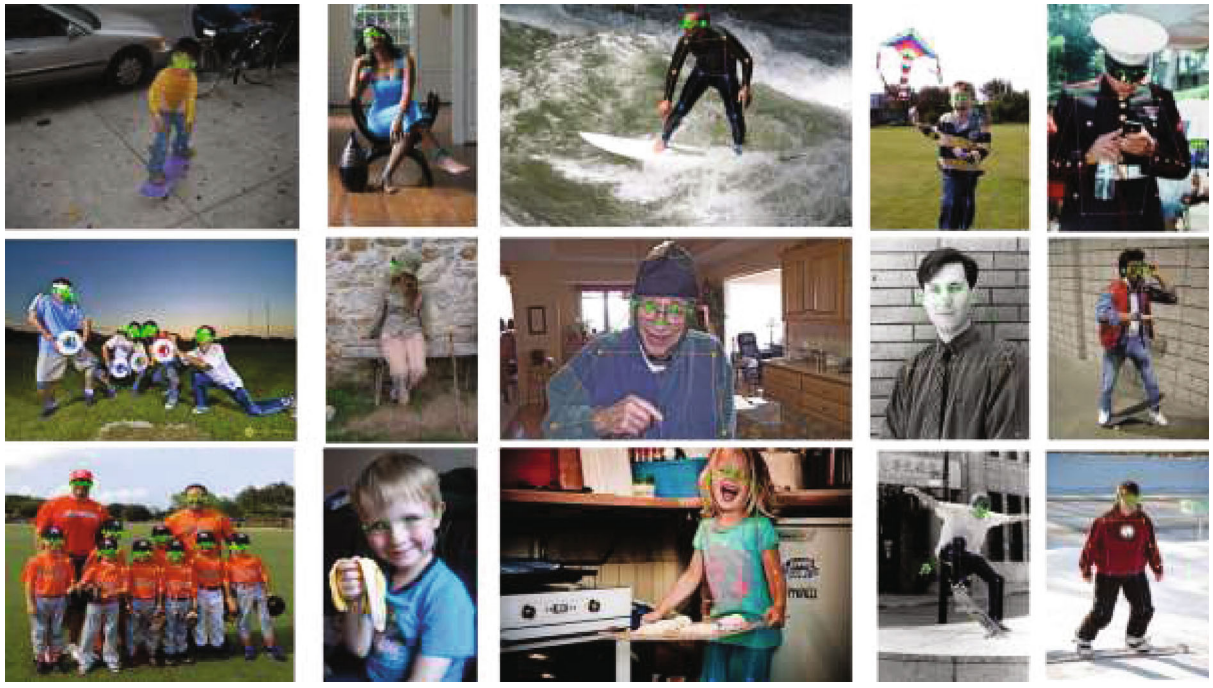

Figure 5: Visualization results of human pose estimation based on MobilePoseNet for MSCOCO validation set.

augmentation and the training strategy are utilized for both datasets. The data augmentation includes random rotation ([-45, 45]), random scale ([0.65, 1.35]), and flipping. In MSCOCO, half body data augmentation is also involved.

We all use the Adam optimizer with initial learning rate $1e-3$. The model was trained on a single Nvidia TITAN RTX GPU with a minibatch size 32 and stop at 210 epochs.

4.3. Experimental Results

4.3.1. Results on MSCOCO Dataset. From the results, as shown in Table 2, we can see that our method has a significant advantage in terms of model size and complexity with comparable accuracy. For input size 256 × 192, our method achieved comparable accuracy with less than 6% the parameters with respect to hourglass network. Compared with MobileNetV2 and ShuffleNetV2, our method obtained better accuracy with low complexity. For the small network HRNet-W16 and Lite-HRNet-18, our model is also better in terms of accuracy although the model size is slightly large. For the input 382 × 288, we can also derive the same conclusion.

Figure 4 illustrates the comparison of accuracy and complexity of small networks. Figure 5 shows the visualization results of our method in MSCOCO. It can be seen that our model achieved better balance between complexity and accuracy and can estimate the accurate joints under different complex scenes.

Table 3 lists the mAP, input size, Params, and GFLOP values of compared methods and our method on the MSCOCO dataset.

