Journal of Network Intelligence ©2024 ISSN 2414-8105
Taiwan Ubiquitous Information Volume **9**, Number **3**, August **2024**

Lite CSW-HRNet: Lightweight High-Resolution Human Pose Estimation Based on Channel Spatial Weighting

Yang Xi*

School of Computer Science
Northeast Electric Power University, Jilin 132012, P. R. China
474465389@qq.com

Zi-Hao Zhang

School of Computer Science
Northeast Electric Power University, Jilin 132012, P. R. China
531999659@qq.com

Si-Yu Meng

Yongji Power Supply Company
State Grid Jilin Electric Power Co., Ltd, Jilin 132012, P. R. China
771742082@qq.com

Jia Fu

Yongji Power Supply Company
State Grid Jilin Electric Power Co., Ltd, Jilin 132012, P. R. China
809818269@qq.com

Zhen-Yu Wu

Department of Orthopedics of Affiliated Hospital of Beihua University
Beihua University, Jilin 132012, P. R. China
myemail780216@sina.com

Wen-Jing Wang

School of Computer Science
Northeast Electric Power University, Jilin 132012, P. R. China
1418252418@qq.com

*Corresponding author: Yang Xi
Received August 31, 2023, revised December 15, 2023, accepted February 22, 2024.

Abstract. *Human pose estimation is widely used in virtual reality, medical diagnosis, video surveillance, etc. However, the high computational cost restricts its application in some terminals, so the research of lightweight models is particularly important. To address the problem of inadequate learning of Lite-HRNet weight coefficients, this paper proposes Lite CSW-HRNet, which independently computes weight maps in parallel along channels and in space during single-resolution and cross-resolution weight computation, respectively, and fully preserves the original features using max pooling and average pooling in the computation process; Adaptive 1D convolution is introduced in the channel weight calculation to aggregate information between adjacent channels, avoiding the adverse effects of channel degradation. In the spatial weight calculation, a 7×7 convolution is used to increase the perceptual field to aggregate a wider range of spatial contextual information. Comparative experiments on the COCO2017 dataset show that compared with Lite-HRNet, Lite CSW-HRNet further improves accuracy with decreasing both Params and FLOPs, and outperforms the state-of-the-art MobileNet and ShuffleNet in all metrics; compared with the large model HRNet, Params and FLOPs are about 1/30 of it, and AP can reach 90% of it, achieving a better balance result between accuracy and complexity of human pose estimation.*

Keywords: deep learning, human pose estimation, channel space weighting, high-resolution network, lightweight network

1. Introduction.

The goal of human pose estimation (HPE) is to determine from an input image the coordinates of important human skeletal points. The domains of action recognition [1], human-computer interaction [2], and medical rehabilitation have all benefited greatly from the rapid development and widespread usage of human pose estimation technologies in recent years. One of the most difficult and active areas of research in computer vision is this one. The task of attitude estimation cannot be limited to the improvement of recognition accuracy only, nowadays, there are more and more various terminals in life, and due to the limitation of size and cost, it is difficult to deploy the complex network, and how the network can become lighter and still guarantee the recognition accuracy has become a big trend in this research field. Yu et al. [3] proposed Lite-HRNet by introducing conditional channel weighting units into HRNet. the introduction of conditional channel weighting units and the network architecture that always maintains high resolution achieves the best results so far for the position-sensitive human pose estimation task. The model is lighter and more accurate than the mainstream networks. Although Lite-HRNet has achieved a good balance result between accuracy and complexity, there is still the problem of inadequate learning of weight coefficients, which is analyzed to be caused by the following reasons:

(1) The importance of key points of human skeleton in spatial location is not considered.

(2) A significant portion of features are lost as a result of average pooling, which severely restricts feature diversity.

(3) After pool, the fully connected layer downscales and upscales the channels, which has a negative impact on learning the dependencies between channels.

We rethink the weight learning process in order to address the issue of insufficient learning of weight coefficients in the Lite-HRNet network, and we propose the lightweight, high-resolution Lite-CSW-HRNet network, which is based on channel spatial weighting. The weight map is independently and concurrently calculated along the channel and space in the processes of single-resolution weight calculation and cross-resolution weight calculation, and it is then multiplied by the input feature map for adaptive feature refinement. In the channel weight map calculation process, we use max pooling and average pooling to complement each other in parallel to fully express the original rich information of the feature and introduce adaptive 1D convolution. The same parallel max pooling and average pooling are used for the computation of the spatial weight map. In addition, a 7×7 convolution is used to broaden the sensory field and aggregate more spatial contextual data in order to capture the correlation of the spatial locations of key points in the human skeleton.

2. Related Work.

Numerous recent human pose estimation techniques, including ResNet (Residual Network) [4], Hourglass [5], HRNet (High-Resolution Net) [6], and GAN (Generating Adversarial Network) [7], are based on neural networks. To address the issue of network deterioration brought on by increased depth, ResNet incorporates residual modules and makes use of a jump-connected residual structure. Hourglass network can more effectively extract the target's multi-scale feature information by cascading numerous Hourglass modules. By connecting high-resolution and low-resolution sub-networks in simultaneously, HRNet repeats multi-scale fusion while retaining a high-resolution representation. Generating adversarial networks incorporates a priori information about the human skeleton to improve the correctness of skeletal keypoint prediction through adversarial training, which in turn improves the accuracy of estimating the location of various body parts. While all of these approaches enable detection with high accuracy, they are often implemented through complex network structures, which in turn generated numerous model parameters and a significant amount of computing work, requiring high costs both for training and mobile terminal deployment, and therefore, some research has shifted the focus to lightweighting. Xiao et al. [8] proposed the Simple Baselines using a combination of a backbone and Deconvolution Module of ResNet50, demonstrating that a simple network structure can also achieve desirable results. Zhang et al. [9] improved Simple Baselines to propose LPN (Lightweight Pose Network) with Depthwise separable convolution [10] and attention mechanism to design lightweight bottleneck block, and also proposed an iterative training strategy and B-soft-Argmax function, this network shows a large advantage in inference speed. Debnath et al. [11], inspired by hourglass networks, improved the detection accuracy by introducing a novel shunt architecture in the last two layers of MobileNetsV1 [12], which reduces the parameters of the model and mitigates overfitting. MobileNetV2 [13] introduces linear bottlenecks and inverse residuals on top of V1 to improve the network's characterization capability and further enhance performance. Ding et al. [14] updated the search space and search strategy of NAS (Neural Architecture Search) by using a multi-branch architecture to provide convolutional encoding of multiple feature resolutions and proposed HR-NAS, which achieved a better accuracy and complexity balance on three dense prediction tasks and classification tasks. Zhang et al. [15] designed differentiable neural network architecture search and spatial information correction modules and proposed the EfficientPose network to automate the design of backbone networks at a very low computational cost and to effectively address the checkerboard effect in prediction, which in turn improves the prediction accuracy. Yu et al. [3] proposed Naive Lite HRNet by replacing the second 3 × 3 convolution and all normal residual blocks in Small HRNet (HRNet-W16) using Shuffle blocks in ShuffleNetV2 [16]. To further optimize the network performance, conditional channel weighting unit is also introduced and Lite-HRNet is proposed.

