This CVPR 2021 workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

MobileHumanPose: Toward real-time 3D human pose estimation in mobile devices

Sangbum Choi Seokeon Choi Changick Kim
Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea
{sangbumchoi, seokeon, changick}@kaist.ac.kr

Abstract

Currently, 3D pose estimation methods are not compatible with a variety of low computational power devices because of efficiency and accuracy. In this paper, we revisit a pose estimation architecture from a viewpoint of both efficiency and accuracy. We propose a mobile-friendly model, MobileHumanPose, for real-time 3D human pose estimation from a single RGB image. This model consists of the modified MobileNetV2 backbone, a parametric activation function, and the skip concatenation inspired by U-Net. Especially, the skip concatenation structure improves accuracy by propagating richer features with negligible computational power. Our model achieves not only comparable performance to the state-of-the-art models but also has a seven times smaller model size compared to the ResNet-50 based model. In addition, our extra small model reduces inference time by 12.2ms on Galaxy S20 CPU, which is suitable for real-time 3D human pose estimation in mobile applications. The source code is available at: https://github.com/SangbumChoi/MobileHumanPose.

1. Introduction

Due to the rapid development of deep convolutional neural networks and heatmap representation, 3D human pose estimation has significant performance improvement. This improvement helps to unlock many problems of widespread applications in human-computer interaction, robotics, surveillance, AR (augmented reality), and VR (virtual reality). In particular, Mobile Augmented Reality (MAR) has recently attracted much interest in both academia and industry. Therefore, constructing a 3D human pose estimation model with the restricted computational power is an important task. However, the performance gain of a deep learning-based model comes with a wider channel size and deeper convolution layer [44]. This leads to an increment of computing cost, which is not suitable for resource-limited devices such as smartphones.

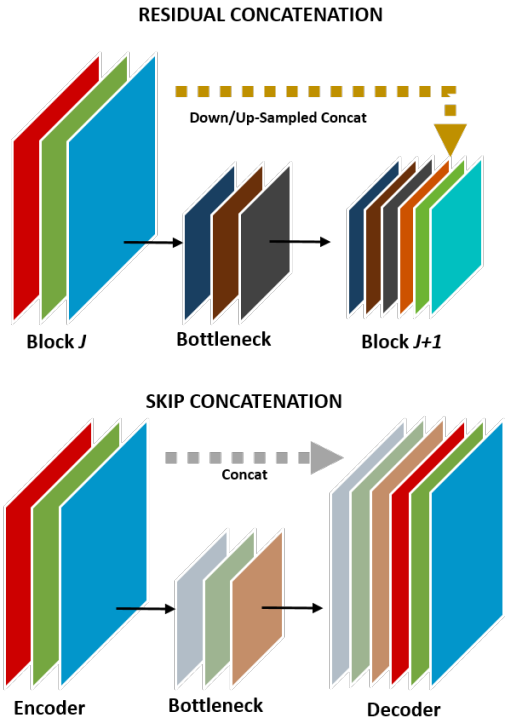


Figure 1. The difference between residual and skip concatenation structures. The residual concatenation is implemented between adjacent blocks with down/up-sampled features. In contrast, skip concatenation is a pure concatenation between the encoder and decoder with the same dimension.

Unfortunately, only two papers [10, 18] have dealt with the issue of model efficiency in various 3D human pose estimation papers. However, both methods have significant drawbacks with the following reasons: (a) Although differential architecture search (DARTS) [10] might effectively search the network architecture of 3D human pose estimation, the number of parameters and computational costs are



Figure 2. Example of the overall implementation process with our model in a mobile application. First, individually cropped images are fed into the MobileHumanPose model and the model gives the 3D joint coordinates. Second, the 3D joint coordinates are plotted on the 3D space with the corresponding skeleton. Lastly, in the visualization process, an effect behind the skeleton is removed and the remaining part of the effect is left in a camera viewpoint. Finally, it enables us to use various virtual 3D effects in a real-world application.

not affordable to run on mobile devices. (b) Hwang *et al.* [18] have proposed a more efficient model concerning the number of parameters and FLoating Operations Per Second (FLOPS). However, its performance is unsatisfactory compared to other state-of-the-art methods. Additionally, there is no clear consensus of generating a lightweight model except for using a teacher-student learning algorithm. Besides these two papers, in the practical aspect, most of the training frameworks are implemented by PyTorch [34] in 3D human pose estimation. However, using Tensorflow [1] lite is the only option to use Android Neural Networks API (NNAPI) delegate [25], which fastens the inference time of some models.

To overcome the limitations, we propose a mobile-friendly network, which considers both performance and computational cost. During the training process, we observed that a lightweight backbone (*e.g.*, MobileNetV2 [38]) is usually well-trained for an image classification task whereas having a problem of fine-tuning for a 3D human pose estimation task. To solve this problem, we use a randomly initialized model, which is not pre-trained on any datasets. Besides that, we modify the number of channels at the first four inverted residual blocks, activation function, and add skip concatenation (see Fig. 1) inspired by U-Net [37]. Specifically, we propose two different types of concatenation structures, which are skip and residual, respectively. Comparing with the residual concatenation, the skip concatenation structure propagates richer contextual information from the encoder to the decoder part, it improves accuracy while maintaining efficiency.