4.3.2. Results on MPII Human Pose Dataset. Table 4 reports the results of our network and other lightweight networks on MPII val data. Compared with MobileNetV2, MobileNetV3,

Table 3: Comparisons of results on MSCOCO test-dev2017 set. #Params and flops are calculated for the pose estimation network, and those for human detection are not included.

Method	Backbone	Input	#Params	GFLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
Bottom-up: key point detection and grouping										
OpenPose [34]	—	—	—	—	61.8	84.9	67.5	57.1	68.2	66.5
Associative embedding [35]	—	—	—	—	65.5	86.6	72.3	60.6	72.6	70.2
PersonLab [4]	—	—	—	—	68.7	89	75.4	64.1	75.5	75.4
MultiPoseNet [36]	—	—	—	—	69.6	86.3	76.6	65.0	76.3	73.5
HigherHRNet [37]	HRNet-w32	512×512	28.6M	47.9	66.4	87.5	72.8	61.2	74.2	—
Top-down: human detection and single-person key point detection										
Large network										
Mask-RCNN [22]	ResNet-50-FPN	—	—	—	63.1	87.3	68.7	57.8	71.4	—
G-RMI [15]	ResNet-101	353×257	42.6M	57	64.9	85.5	71.3	62.3	70.0	69.7
IPR [27]	ResNet-101	256×256	45.0M	11	67.8	88.2	74.8	63.9	74.0	—
RMPE [38]	PyraNet [39]	320×256	28.1M	26.7	72.3	89.2	79.1	68.0	78.6	—
CPN [28]	—	384×288	—	—	72.1	91.4	80.0	68.7	77.2	78.5
SimpleBaseline [25]	ResNet-152	384×288	68.6M	35.6	73.7	91.9	81.1	70.3	80.0	79.0
Small network										
MobileNetV2 [19]	MobileNetV2	384×288	9.8M	3.33	66.8	90.0	74.0	62.6	73.3	72.3
ShuffleNetV2 [33]	ShuffleNetV2	384×288	7.6M	2.87	62.9	88.5	69.4	58.9	69.3	68.9
Small HRNet [17]	HRNet-W16	384×288	1.3M	1.21	55.2	85.8	61.4	51.7	61.2	61.5
Lite-HRNet [17]	Lite-HRNet-18	384×288	1.1M	0.45	66.9	89.4	74.4	64.0	72.2	72.6
MobilePoseNet	MobileNetv3 [13]	256×192	1.5M	0.55	64.8	88.8	72.4	61.9	70.2	70.7
MobilePoseNet	MobileNetv3	384×288	1.5M	1.23	67.4	89.4	74.2	64.1	73.3	73.3

Table 4: Comparisons on the MPII val set. The GFLOPs is computed with the input size 256×256.

Model	#Params	GFLOPs	PCKh
MobileNetV2 1×	9.6M	1.97	85.4
MobileNetV3 1×	8.7M	1.82	84.3
ShuffleNetV2 1×	7.6M	1.70	82.8
Small HRNet-W16	1.3M	0.72	80.2
Lite-HRNet-18	**1.1**M	**0.27**	86.1
MobilePoseNet	1.5M	0.74	**87.3**

ShuffleNetV2, and SmallHRNet-W16, our model achieves better accuracy with lower number of parameters and calculation weights. Compared to Lite-HRNet-30, our model achieves 87.3 PCKh@0.5 in terms of the number of parameters with 0.3M less than Lite-HRNet-30. Compared to MobileNetV2, MobileNetV3, ShuffleNetV2, and Small HRNet-W16, our model improved by 1.9%, 3.0%, 4.5%, and 7.1%, respectively. Figure 6 illustrates the comparison of accuracy and complexity.

4.4. Inference Speed. FLOPs and Param are only the properties that measure the size and complexity of the model. In this section, we study the actual inference speed of the human pose estimation network by inference items per second (Inference Items Per Second). The speed is tested on devices with GPU and without GPU, respectively, with a batch size of 32 and full precision (fp32). We use the Nvidia TITAN TRX as the GPU device and the Intel Core I9-10900k device without GPU as the non-GPU device. To better reflect the running speed of the model, all methods are tested on the MSCOCO validation set. We use the same person detector provided by the SimpleBaseline validation set. In the tests without GPU, a thread was used for evaluation. As can be seen in Table 5, thanks to the simple structure of our model, our actual inference is 3 times faster than the less computationally intensive Lite-HRNet on the GPU speed test. In the GPU-free speed test, our method is faster than a large network like HRNet. Also, our model has a significant advantage in complexity and computational power compared to other models, which means easier deployment to embedded devices.