3. Method.

3.1. Lite CSW-HRNet.

The network structure of Lite-CSW-HRNet (Figure 1, Table 1) still follows Lite-HRNet and has a different parallel structure. Firstly, the stage1 takes 1/4 of the input image resolution as the input of the network, and then the stage is formed by adding 1/2 of the lowest resolution of the previous stage as a new branch in parallel to the current stage, and fusing the resolutions with each other.

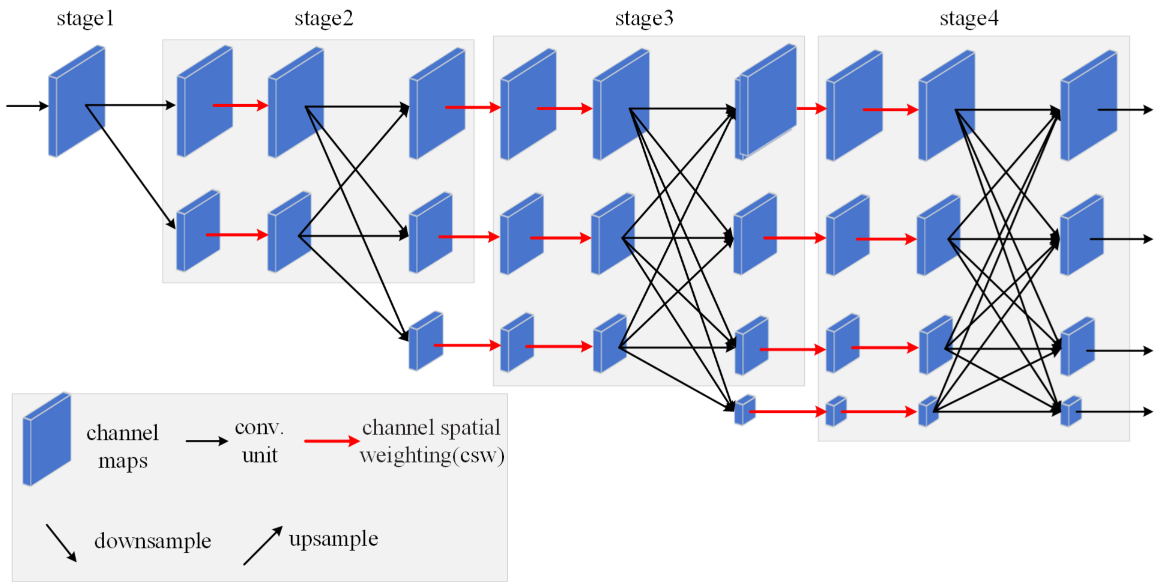

Figure 1. The structure of Lite CSW-HRNet.

Table 1. Composition of Lite CSW-HRNet. CSW is channel spatial weight.

layer	output	number of layers	operator	resolution branch	output channels	repeat
input	256×256			1×	3	
stage1	64×64	1	conv2d, shuffle block	2×, 4×	32	1
stage2	64×64	2	CSW block, fusing block	4×8×	40, 80	2
stage3	64×64	4	CSW block, fusing block	4×8×16×	40, 80, 160	2
stage4	64×64	2	CSW block, fusing block	4×8×16×32×	40, 80, 160, 320	1

In Lite CSW-HRNet, we call the original conditional channel weighting as channel spatial weighting. As shown in Figure 2, $S(\bullet)$ is the single-resolution weighting function, $C(\bullet)$ is the cross-resolution weighting function, and the dashed line indicates the information interaction between the cross-resolution. The weighting process for the s-th resolution branch is expressed as,

$$Y_s = W_s \otimes X_s \tag{1}$$

where $\otimes$ represents the primary element multiplication and W_s is a three-dimensional tensor of size $C_s \times W_s \times H_s$.

In order to communicate information simultaneously, we compute the weights using both the single-resolution channel space and the all-resolution channel space.

3.2. **Single-resolution weight computation.** Given a feature map $F \in \mathbb{R}^{C\times H\times W}$ as input, a 1D channel weight map $M_c \in \mathbb{R}^{C\times 1\times 1}$ and a 2D spatial weight map $M_s \in \mathbb{R}^{1\times H\times W}$ are computed according to (2), respectively.

$$\begin{aligned} F'_c &= M_c(F) \\ F'_s &= M_s(F) \end{aligned} \tag{2}$$

As shown in Figure 3, we connect these two weight computation modules in parallel to obtain the weight map separately, and then perform adaptive feature refinement on the feature map, and F'' is the final refinement output.

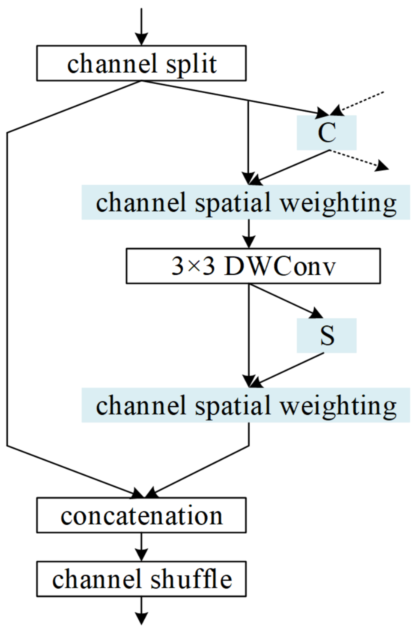

Figure 2. Channel Spatial Weighting block.

$$F'' = F'_c \otimes F'_s \otimes F \tag{3}$$

As a result, channel weight values are broadcast along the spatial dimension during multiplication and vice versa.