In addition, we construct large, small, and extra small models for a fair comparison with a viewpoint of model performance and efficiency. With 68.9% fewer parameters and 48.8% fewer FLOPS, our large model achieves comparable result to state-of-the-art models, which is 51.4mm Mean Per Joint Position Error (MPJPE) on the Human3.6M [20] dataset, and 79.6% 3D Percentage of Correct Keypoints (3DPCK$_{rel}$) on the MuPoTS [29] dataset. We provide the extra small model not only has a low model size (2.98MB) without post-quantization but also overwhelming accuracy of about 27.7mm compared to MoVNect (1.31MB) [18] while having faster inference speed (12.2ms vs. 22.0ms).

Our main contributions are summarized as follows:

- We investigate the under-studied efficiency problem in 3D human pose estimation while existing methods are focusing on improving the accuracy with the high computational cost.
- We propose an efficient model, called MobileHumanPose, for real-time 3D human pose estimation. It consists of various modifications from MobileNetV2, the skip concatenation from U-Net, and a parametric activation function. This architecture yields the best performance with the restricted computational power (less than 10 GFLOPS) compared to the similar alternative [18].
- We contribute to develop a real-world mobile application based on 3D human pose estimation by releasing our code. Figure 2 visualizes the examples of visual effects that can be derived from 3D pose estimation results.

2. Related work

Efficient architecture After the public release of AlexNet [23] and VGG [40], many researchers have proposed deeper and wider neural networks. However, there was a gradient vanishing problem as the layer depth of the network increases. To solve this problem, He *et al*. [16] used residual connections to keep the signal strength of back-propagation. While the above neural networks are aimed at the performance of diverse tasks, some networks are specifically intended for resource-constrained circumstances. For example, the inverted residual block has a small number of parameters and low computational cost in MobileNetV2 [38] by using depthwise convolution in the expansion layers. Furthermore, Zhou *et al*. [53] pointed out the bottleneck problem of the inverted residual block and proposed MobileNeXt, which is the revised version of the inverted residual block to enhance accuracy in diverse tasks.

In addition, many Neural Architecture Search (NAS) based papers came out for model efficiency. MNasNet [43] used the NAS algorithm specifically aimed at mobile devices. Additionally, MobileNetV3 [17] is improvement of MNasNet by using NetAdapt [49]. Besides using cheap operators such as pointwise and depthwise convolution, recently Han *et al*. [13] proposed 'ghost module', which reduces a channel dimension to have low computational power while maintaining information.

In these existing networks, we study the compatibility of various lightweight backbones in the pose estimation task and examine the problem of weight initialization.

2D human pose estimation Existing 2D human pose estimation methods can be divided into bottom-up and top-down approaches. The bottom-up approach [19, 22, 32] is to search all possible candidates of human joints and obtain the most accurate connection between the joints. Cao *et al*. [7] used Part Affinity Field (PAF) to match corresponding joints in multi-person circumstances.

In contrast, the top-down approach [9, 48, 33, 47] is to use a human detector to find the bounding boxes of each person in a single image. Individually cropped images are fed into the model of the human pose estimator to extract a 2D heatmap for each joint. Tang *et al*. [45] proposed quantized and densely-connected U-Net [37] to improve information flow. Otherwise, Tosheve *et al*. [46] suggested extracting joints' value with a fully connected layer to reduce model complexity. The current state-of-the-art of this task [4] not only used a hybrid network that consists of Hourglass [33] and U-Net [37] but also proposed soft-gated skip connections to learn more complicated functions.

We investigate the effectiveness of the U-Net structure used in 2D human pose estimation and transfer this effectiveness to the 3D task by proposing two types of different concatenations.

3D human pose estimation 3D human pose estimation methods can be categorized into two-stage and one-stage approaches. The two-stage approach [8, 26, 51, 11] is separated by extracting 2D joints of humans and lifting them into the 3D spaces. In contrast, the one-stage approach [3, 41, 35, 50, 27, 39, 31] is based on a volumetric heatmap to extract 3D joints. Sun *et al*. [42] introduced a volumetric heatmap, which uses a regression method to estimate 3D joints by integrating all weights by their probabilities. Since the accuracy of two-stage approach is highly related to the 2D joint prediction, our method follows the one-stage approach.