5. Ablation Study

We study the effect of each component of our approach on the validation set of MSCOCO.

5.1. Deconvolution Blocks. In this section, we analyzed the impact of reducing the number of upsampling and using different upsampling blocks in terms of accuracy with resolution 384×288. From Table 6, it can be seen that the number of parameters and the computation of our model

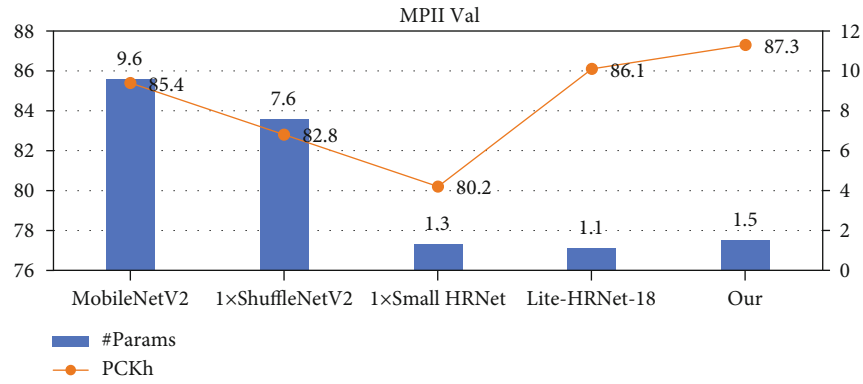

Figure 6: The complexity and accuracy comparison of MPII val sets for 256×256 input size.

Table 5: Inference speed comparisons on the MSCOCO validation set. Speed* refers to the result on non-GPU device. Speed refers to the result on GPU device. Bold values are the optimal results.

Method	Backbone	#Params	GFLOPs	Input size	AP	Speed*	Speed
HRNet	HRNetV1-W32	28.5M	7.1	256×192	74.4	7.5	19.2
HRNet	HRNetV1-W32	28.5M	16	384×288	**75.8**	4	18.8
SimpleBaseline	ResNet-50	34.0M	8.9	256×192	70.4	8.1	**273.1**
NLite-HRNet-18	HRNet-W16	0.7M	0.19	256×192	62.8	11	18.9
WNLite-HRNet-18	HRNet-W16	1.3M	0.3	256×192	66	12	18.6
ShuffleNetV2 1×	ShuffleNetV2	7.6M	1.28	256×192	59.9	**17**	71.3
ShuffleNetV2 1×	ShuffleNetV2	7.6M	2.87	384×288	63.6	10	64.1
MobileNetV2 1×	MobileNetV2	9.6M	1.48	256×192	64.6	6.8	83.1
MobileNetV2 1×	MobileNetV2	9.6M	3.33	384×288	67.3	4.5	73.1
Lite-HRNet	Lite-HRNet-18	1.1M	0.2	256×192	64.8	12	17.4
Lite-HRNet	Lite-HRNet-18	1.1M	0.45	384×288	67.6	7.1	16.3
MobilePoseNet	MobileNetV3	1.5M	0.55	256×192	66.2	7.8	54.8
MobilePoseNet	MobileNetV3	1.5M	1.23	384×288	69.0	5.1	50.8

Table 6: Ablation experiments on reduced downsampling with the use of lightweight upsampling blocks, on the MSCOCO val dataset. V1 denote the model that uses C5 as the input for upsampling, using the first 16 layers of MobileNetV3 as the downsampling and three layers of deconvolution as the upsampling part. V2 denote the model that uses C4 as the input for upsampling, using the first 13 layers of MobileNetV3 as the downsampling part, then uses three layers of 5×5 bottleneck with a stride of 1, and finally uses two layers of the same deconvolution as V1 as the upsampling part.