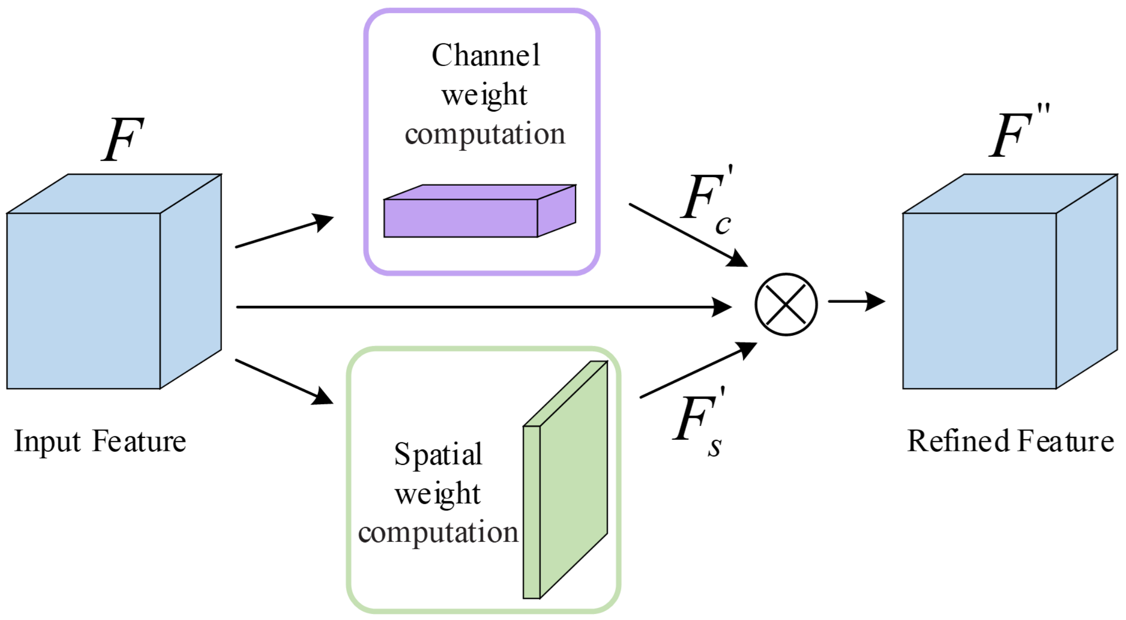

Figure 3. The process of Single-resolution weight computation.

The human skeletal keypoints are smaller than the global image targets, and compared with the channel weight computation and spatial weight computation in series, the parallel approach can extract the shallow features of the human skeletal keypoints more adequately, so that the input feature maps can be learned separately and the latter weight computation module will not be interfered by the previous weight computation module.

Channel weight computation. In the specific details of the channel weight computation implementation, firstly, average pooling and max pooling are applied to the input feature map $F \in \mathbb{R}^{C\times H\times W}$. Through experiments, we discovered that a single average pooling or max pooling significantly reduces the diversity of features for feature maps in various contexts, leading to a significant loss of features. Take the single-channel grayscale Figure 4 as an example, when the foreground image is darker than the background image, max pooling leads to a large amount of weakened features in the foreground image, and conversely when the foreground image is brighter than the background image, average pooling leads to a large amount of weakened features in the foreground image. As a result, we considerably enhance the network's ability to represent features in various contexts by using both average and maximum pooling. We also employ this approach in the computation of the spatial weights below.

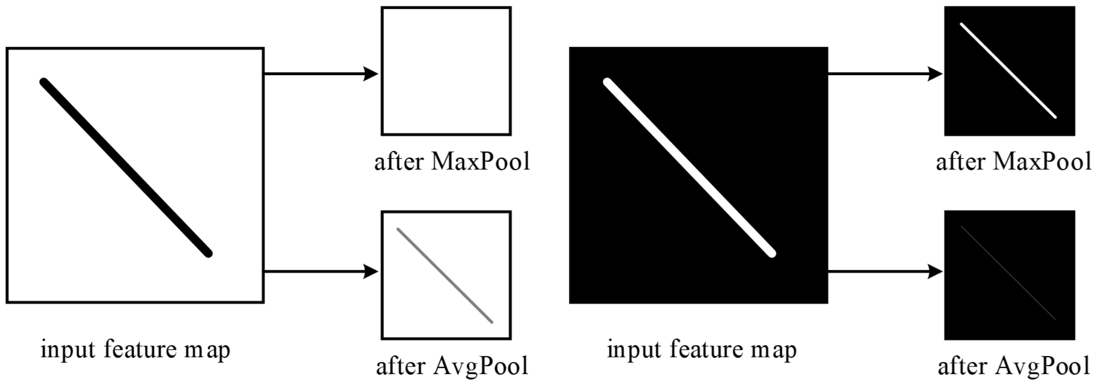

Figure 4. The distinction between maximum and average feature extraction pooling.

After pooling the feature maps, we borrowed the idea from ECA [17, 18, 19] that any given intermediate feature maps in CNNs have greater correlation between adjacent channels, and that mapping channel features using fully connected layers generates many redundant computations. Instead of using a fully connected layer, we include a 1D convolutional layer as part of our local cross-channel interaction method, while applying an adaptive channel dimension function to determine the size of the 1D convolutional kernel for more efficient training of the network, which improves accuracy while reducing the number of parameters. Only the interactions between each channel and its k surrounding channels are taken into account in order to capture the local cross-channel interactions, and the weights ω_i imposed on the channels are generated by (4), using the average pooling branch as an example,

$$\omega_i = \sigma\left(\sum_{j=1}^{k} w_i^j y_i^j\right), y_i^j \in \Omega_i^k \tag{4}$$

where Ω_i^k denotes the k neighboring channels of y_j', y_j' denotes the value of the feature map after pooling, and w_{ij} denotes the connection weights of the k neighboring channels to the next layer. At this point, there are a total of $k \times C$ parameters within the average pooling branch, and C stands for the input feature map's channel count. To further reduce the complexity of the network and improve efficiency, the parameters are shared among all connected channels, as shown in (5):

$$\omega_j = \sigma\left(\sum_{j=1}^{k} w_j' y_j'\right), y_j' \in \Omega_i^k \tag{5}$$

Compared with (4), w'_j becomes w_j. After the channel connection parameters are shared, fast 1D convolution with a kernel size of k can easily reach (5), when the number of parameters is only k.

$$w = \sigma\left(C1D_k\left(y\right)\right) \tag{6}$$

In (6), $C1D$ stands for 1D convolution, and k for the quantity of convolution kernels. In the meantime, the channel dimension C and k are associated. Between k and C, there might be a specific mapping relationship φ. k is nonlinearly proportional to C. As a result, using the exponential function to represent the mapping relationship between k and C is a viable option,

$$C = \varphi\left(k\right) \approx \exp\left(y \cdot k - b\right) \tag{7}$$

since the channel dimension C is often set to an integer power of 2, Equation (7) is changed to Equation (9) where y and b are two hyperparameters that are typically set to 1 and 2.

$$k = \psi\left(C\right) = \left\lfloor \frac{\log_2\left(C\right)}{y} + \frac{b}{y} \right\rfloor_{\text{odd}} \tag{8}$$

where $\lfloor t \rfloor_{\text{odd}}$ denotes the odd number closest to t. As a result, for a given channel dimension C, the convolution kernel size k can be computed adaptively in the manner described above.