However, most of the pose estimation papers focus on the performance and the models require high computational costs beyond the budgets of many mobile applications. Among various 3D human pose estimation papers, only two of them introduce the model efficiency problem. Chen *et al*. [10] used NAS to obtain a part-specific architecture in this task. As a result, the method achieved state-of-the-art on various datasets. Hwang *et al*. [18] proposed MoVNect, which is a combination of VNect [30] and MobileNetV2 [38]. They also used knowledge distillation to transfer the information from the teacher model. However, both methods [18, 10] did not satisfy the perfect balance between model accuracy and efficiency. To handle this problem, we propose a new lightweight model for 3D human pose estimation from a single RGB image.

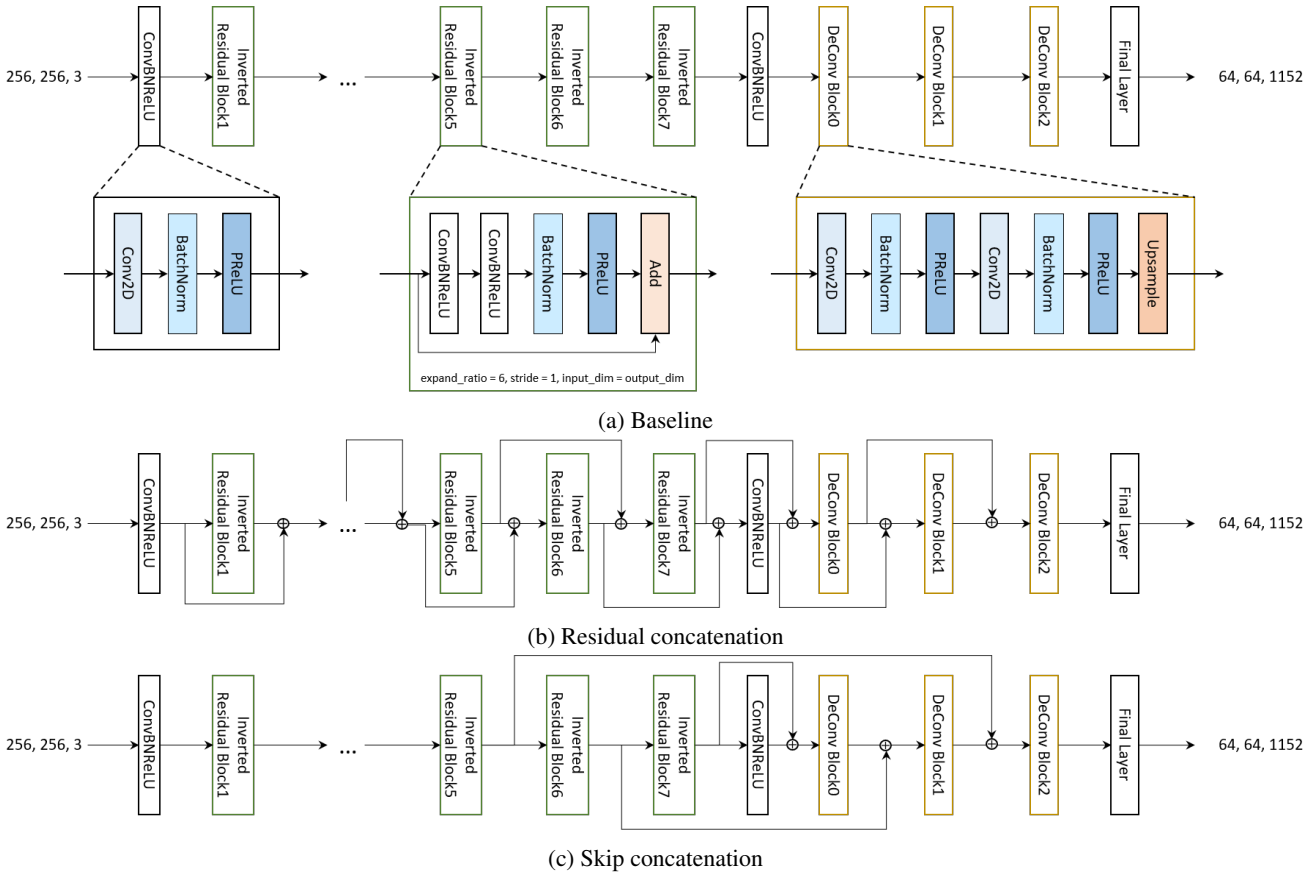
Method	Avg.
Random Initialization	56.38
Pre-trained on ImageNet	57.09

Table 1. Performance comparison between a randomly initialized model and pre-trained model on ImageNet. For evaluation, we used Human36M protocol 2 [20]. Metric: MPJPE (mm).

3. Proposed Method

3.1. Issue of initialization

Most of the 3D human pose estimation papers [48, 31, 18, 36, 39, 42] use backbone networks pre-trained on the ImageNet dataset [23]. However, He *et al*. [14] re-emphasized that a randomly initialized network is no worse than the pre-trained network on the ImageNet dataset. To prove this hypothesis in 3D human pose estimation, we analyze the training loss of our backbone network when each learning process is initialized from the following parameters: random initialization and pre-trained on ImageNet. As a result, the randomly initialized network has a lower evaluation error, which is 56.38mm Mean Per Joint Position Error (MPJPE) shown in Table 1.