Model	Input size	FLOPs	#Params	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
V1	256×192	604M	2.5M	65.22	87.05	73.05	62.2	71.47	71.45
V2	256×192	684M	2.1M	66.23	87.23	74.19	63.21	72.48	72.38
V1	384×288	1.33G	2.5M	68.44	87.71	75.41	64.89	75.01	74.47
V2	384×288	1.5G	2.1M	68.97	87.72	75.47	65.32	**75.7**	74.85
Ours	256×192	**557**M	1.5M	66.23	87.38	74.25	63.13	72.52	72.4
Ours	384×288	1.23G	1.5M	**69.03**	**87.72**	**75.95**	**65.52**	75.55	**74.98**

are reduced compared to other models, while the precision has indeed been improved.

5.2. OKS-Based Nonmaximum Suppression. We compared the proposed OKS-based nonmaximum suppression and other OKS-based nonmaximum suppression methods on the accuracy and speed with the same pose estimator. As shown in Table 7, we can find that our proposed OKS-based nonextreme suppression has significant advantages in terms of accuracy and speed.

Table 7: Performance (AP) and speed (ms) of the MSCOCO validation set for different pose NMS. Matrix-OKS-NMS outperforms both Hard- and Soft-OKS-NMS methods in terms of speed and accuracy. Input size is 384×288.

Method	AP	AP^{50}	AR	Time (ms)
Hard-OKS-NMS	67.7	**87.1**	91.4	6.14
Soft-OKS-NMS	67.7	86.7	**91.5**	6.8
Parallel-OKS-NMS	**67.8**	86.7	**91.5**	**6.0**

6. Conclusion

In this paper, we propose a lightweight pose estimation network, which can achieve an AP score of 69.0 on the MSCOCO val set with only 1.5M parameters and 1.23 GFLOPs. However, we found that our model has some gaps compared to high-performance algorithms, mainly because we are missing the fusion of multiscale information. Designing complex networks and introducing the fusion of multiscale information will increase the inference speed of the model. In future work, we will redesign the backbone network for human pose estimation by introducing multiscale information to balance accuracy and speed.

Data Availability

The datasets used in this paper are the public datasets MSCOCO and MPII.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 61962010 and 61976107), the Excellent Young Scientific and Technological Talent of Guizhou Province ([2019]-5670), the Natural Science Foundation of Guizhou Province (Grant No. [2017]5726-32), the National Natural Science Foundation (No. 61863006), and the Basic Research Project (Key Project) of Guizhou Province ([2019]-1416).

References

[1] J. Shotton, A. Fitzgibbon, M. Cook et al., "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition 2011*, pp. 1297–1304, Colorado Springs, USA, 2011.

[2] N.-G. Cho, A. L. Yuille, and S.-W. Lee, "Adaptive occlusion state estimation for human pose tracking under self- occlusions," *Pattern Recognition*, vol. 46, no. 3, pp. 649–661, 2013.

[3] G. Cheron, I. Laptev, and C. Schmid, "P-CNN: pose-based CNN features for action recognition," in *International Conference on Computer Vision*, pp. 3218–3226, Santiago, Chile, 2015.

[4] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy, "PersonLab: person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model," in *European Conference on Computer Vision*, pp. 282–299, Munich, Germany, 2018.

[5] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5693–5703, Long Beach, USA, 2019.

[6] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European Conference on Computer Vision*, pp. 483–499, Glasgow, United Kingdom, 2016.

[7] F. Zhang, X. Zhu, and M. Ye, "Fast human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3517–3526, Long Beach, USA, 2019.

[8] Y. Chen, Y. Tian, and M. He, "Monocular human pose estimation: a survey of deep learning-based methods," *computer vision and image understanding*, vol. 192, article 102897, 2020.

[9] W. Li, Z. Wang, B. Yin et al., "Rethinking on multi-stage networks for human pose estimation," 2019, https://arxiv.org/abs/1901.00148.

[10] A. Bulat and G. Tzimiropoulos, "Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 3726–3734, Venice, Italy, 2017.

[11] Z. Zhang, J. Tang, and G. Wu, "Simple and lightweight human pose estimation.," https://arxiv.org/abs/1911.10346.

[12] M. Ding, X. Lian, L. Yang et al., "HR-NAS: searching efficient high-resolution neural architectures with lightweight transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2982–2992, 2021.