In conclusion, we create a channel weight map by utilizing the inter-channel relationships of the features. First, we aggregate the spatial information of the feature mapping using average pooling and max pooling operations, creating two distinct channel descriptors, F^c_{avg} and F^c_{max}, which stand for average pooling features and max pooling features, respectively. The information from the nearby k channels is then combined using a 1D convolution with a convolution kernel of length k. After convolution, the two features are added by elements and the Sigmoid function is used to produce a channel weight map $M_c\left(F\right) \in \mathbb{R}^{C\times 1\times 1}$. In order to acquire the feature map after injecting the channel weights, the created channel weights are then multiplied by the input feature map by corresponding elements and then enlarged to $\mathbb{R}^{C\times H\times W}$ along two dimensions on the space. The channel weight calculating procedure can be specifically stated as follows:

$$M_c\left(F\right) = \sigma\left(f^k_{1D}\left(AvgPool\left(F\right)\right) + f^k_{1D}\left(MaxPool\left(F\right)\right)\right) \tag{9}$$

The one-dimensional convolution procedure with the convolution kernel size k is represented by f^k_{1D}, and σ stands in for the Sigmoid function. Equation (8) in [3] adaptively calculates size of k.

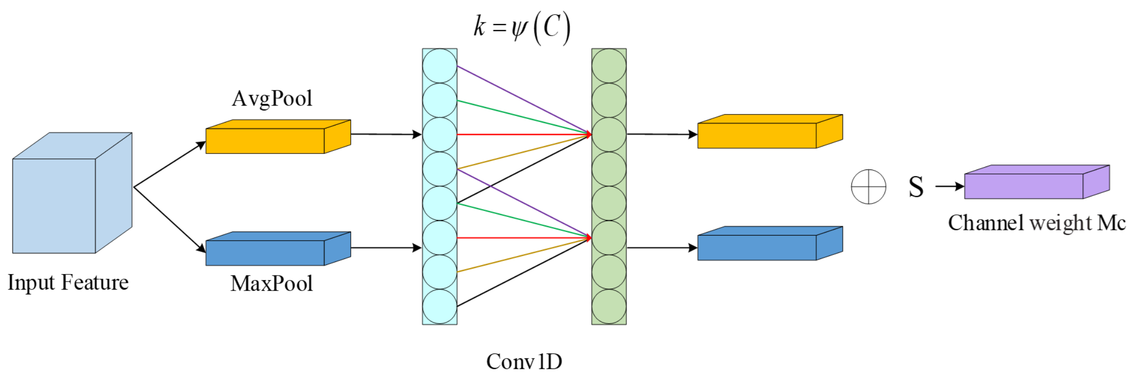

Figure 5. Channel weight computation module.

Spatial weight computation. As an addition to the channel weight calculation, the spatial weight computation first runs average pooling and max pooling operations along the input feature map's channel axis to produce the two separate spatial context descriptors F^s_{avg} and F^s_{max}. To create a functional spatial feature descriptor, the independently generated descriptors are stitched along the channel axis. In aggregating spatial information, we found that the association between key points of the human skeleton requires a larger perceptual field compared to other visual tasks.

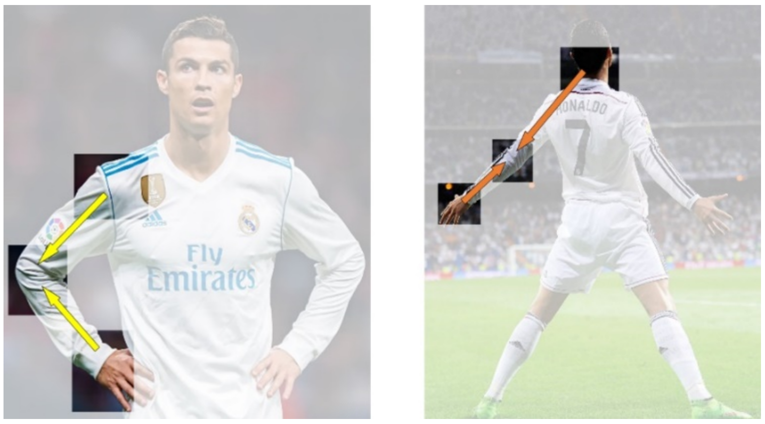

Figure 6. Association between key points.

For a small feature extractor whose receptive field can only cover the elbow joint itself, it is challenging to distinguish the elbow joint from the knee joint in the left panel of Figure 6 due to their comparable appearances. However, if the receptive field can also see the nearby wrist or shoulder, it is much easier to categorize it as an elbow. Similarly, in the figure on the right, to determine whether a part of the body is left or right, the orientation of the person's head and hands is important information, but this requires a larger receptive field. We choose to use 7×7 convolution to encode the mapping of information in the space that needs to be emphasized or suppressed region to aggregate the spatial context information more efficiently, and the convolved features are subjected to Sigmoid function operation to generate the spatial weight map $M_s(F) \in \mathbb{R}^{1\times H\times W}$. The created spatial weights are then broadcast extended to $\mathbb{R}^{C\times H\times W}$ along the channel dimension and multiplied by the corresponding elements of the input feature map to produce the feature map after injecting the weights. Specifically, the spatial weight computation process can be expressed as follows:

$$M_s(F) = \sigma\left(f^{7\times 7}\left([\mathrm{AvgPool}(F); \mathrm{MaxPool}(F)]\right)\right) = \sigma\left(f^{7\times 7}\left([\mathrm{F}^s_{\mathrm{avg}}; \mathrm{F}^s_{\mathrm{max}}]\right)\right) \qquad (10)$$

where $f^{7\times 7}$ denotes a 7×7 regular convolution.

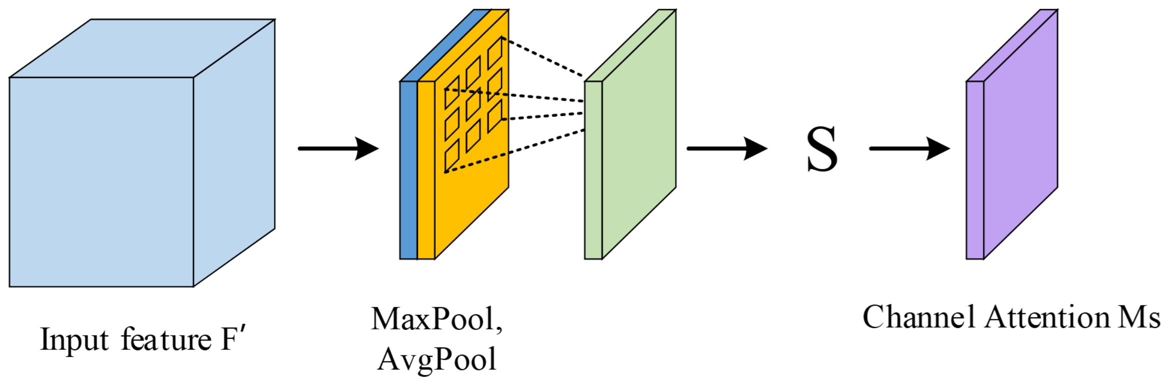

Figure 7. Spatial weight computation module.

In summary, Single-resolution weight was calculated using Equation (11):

$$w_s = S(X_s) \tag{11}$$

where X_s represent the single-resolution maps' input maps. The function $S(\cdot)$ is implemented as:

$$X_s \to M_c \to \otimes \to W_s X_s \to M_s \to \otimes \to W_s \tag{12}$$

The single-resolution weight computation process is shown in Figure 8.