(a) Baseline

(b) Residual concatenation

(c) Skip concatenation

Figure 3. An illustration of the proposed concatenation structure. The skip concatenation is used for our final model.

3.2. MobileHumanPose

Among the previous 3D human pose estimation methods, models proposed by Moon *et al*. [31] and Chen *et al*. [10] focused on accuracy rather than efficiency. We follow the basic encoder-decoder structure (*e.g.,* bottleneck dimension and volumetric heatmap dimension) where the encoder performs global feature extraction and the decoder estimates the pose estimation. However, we modified the backbone block itself and channel size at the first four inverted residual blocks, an activation function, and implement the skip concatenation from U-Net [37].

Backbone network Although we choose the baseline structure of Chen *et al*., it is necessary to find an appropriate lightweight backbone for model efficiency. We compare the performance of the ResNet family [16] and that of various lightweight networks (see Table 2). All backbones are implemented from the initial layer to the first intermediate block that reaches 8×8 dimensions from the aforementioned pose estimation structure. In Table 2, MNasNet [43] scores the lowest MPJPE among the lightweight backbones. However, for a fair comparison with MNasNet, MobileNetV2 [38] still has more computational budget (3.36M/1.48G) to have a similar model efficiency (4.23M/1.49G). Therefore, we modify the channel size at the first four inverted residual blocks of MobileNetV2 for performance gain. Finally, by comparing the modified version of MobileNetV2 and MNasNet, we choose the former network as our backbone.

Activation function Bulat *et al*. [5] show that performance increases when the Parametric ReLU (PReLU) [15] function is used in a 2D human pose estimation task. Since the PReLU function has a learnable parameter contrast to ReLU, it derives additional information in each layer. The PReLU function, f is defined as the following equation:

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \leq 0 \end{cases}, \tag{1}$$

where a_i and y_i denote as a learnable parameter and input signal.

Therefore, we apply PReLU to 3D pose estimation as well. We also compare the PReLU and ReLU activation functions with the same training condition in this study. Figure 3 (a) shows the baseline of our method after modifying the activation function.

Skip concatenation Most of the functions (*e.g.*, Conv2D, ReLU) in PyTorch [34] are counted as FLOPS. Even though our baseline structure achieves great model efficiency, it is important to use FLOPS uncountable functions (*e.g.*, Concat, Bilinear) appropriately in a viewpoint of the inference speed. Considering the above factor, we propose two types of different concatenation structures. Specific details are shown in Fig. 3 (b) and (c).

For the residual concatenation, we use an average pooling function to match the dimension in the encoder part and use a bilinear function in the decoder part. We use a pure concatenation of two different outputs that have the same dimension in the skip concatenation. Bulat *et al*. [4] claimed that at least for some cases, residual connections will not fully perform and degrade the performance. Accordingly, we choose the skip concatenation, which can derive the low-level feature signal from the encoder to the decoder in contrast to the residual concatenation.

Loss We use the L1 loss [31, 10] between the ground-truth and predicted coordinates. Our loss L is defined as the following equation:

$$L = \frac{1}{J}\sum_{j=1}^{J} \|P_j - P_j^{GT}\|, \tag{2}$$

where GT denotes the ground-truth.

4. Experiment

4.1. Datasets and settings

Datasets and evaluation metrics We used Human3.6M [20] and MuCo-3DHP [29] as 3D human pose datasets.

While training Human3.6M, we employed the additional 2D human pose MPII dataset [2]. A z-value for 3D coordinate becomes zero for the input of this training while using MPII. Human3.6M dataset is split into two protocols named Protocol 1 and Protocol 2. Protocol 1 consists of S1, S5, S6, S7, S8, S9 for training and S11 for testing. Protocol 2 contains S1, S5, S6, S7, S8 for training and S9, S11 for testing. For the evaluation of pose estimation methods, two popular metrics are used: Mean Per Joint Position Error (MPJPE) for protocol 1 and Procrustes Analysis Mean Per Joint Position Error (PA-MPJPE) for protocol 2.

For another training process, we used MuCo-3DHP and the additional 2D human keypoint detection COCO dataset [24]. MuCo-3DHP is a 3D single-person pose dataset, which consists of MPI-INF-3DHP dataset. The test set, MuPoTS-3DHP, includes 20 real-world scenes with the ground-truth of 3D poses for up to three subjects captured in outdoor circumstances. We adopted a 3D Percentage of Correct Keypoints (3DPCK$_{rel}$) evaluation metric after root alignment with ground-truth data.