[13] A. Howard, R. Pang, H. Adam et al., "Searching for MobileNetV3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1314–1324, Seoul, Korea (south), 2019.

[14] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "SOLOv2: dynamic and fast instance segmentation," pp. 17721–17732, 2020, https://arxiv.org/abs/2003.10152.

[15] G. Papandreou, T. Zhu, N. Kanazawa et al., "Towards accurate multi-person pose estimation in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3711–3719, Hawaii, USA, 2017.

[16] A. Toshev and C. Szegedy, "DeepPose: human pose estimation via deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1653–1660, Columbus, USA, 2014.

[17] C. Yu, B. Xiao, C. Gao et al., "Lite-HRNet: a lightweight high-resolution network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10440–10450, 2021.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),*, pp. 770–778, Las Vegas, USA, 2016.

[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, Salt Lake City, USA, 2018.

[20] W. Lu, R. Yu, S. Wang, C. Wang, P. Jian, and H. Huang, "Sentence semantic matching based on 3D CNN for human–robot language interaction," *ACM Transactions on Internet Technology (TOIT)*, vol. 21, no. 4, pp. 1–24, 2021.

[21] Q. Zhou, X. Wu, S. Zhang, B. Kang, Z. Ge, and L. Jan Latecki, "Contextual ensemble network for semantic segmentation," *Pattern Recognition*, vol. 122, article 108290, 2022.

[22] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, Venice,Italy, 2017.

[23] Q. Zhou, Y. Wang, Y. Fan et al., "AGLNet: towards real-time semantic segmentation of self-driving images via attention-guided lightweight network," *applied soft computing*, vol. 96, p. 106682, 2020.

[24] Q. Zhou, Y. Wang, J. Liu, X. Jin, and L. J. Latecki, "An open-source project for real-time image semantic segmentation," *SCIENCE CHINA Information Sciences*, vol. 62, no. 12, article 227101, 2019.

[25] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 472–487, Munich,-Germany, 2018.

[26] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2011–2023, Salt Lake City, USA, 2018.

[27] X. Sun, B. Xiao, F. Wei, S. Liang, and Y. Wei, "Integral human pose regression," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 536–553, Munich,Germany, 2018.

[28] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, "Cascaded pyramid network for multi-person pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7103–7112, Salt Lake City, USA, 2018.

[29] T.-Y. Lin, M. Maire, S. J. Belongie et al., "Microsoft COCO: common objects in context," in *European Conference on Computer Vision*, pp. 740–755, Zurich, Switzerland, 2014.

[30] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, "2D human pose estimation: new benchmark and state of the art analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3686–3693, Columbus, USA, 2014.

[31] S. Huang, M. Gong, and D. Tao, "A coarse-fine network for keypoint localization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 3047–3056, Venice, Italy, 2017.

[32] F. Zhang, X. Zhu, H. Dai, M. Ye, and C. Zhu, "Distribution-aware coordinate representation for human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7093–7102, Seattle, USA, 2020.

[33] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: practical guidelines for efficient CNN architecture design," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 122–138, Munich,Germany, 2018.

[34] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using part affinity fields," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1302–1310, Hawaii, USA, 2017.

[35] A. Newell, Z. Huang, and J. Deng, "Associative embedding: end-to-end learning for joint detection and grouping," *Advances in neural information processing systems*, vol. 30, pp. 2278–2288, 2017.

[36] M. Kocabas, S. Karagoz, and E. Akbas, "MultiPoseNet: fast multi-person pose estimation using pose residual network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 437–453, Munich,Germany, 2018.

[37] B. Cheng, B. Xiao, J. Wang, H. Shi, T. S. Huang, and L. Zhang, "HigherHRNet: scale-aware representation learning for bottom-up human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5386–5395, Seattle, USA, 2020.

[38] H.-S. Fang, S. Xie, Y.-W. Tai, and C. Lu, "RMPE: regional multi-person pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2353–2362, Venice,Italy, 2017.

[39] W. Yang, S. Li, W. Ouyang, H. Li, and X. Wang, "Learning feature pyramids for human pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1290–1299, Venice,Italy, 2017.