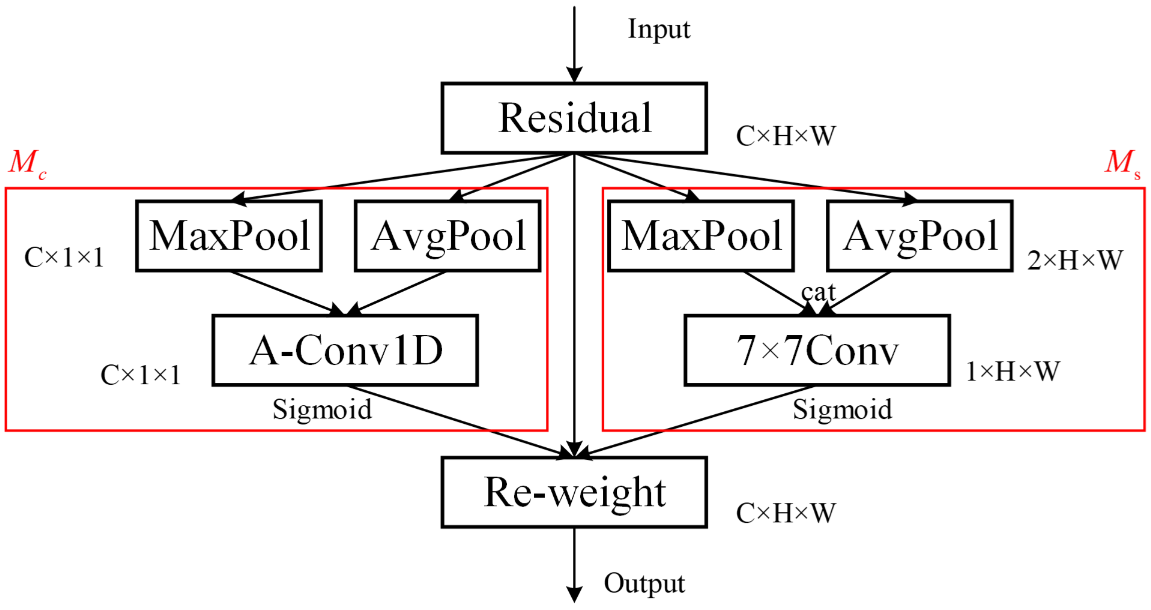

Figure 8. Single-resolution weight computation process.

3.3. **Cross-resolution weight computation.** Take into account that at the s-th stage, there are s parallel resolutions and s weight mappings $W_1, W_2, \cdots, W_s$, each of which corresponds to a certain resolution. Compute s weight mappings for all channels and spaces in different resolution subnets using the function $C(\cdot)$,

$$(W_1, W_2, \cdots, W_s) = C(X_1, X_2, \cdots, X_s) \tag{13}$$

where $\{X_1, X_2, \cdots, X_s\}$ are the input maps for the s resolution, X_1 is the highest resolution, and X_s is the s-th high resolution. The implementation of function $C(\cdot)$ is as follows: We use adaptive average pooling (AAP) to process $\{X_1, X_2, \cdots, X_{s-1}\}$:

$$\begin{cases} X_1' = AAP\left(X_1\right) \\ X_2' = AAP\left(X_2\right) \\ \vdots \\ X_{s-1}' = AAP\left(X_{s-1}\right) \end{cases} \tag{14}$$

where AAP combines any input size into a given output size $W_s \times H_s$, at this time each parallel resolution is the same size as the minimum resolution, we will $\{X_1', X_2', \cdots, X_{s-1}'\}$ and X_s stitch together, the subsequent computations are implemented on is small resolution, so its computational complexity is very small. For the subsequent operation, we directly use the single-resolution weighting method,

$$\begin{tikzcd} & M_c \arrow[dr] & & \\ \{X_1', X_2', \cdots, X_{s-1}', X_s\} \arrow[ur] \arrow[dr] & & \otimes \arrow[r] & \{W_1', W_2', \cdots, W_s'\} \\ & M_s \arrow[ur] & & \end{tikzcd} \tag{15}$$

to generate a weight map $\{W_1', W_2', \cdots, W_s'\}$ containing s branches, with weights at each location for each resolution depending on the channel and spatial features at the same location from the pooled multi-resolution feature map.

4. **Experiments and Analysis.**

We evaluated the Lite-CSW-HRNet on the human pose estimation dataset COCO2017 [20] and conducted comparison experiments with the original Lite-HRNet.

4.1. **Datasets.**

The experiments were conducted using the COCO2017 dataset for training, validation and testing, which is a more widely used benchmark dataset. the COCO2017 dataset is divided into the training dataset train2017 (57000 images and 15000 person instances), the validation dataset val2017 (5000 images) and the testing dataset test-dev2017 (20,000 images). There are 17 skeletal key points for each human body in the dataset, which are nose, right eye, left eye, right ear, left ear, right shoulder, right elbow, right wrist, left shoulder, left elbow, left wrist, right hip, right knee, right ankle, left hip, left knee and left ankle.

The sizes of the original images in the COCO2017 dataset are not uniform, so the images need to be pre-processed before training. In this experiment, two sizes of images are trained and compared for analysis, the image sizes are 256×192 and 384×288, and the whole preprocessing process is shown as follows:

Step1. The images of the dataset were cropped by centering on the hip of the main human body, re-cropping the image size to 256×192 or 384×288, and adjusting the human detection frame to a fixed aspect ratio of 4:3, so that the network could be trained.

Step2. A number of data augmentation procedures, including random rotation ([-30°,30°]), random scaling ([0.75,1.25]), random flip, and additional half body data augmentation, are carried out on the training images for certain partial human images in the COCO2017 dataset.

4.2. **Environment Configuration.**

The experiments were done on a server consisting of Ubuntu version 18.04 with Linux kernel, python 3.7, pytorch 1.8.1+cu101 and an NVIDIA Tesla T4 GPU. During the experiments, the Adam optimizer was chosen to optimize the model with a learning rate of 0.002 and a total training epoch of 210.

4.3. **Training and Testing.** The train2017 dataset was used to train the Lite CSW-HRNet, while the val2017 dataset and test-dev2017 dataset were used to validate and test it. Using a two-stage top-down methodology, we first detected humans using SimpleBaseline [8] person detector before moving on to keypoint identification.