Implmentation details We used a random initialization method based on the experiment of section 3.1 because the pre-trained backbone is not necessary for the performance gain. The initial learning rate is set to 10^{-3} and divided by a factor of 10 at the 17th and 21th epochs. The Adam [21] optimizer with a mini-batch size of 24 was used in this training. We trained the MobileHumanPose mdoel for 25 epochs with two NVIDIA TITAN RTX GPUs, which took three days. We performed the data augmentation followed by [31], which includes rotation, horizontal flip, color jittering, and synthetic occlusion [52]. We also presented the horizontal flip augmentation in the testing phase.

Framework We used the PyTorch [34] framework for training our baseline and the TFLite [1] framework for implementing in mobile devices. The default hyperparameter values (*i.e.*, momentum, epsilon, and other values) between PyTorch and Tensorflow are fixed to PyTorch values when training on Tensorflow.

Backbone	Param.	FLOPS	Avg.
GhostNet	2.02M	1.21G	83.73
MobileNetV3	4.28M	1.39G	70.94
MobileNeXt	1.83M	1.39G	61.27
MobileNeXt*	1.83M	1.39G	61.19
MNasNet	4.23M	1.49G	55.85
MobileNetV2	3.36M	1.48G	56.38
MobileNetV2*	3.36M	1.48G	57.09
ResNet18	12.55M	3.46G	56.96
ResNet18*	12.55M	3.46G	53.45
ResNet50	28.42M	7.36G	54.80
ResNet50*	28.42M	7.36G	53.79

Table 2. Comparison with various backbone networks. * indicates the corresponding network pre-trained on ImageNet. Metric: MPJPE (mm).

4.2. Comparison with various backbones

As shown in Table 2, the performance of GhostNet [13], MobileNetV3 [17], and MobileNeXt [53] is not robust enough compared to the remainder, so we excluded these candidates. We chose the best performing backbone, which has the different channel size at the first four inverted residual blocks in MobileNetV2 [38].

4.3. Ablation study

Initialization and channel modification Table 3 shows that our modified version of MobileNetV2 [38] has lower MPJPE in both random initialization and pre-trained on the ImageNet dataset [23].

Activation function Table 4 shows the effectiveness of activation functions. Using the PReLU function achieves 2.65mm lower MPJPE contrast to the ReLU function. Whereas the PReLU function requires additional back-propagation memory consumption on GPU due to the extra

Backbone	Param.	FLOPS	Avg.
MobileNetV2	3.36M	1.48G	56.38
MobileNetV2*	3.36M	1.48G	57.09
MobileNetV2†	3.36M	1.74G	55.53
MobileNetV2*†	3.36M	1.74G	56.24

Table 3. Comparison between random initialization and pre-training methods on the Human36M protocol 2 [20]. * and † indicate the network pre-trained on ImageNet and the modified version of the initial four inverted residual blocks, respectively. Metric: MPJPE (mm).

Activation	Param.	FLOPS	Avg.
ReLU	4.03M	5.43G	54.09
PReLU	4.03M	5.43G	53.09

Table 4. Comparison with different activation functions. The default value of the width multiplier and intermediate layer size is set to 1.0 and 2048, respectively. Metric: MPJPE (mm).

Res. con.	Skip con.	Param.	FLOPS	Avg.
-	-	4.03M	5.43G	53.09
✓	-	7.25M	11.45G	54.77
-	✓	4.07M	5.49G	51.44

Table 5. Comparison of different ways to construct the various types of concatenation. The default value of the width multiplier and the intermediate layer size is set to 1.0 and 2048, respectively. Metric: MPJPE (mm).

variable of slope, there was no significant difference with the ReLU activation function in terms of throughput and inference time, which allows us to use PReLU in a real-world application.

Concatenation structure Table 5 shows the effectiveness of concatenation in the baseline architecture. Compared to Fig. 3 (a) and (c), it indicates that the skip concatenation derives richer information to a deeper layer of the architecture to enhance the performance of this task. The performance result of Fig. 3 (a) and (b) shows that the residual concatenation is not performed well even though FLOPS and the number of parameters dramatically increased due to enlarging a channel size in each layer.

Cost effectiveness We extensively examined our proposed architecture design from a viewpoint of two dimensions: backbone width multiplier and the channel size at the intermediate layer (see the intermediate layer shown in Fig. 3). We defined our small model as 0.75 width multiplier and 512 channels in the intermediate layer and large model as 1.0 width multiplier and 2048 channels in the intermediate layer. Table 6 shows the result of architecture efficiency and the performance of diverse architectures. Since the width multiplier is related to the whole encoder structure, reducing this parameter gave a strong architecture efficiency, which is related to the throughput and inference time. However, the result also shows a strong dependency between information capacity and width multiplier. On the other hand, the results of 512 and 1024 intermediate channel sizes show a relatively small performance variation in contrast to 1024 and 2048. This indicates that the bottleneck of information capacity relies on the encoder part and 512 intermediate channel size is enough to be implemented in a small model.

Methods	Width multi.	Inter. layer	Param.	FLOPS	Avg.
Ours-S	0.75	512	2.24M	3.92G	56.94
Ours	1	512	3.33M	5.44G	54.24
Ours	0.75	1024	3.43M	3.93G	56.57
Ours	1	1024	3.58M	5.46G	54.79
Ours	0.75	2048	2.80M	3.96G	53.39
Ours-L	1	2048	4.07M	5.49G	51.44

Table 6. Cost effectiveness analysis of our proposed model. Metric: MPJPE (mm).

4.4. Comparison with state-of-the-art methods

We compared the performance and efficiency on Human3.6M [20] and MuPoTS [29] including the state-of-the-art methods. Figure 4 shows the qualitative result by using our large model in various images.

H36M Table 8 shows the comparison of various models on Human3.6M protocol 2. In a viewpoint of accuracy, our large model achieves 51.44mm Mean Per Joint Position Error (MPJPE), which is a comparable result in contrast to state-of-the-art methods. However, computational costs (*i.e.*, FLOPS and number of parameters) of our model have only 5.0 times fewer number of parameters (4.07M vs. 20.4M) and 2.6 times fewer FLOPS (5.49G vs. 14.1G) in contrast to the model of Chen *et al.* [10]. In addition, we evaluate a small model of our method. Our small model achieves 56.94mm MPJPE with having 2.24M number of parameters and 3.92 GFLOPS.

Table 7 shows the comparison of various models on Human3.6M protocol 1. The tendency of the evaluation score is similar to Table 8. However, the average difference between the large and small model is decreased compared to Table 8.

MuPoTS For the multi-person 3D pose estimation task, we used RootNet [31] to estimate the absolute coordinate for each person. In Table 9, we compared with the small model of Chen *et al.* [10] and Moon *et al.* [31]. Among 20 scenes on MuPoTS, our large model achieves

Methods	Dir.	Dis.	Eat.	Gre.	Phon.	Pose	Pur.	Sit.	SitD.	Smo.	Phot.	Wait	Walk	WalkD.	WalkP.	Avg.	Param.
Martinez [26]	39.5	43.2	46.4	47.0	51.0	56.0	41.4	40.6	56.5	69.4	49.2	45.0	49.5	38.0	43.1	47.7	-
Fang [12]	38.0	41.7	43.7	44.9	48.5	55.3	40.2	38.2	54.5	64.4	47.2	44.3	47.3	36.7	41.7	45.7	-
Sun [42]	36.9	36.2	40.6	40.4	41.9	34.9	35.7	50.1	59.4	40.4	44.9	39.0	30.8	39.8	36.7	40.6	34.00M
Moon [31]	31.0	30.6	39.9	35.5	34.8	30.2	32.1	35.0	43.8	35.7	37.6	30.1	24.6	35.7	29.3	34.0	34.34M
Chen [10]	27.5	30.9	34.0	35.5	32.4	30.8	31.9	32.7	41.9	36.3	39.1	28.4	23.3	37.1	27.0	32.7	20.40M
Ours-S	30.3	32.9	38.4	35.4	34.9	32.1	32.3	37.6	49.6	38.2	42.2	31.3	26.9	37.8	31.6	35.7	2.24M
Ours-L	31.0	32.7	37.5	34.3	35.1	31.4	32.1	37.3	47.9	38.7	40.6	30.6	26.2	37.5	30.6	35.2	4.07M

Table 7. Comparison with state-of-the-art methods on Human3.6M using Protocol 1. Metric: PA-MPJPE (mm).

Methods	Dir.	Dis.	Eat.	Gre.	Phon.	Pose	Pur.	Sit.	SitD.	Smo.	Phot.	Wait	Walk	WalkD.	WalkP.	Avg.	Param.
Hwang [18]	72.4	83.4	76.9	82.1	101.9	70.4	91.8	156.5	193.0	92.8	108.4	85.1	76.8	97.2	70.5	97.3	1.03M
Mehta [30]	62.6	78.1	63.4	72.5	88.3	63.1	74.8	106.6	138.7	78.8	93.8	73.9	55.8	82.0	59.6	80.5	14.60M
Martinez [26]	51.8	56.2	58.1	59.0	69.5	55.2	58.1	74.0	94.6	62.3	78.4	59.1	49.5	65.1	52.4	62.9	-
Fang [12]	50.1	54.3	57.0	57.1	66.6	53.4	55.7	72.8	88.6	60.3	73.3	57.7	47.5	62.7	50.6	60.4	-
Moon [31]	50.5	55.7	50.1	51.7	53.9	46.8	50.0	61.9	68.0	52.5	55.9	49.9	41.8	56.1	46.9	53.3	34.34M
Sun [42]	47.5	47.7	49.5	50.2	51.4	43.8	46.4	58.9	65.7	49.4	55.8	47.8	38.9	49.0	43.8	49.6	34.00M
Chen [10]	41.4	48.6	42.0	45.3	47.1	42.3	46.0	57.9	62.1	47.8	51.2	43.6	36.1	51.1	41.4	47.3	20.40M
Ours-S	51.1	58.7	49.9	53.0	58.3	48.9	53.0	70.9	77.5	58.2	61.0	51.9	42.9	58.6	50.0	56.9	2.24M
Ours-L	45.5	51.8	45.9	48.4	52.1	43.7	48.2	63.6	70.2	52.4	56.2	46.2	40.2	54.9	45.4	51.4	4.07M

Table 8. Comparison with state-of-the-art methods on Human3.6M using Protocol 2. Metric: MPJPE (mm).