4.4. **Evaluation.** We use AP (Average Precision), AR (Average Recall), Params, and FLOPs as the evaluation metrics of the network. Where AP and AR are based on OKS (Object Key Point Similarity), OKS is expressed as:

$$OKS = \frac{\sum_i \exp\left(-d_i^2/2s^2k_i^2\right)\delta(v_i > 0)}{\sum_i \delta(v_i > 0)} \tag{16}$$

where d_i is the distance in Euclid between the predicted keypoints and the actual value, s is the target scale factor, k_i is the decay constant connected to each keypoint, sk_i is the standard deviation of each keypoint, and $v_i \in \{0, 1, 2\}$ is the true keypoint visibility identifier, where 0 denotes unlabeled, 1 denotes labeled but not visible, and 2 denotes labeled and visible. The range $[0, 1]$ encompasses the precision of keypoint detection, with a bigger OKS indicating a more precise spatial positioning of keypoints.

AP denotes the average accuracy at T_{OKS} of 0.50, 0.55, 0.60,..., 0.90, 0.95 respectively.

The detection accuracy for $T_{OKS} = 0.50$ is indicated by AP^{50}, and at $T_{OKS} = 0.75$ by AP^{75}. AP^M stands for medium-scale target detection accuracy, AP^L for large-scale target detection accuracy, and AR for average recall at T_{OKS} of 0.50, 0.55, 0.60,..., 0.90, 0.95.

The entire number of parameters that must be taught for the network model is called params. One billion floating point operations per second are known as GFLOPs.

4.5. **Results.** To demonstrate the effectiveness of Lite-CSW-HRNet, we develop comparison experiments. Params and FLOPs do not include human detection and key point grouping.

The comparison between Lite-CSW-HRNet and several models using the COCO val2017 dataset is shown in Table 2. When the image input size of Lite CSW-HRNet is 256×192, it obtains 64.2% AP score, which is 0.8% better than Lite-HRNet, 0.5% better on AP^{50}, and 0.7% better on AP^{M}. Compared with the other three small models MobileNetV2, ShuffleNetV2, and Small HRNet, the evaluation metrics are similarly better than them. Although the accuracy of Lite CSW-HRNet is reduced compared with the large models SimpleBaseline and HRNetV1, the number of Params is about 1/30 of theirs and the FLOPs is about 1/40 of theirs, which is better to achieve the balance of the model indicators. When the input image size is 384×288, Lite CSW-HRNet improves 1% over Lite-HRNet in AP score to 67.2%, 0.2% in AP^{50}, and 1.2% in AP^{L} at the same time. It further shows that Lite CSW-HRNet can effectively improve the detection of small and medium scale targets without sacrificing the accuracy of large scale targets. Table 3 displays the comparison outcomes of the Lite CSW-HRNet with several models on the COCO test-dev2017. The comparison shows that when the input image size is 256×192, Lite CSW-HRNet obtains 63.7% AP score, which is 0.8% better than Lite-HRNet, while AP^{50} and AP^{75} are also improved. When the input image size is 384×288, each evaluation metric is equally better than the four smaller models. The effectiveness of Lite CSW-HRNet is further demonstrated by comparison experiments with two datasets and two different input sizes.

Table 2. Comparisons using the COCO val 2017 set.

Model	Backbone	Pretrain	Input size	#Params	GFLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
Large networks											
SimpleBaseline[8]	RestNet-50	Y	256×192	34.0M	8.90	70.4	88.6	78.3	67.1	77.2	76.3
HRNet[6]	HRNetV1-W32	N	256×192	28.5M	7.10	73.4	89.5	80.7	70.2	80.1	78.9
Small networks											
MobileNetV2	MobileNetV2	Y	256×192	9.6M	1.48	63.3	86.4	70.9	60.0	69.6	69.5
ShuffleNetV2[3]	ShuffleNetV2	Y	256×192	7.6M	1.28	59.9	85.4	66.5	56.2	66.2	66.4
Small HRNet[3]	HRNet-W16	N	256×192	1.3M	0.34	55.2	80.4	61.0	53.3	61.0	62.1
Lite-HRNet	Lite-HRNet-18	N	256×192	1.1M	0.28	64.1	86.1	71.3	60.9	71.4	69.9
Lite-HRNet	Lite-HRNet-18	N	384×288	1.1M	0.46	66.2	87.3	74.1	62.4	72.1	72.3
Lite CSW-HRNet	Lite-HRNet-18	N	256×192	0.9M	0.20	64.2	86.6	71.8	61.4	70.1	70.5
Lite CSW-HRNet	Lite-HRNet-18	N	384×288	0.9M	0.45	67.2	87.3	74.3	64.1	73.3	73.1

Table 3. Comparisons using the COCO test-dev2017 set.

Model	Backbone	Pretrain	Input size	#Params	GFLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
Large networks											
SimpleBaseline[8]	RestNet-152	Y	384×288	68.6M	35.60	73.7	91.9	81.8	70.3	80.9	79.0
HRNet[6]	HRNetV1-W32	N	384×288	28.5M	16.00	74.9	92.5	82.8	71.3	80.9	80.1
Small networks											
MobileNetV2	MobileNetV2	Y	384×288	9.6M	3.38	66.1	89.2	73.3	62.4	71.2	71.4
ShuffleNetV2[3]	ShuffleNetV2	Y	384×288	7.6M	2.87	62.9	86.4	69.4	58.9	69.3	68.9
Small HRNet[3]	HRNet-W16	N	384×288	1.3M	1.21	55.3	82.0	61.5	52.1	61.5	61.2
Lite-HRNet	Lite-HRNet-18	N	256×192	1.1M	0.28	64.1	86.4	71.3	60.9	71.4	69.9
Lite-HRNet	Lite-HRNet-18	N	384×288	1.1M	0.46	66.2	89.4	74.2	63.7	72.4	71.3
Lite CSW-HRNet	Lite-HRNet-18	N	256×192	0.9M	0.20	63.7	86.7	71.1	61.3	68.5	69.7
Lite CSW-HRNet	Lite-HRNet-18	N	384×288	0.9M	0.45	66.6	89.5	74.2	63.7	71.8	72.4

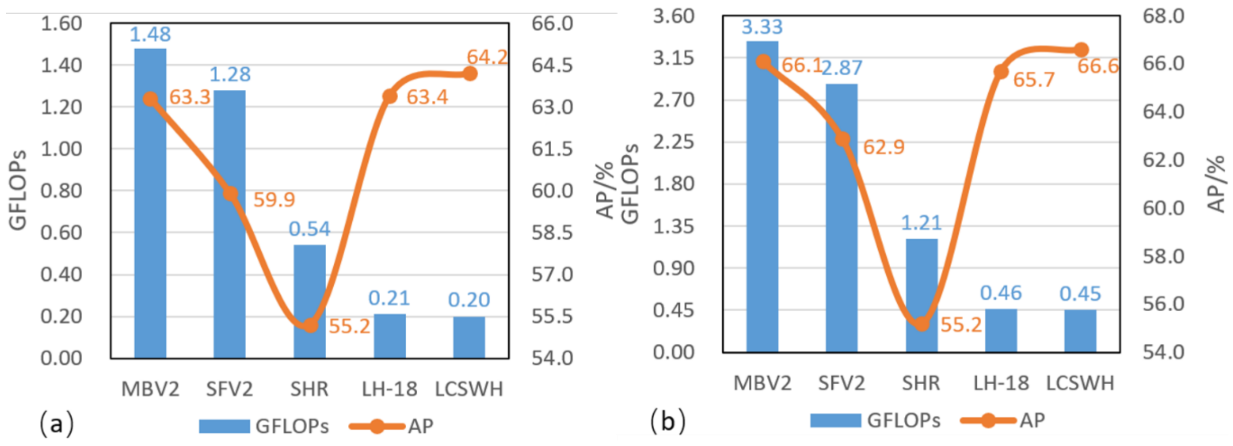

Figure 9. Comparison plots of Lite CSW-HRNet with small model FLOPs and AP. (a) Results of comparison in the COCO val2017 dataset with 256×192 input size. (b) Results of comparison on the COCO test-dev2017 dataset with a 384 x 288 input size.