Methods	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	Avg.
Mehta*[28]	89.7	65.4	67.8	73.3	77.4	47.8	67.4	63.1	78.1	85.1	75.6	73.1	65.0	59.2	74.1	84.6	87.8	73.0	78.1	71.2	72.8
Chen†[10]	93.1	76.7	79.9	78.2	83.6	64.6	79.0	72.5	87.6	88.3	76.1	79.4	71.1	70.6	77.7	86.6	87.1	80.3	79.5	72.0	79.2
Moon [31]	94.4	77.5	79.0	81.9	85.3	72.8	81.9	75.7	90.2	90.4	79.2	79.9	75.1	72.7	81.1	89.9	89.6	81.8	81.7	76.2	81.8
Ours-S	89.5	72.1	72.4	75.4	81.0	67.1	80.2	64.0	83.7	89.6	76.5	80.9	68.4	68.2	80.9	89.5	88.4	82.8	72.8	68.7	77.6
Ours-L	91.0	82.5	77.1	74.3	84.1	67.9	79.2	76.0	84.0	90.7	74.5	77.5	68.1	76.0	77.7	88.0	87.9	82.7	78.3	75.0	79.6

Table 9. Comparison with state-of-the-art methods on MuPoTS-3D using all ground-truth. $\star$ denotes as stage ii and † denotes as a small model of the corresponding paper. Metric: 3DPCK$_{rel}$(%).

superior performance in contrast to the small model of Chen *et al*. on half of the scenes and scores state-of-the-art performance on few scenes. Additionally, our large model (2.24M/3.92GFLOPS) outperforms the small model of Chen *et al*. (13.0M/10.7GFLOPS) in a viewpoint of model efficiency. Qualitative results can be seen in Fig. 5.

4.5. Inference analysis

Since our method contains a heavy computation function, which is a softmax function to give actual 3D joint coordinates from a volumetric heatmap, we compared both the softmax and non-softmax model in the inference stage because the softmax function can be easily implemented in C++ to reduce the inference time. To be specific, the output after the softmax function is denoted as P_j and before is P_j^S, which is an output of the volumetric heatmap.

$$P_j = \mathbf{E}(\text{softmax}(P_j^S)) \in \mathcal{R}^{1\times1\times3}, \quad P_j^S \in \mathcal{R}^{1\times64\times64\times64}, \tag{3}$$

where j indicates a single joint in the human body.

In this experiment, we define 'Throughput' usually known as frame per second when the batch size is set to one [6]. It is a variable to measure the speed performance of the model in diverse processing units. N is denoted as the number of iterations, which is 100, B is the optimal batch size, which is one, and T is set to the total inference time of the whole inference process.

$$Throughput = \frac{NB}{T}. \tag{4}$$

We compared the inference time of Hwang *et al*. [18], Moon *et al*. [31], and our models. In Table 10, we used Titan RTX Graphic Processing Unit (GPU) to evaluate the PyTorch [34] framework experiment, and TFlite is based on the Tensorflow [1] framework tested on Galaxy S20. Our large model outperforms the model of Moon *et al*. in all sectors, which are throughput, inference time, accuracy, and model size. Our extra small model is specially aimed for a mobile application, which has 0.5 width multiplier, 512 channels in the intermediate layer, and half resolution $(32 \times 32 \times 32)$ of the volumetric heatmap compared to the original shape $(64 \times 64 \times 64)$. Our extra small model has the comparable size (2.98MB) to MoVNect (1.31MB) [18] but advances 27.7mm MPJPE and reduces 9.8 $\sim$ 12.2ms inference time.

Methods	PyTorch - Titan RTX ($Throughput$)	TFlite - Galaxy S20 (ms)			Avg.	Param.	TFlite size (MB)
	GPU	CPU	GPU	NNAPI			
Moon†*[31]	107.0 ± 0.7	251.0 ± 4.9	231.0 ± 2.7	809.0 ± 10.4	53.3	34.34M	133.90
Ours-L†	141.6 ± 1.3	108.0 ± 6.9	109.0 ± 3.2	135.0 ± 5.6	**51.4**	4.07M	5.59
Ours-S†	**147.4 ± 1.1**	**85.2 ± 3.2**	**87.4 ± 2.8**	**110.0 ± 4.5**	56.9	**2.24M**	**3.81**
Hwang⋆[18]	-	22.0 ± 5.0	25.2 ± 4.9	32.0 ± 5.2	97.3	1.03M	**1.31**
Ours-XS†	**161.7 ± 1.3**	**12.2 ± 4.2**	**13.0 ± 4.0**	**21.1 ± 5.6**	**69.6**	**0.96M**	2.98

Table 10. Measuring throughput on Titan RTX and inference time on Galaxy S20 settings. The higher value in throughput and lower value in inference time designate better performance. † indicates the baseline architecture without the softmax layer. ∗ and ⋆ indicate official/unofficial implementation in both frameworks, repectively. Metric: MPJPE (mm).