Figure 10 shows the results of the single person pose estimation and heat map comparison between Lite CSW-HRNet and the other three small models without the human detector. Among them, (a) MobileNetV2 and (b) ShuffleNetV2 misidentified the left hand of the person; (c) Lite-HRNet could roughly identify each key point but with a large error; from the heat map, we can see that (d) Lite CSW-HRNet can accurately locate and identify each key point.

The non-normal human pose can also measure the merit of the model. In Figure 11, (a) MobileNetV2, (b) ShuffleNetV2 and (c) Lite-HRNet fail to identify the legs of the

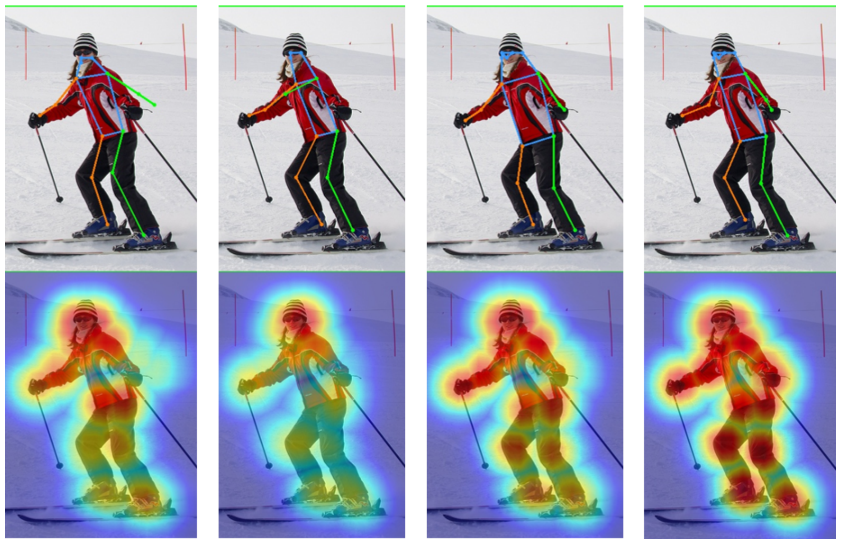

Figure 10. Comparison results of single person pose estimation and heat map without human detector. (a) MobileNetV2. (b) ShuffleNetV2. (c) Lite-HRNet. (d) Lite CSW-HRNet.

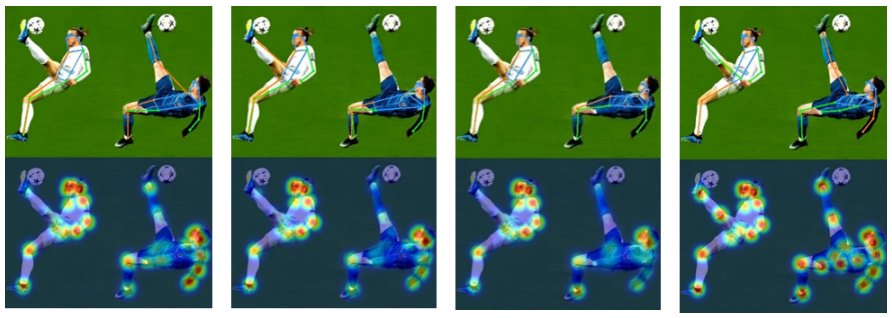

Figure 11. Comparison results of human pose estimation and heat map under human pose distortion using Faster-RCNN as a human detector. (a) MobileNetV2. (b) ShuffleNetV2. (c) Lite-HRNet. (d) Lite CSW-HRNet.

two backwards hooked soccer players. (a) MobileNetV2 and (b) ShuffleNetV2 incorrectly identify the legs as arms, compared to (d) Lite CSW-HRNet which identifies the characters completely and correctly.

Figure 12 shows the results of Lite CSW-HRNet using Faster RCNN as a human detector for multi-person pose estimation and heat map comparison with three other small models in the occlusion case. Among them, (a) MobileNetV2, (b) ShuffleNetV2 and (c) Lite-HRNet have poor recognition of the key points of the crouching person in the figure, and have different degrees of detection errors for the right leg of the standing person. From the heat map and the pose estimation results, we can see that (d) Lite CSW-HRNet can recognize each key point of the squatting person more accurately, and the right leg of the standing person is also recognized more accurately.

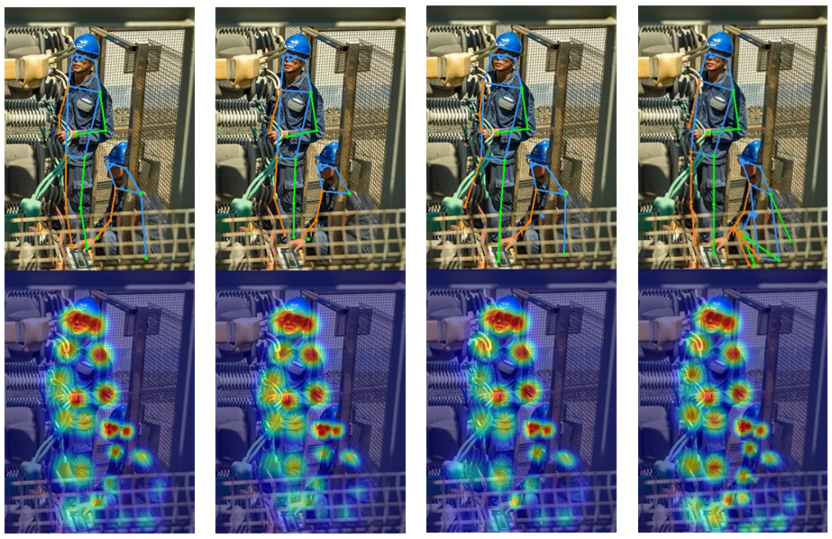

Figure 12. Comparison results of multi-person pose estimation and heat map in the occlusion case using Faster-RCNN as human detector. (a) MobileNetV2. (b) ShuffleNetV2. (c) Lite-HRNet. (d) Lite CSW-HRNet.

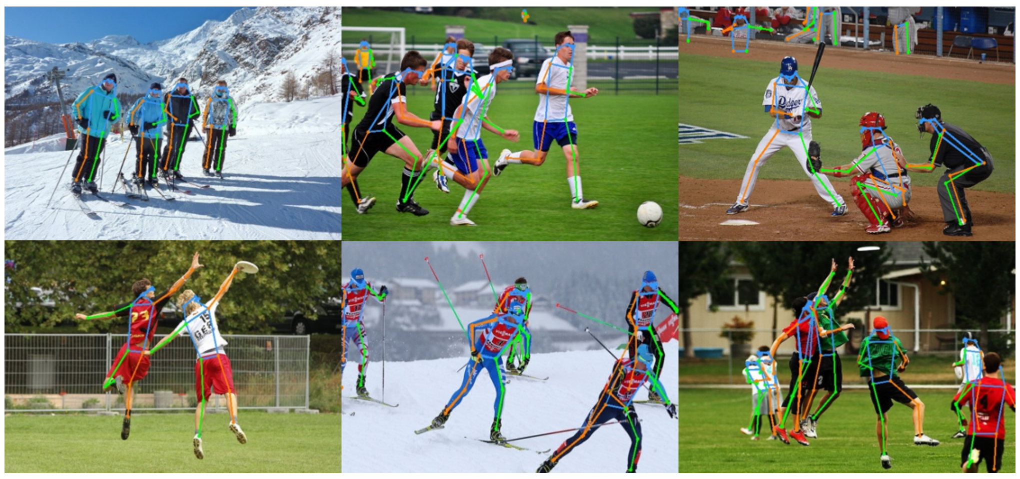

Figure 13. Lite CSW-HRNet contains more pose estimation results for viewpoint changes, occlusions, and multiple people.

5. **Conclusion.** In this paper, we optimize the weight coefficient learning method of Lite-HRNet and propose Lite CSW-HRNet, which independently computes weight maps in parallel along channels and spaces during single-resolution weight calculation and cross-resolution weight calculation, respectively, and performs adaptive feature refinement on the input feature maps. A better balance between accuracy and complexity of human pose estimation is achieved by Lite CSW-HRNet, as evidenced by the comparison experimental results, which show that all evaluation indexes of Lite CSW-HRNet outperform the current advanced lightweight human pose estimation network. At present, the posture estimation only recognition accuracy improvement can not meet the needs of society, life in a variety of terminal products more and more, due to the volume and cost constraints, the complex network is difficult to deploy, the network how to become more lightweight but can ensure the recognition accuracy has become the general trend in this research field. Under the premise of guaranteeing the light weight of the model, the next step is to explore how to

promote the channel weights in the frequency domain [22] can be better applied to the task of human posture estimation.

Acknowledgment. This work was financially supported by Jilin Scientific and Technological Development Program (grant number: 20220203135SF).

REFERENCES

[1] F. Zhang, T.-Y. Wu, J.-S. Pan, G.-Y. Ding, and Z.-Y. Li, “Human motion recognition based on SVM in VR art media interaction environment,” *Human-centric Computing and Information Sciences*, vol. 9, no. 40, pp. 1-15, 2019.

[2] S. Zhou, X.-T. Huang, M. S. Obaidat, B. A. Alzahrani, X.-M. Han, S. Kumari, and C.-M. Chen, “Transferability of Adversarial Attacks on Tiny Deep Learning Models for IoT Unmanned Aerial Vehicles,” *IEEE Internet of Things Journal*, 2023. [Online]. Available: https://doi.org/10.1109/JIOT.2023.3329954.

[3] C.-Q. Yu, B. Xiao, C.-X. Gao, L. Yuan, L. Zhang, N. Sang, and J.-D. Wang, “Lite-hrnet: A lightweight high-resolution network,” in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, pp. 10440-10450, 2021.

[4] K.-M. He, X.-Y. Zhang, S.-P. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 770-778, 2016.

[5] A. Newell, K.-Y. Yang, and J. Deng, “Stacked hourglass networks for human pose estimation,” in *Proc. European Conf. Computer Vision*, pp. 483-499, 2016.

[6] K. Sun, B. Xiao, D. Liu, and J.-D. Wang, “Deep high-resolution representation learning for human pose estimation,” in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, pp. 5693-5703, 2019.

[7] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, “Generative adversarial networks: An overview,” *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53-65, 2018.

[8] B. Xiao, H.-P. Wu, and Y.-C. Wei, “Simple baselines for human pose estimation and tracking,” in *Proc. European Conf. Computer Vision*, pp. 466-481, 2018.

[9] Z. Zhang, J. Tang, and G.-S. Wu, “Simple and lightweight human pose estimation,” *arXiv preprint arXiv:1911.10346*, 2019.

[10] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1251-1258, 2017.

[11] B. Debnath, M. O’. Brien, M. Yamaguchi, and A. Behera, “Adapting MobileNets for mobile based upper body pose estimation,” in *Proc. IEEE Int. Conf. Advanced Video and Signal Based Surveillance*, pp. 1-6, 2018.

[12] A. G. Howard, M.-L. Zhu, B. Chen, D. Kalenichenko, W.-J. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.

[13] M. Sandler, A. Howard, M.-L. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 4510-4520, 2018.

[14] M.-Y. Ding, X.-C. Lian, L.-J. Yang, P. Wang, X.-J. Jin, Z.-W. Lu, and P. Luo, “Hr-nas: Searching efficient high-resolution neural architectures with lightweight transformers,” in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, pp. 2982-2992, 2021.

[15] W.-Q. Zhang, J.-M. Fang, X.-G. Wang, and W.-Y. Liu, “Efficientpose: Efficient human pose estimation with neural architecture search,” *Computational Visual Media*, pp. 335-347, 2021.

[16] N.-N. Ma, X.-Y. Zhang, H.-T. Zheng, and J. Sun, “Shufflenet v2: Practical guidelines for efficient cnn architecture design,” in *Proc. European Conf. Computer Vision*, pp. 116-131, 2018.

[17] Q.-L. Wang, B.-G. Wu, P.-F. Zhu, P.-H. Li, W.-M. Zuo, and Q.-H. Hu, “ECA-Net: Efficient channel attention for deep convolutional neural networks,” in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, pp. 11534-11542, 2020.

[18] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, “Cbam: Convolutional block attention module,” in *Proc. European Conf. Computer Vision*, pp. 3-19, 2018.

[19] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 7132-7141, 2018.

[20] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. European Conf. Computer Vision*, pp. 740-755, 2014.

[21] B. Yang, G. Bender, Q. V. Le, and J. Ngiam, "Condconv: Conditionally parameterized convolutions for efficient inference," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[22] F. Zhang, T.-Y. Wu, G. Zheng, "Video salient region detection model based on wavelet transform and feature comparison," *EURASIP Journal on Image and Video Processing*, vol. 2019, no. 58, pp. 1-10, 2019.