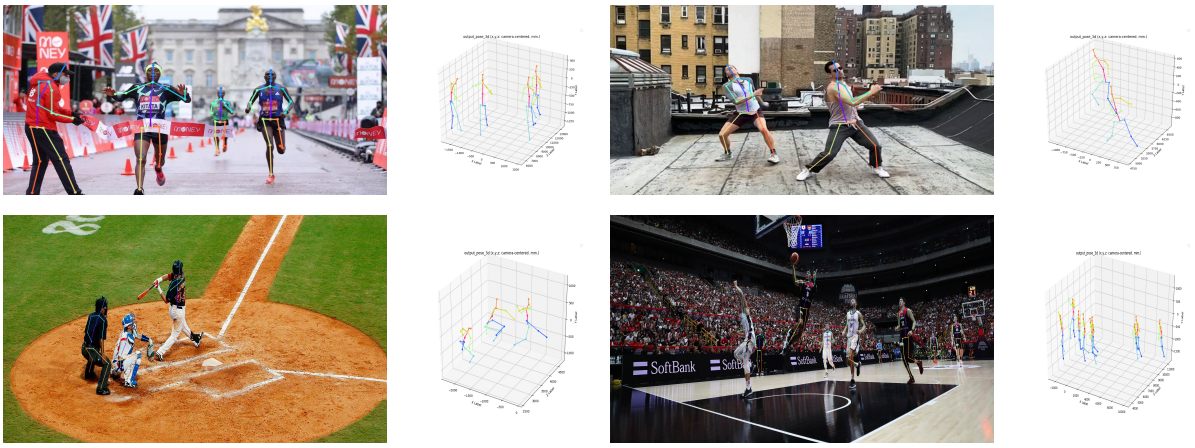


Figure 4. Qualitative results on various images crawled from the Web.

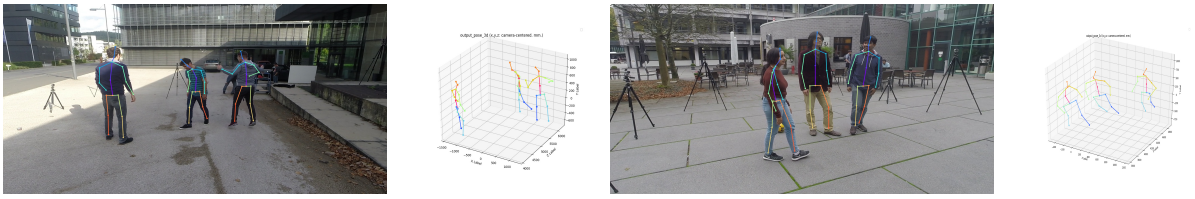


Figure 5. Qualitative results on MuPoTS.

4.6. Applications

Our proposed model can be implemented on various mobile applications that have a suitable format for 3D avatar control. In particular, our extra small model has extremely low inference time, it is capable of a real-time application. We developed Android and Unity3D a custom dispatcher to communicate 3D joints coordinate to use various visual effects from Unity3D. We will also release our code for a development of mobile applications based on 3D human pose estimation.

5. Conclusion

In this paper, we mainly focused on constructing a lightweight model and improving accuracy with detailed experiments. We claimed that the lightweight backbones are not performing well in the pose estimation task. To handle this problem, we suggested the modified version of backbone network based on MoblieNetV2 [38]. Also, we employed the skip concatenation and the parametric activation function to improve the accuracy while maintaining cost efficiency in contrast to the baseline architecture. Accordingly, we constructed MobileHumanPose, which is the most precise and compact model that can be implemented in mobile devices. The strategies implemented in this paper are not only restricted to the top-down and one-stage methods but also can be used in all kinds of 3D human pose estimation. We hope that this work gives a new idea for real-time 3D human pose estimation in mobile devices.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015.

[2] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt 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*, pages 3686–3693, 2014.

[3] Mathieu Salzmann Vincent Lepetit Bugra Tekin, Isinsu Katircioglu and Pascal Fua. Structured prediction of 3d human pose with deep neural networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 130.1–130.11. BMVA Press, September 2016.

[4] A. Bulat, J. Kossaifi, G. Tzimiropoulos, and M. Pantic. Toward fast and accurate human pose estimation via soft-gated skip connections. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 8–15, 2020.

[5] Adrian Bulat, Georgios Tzimiropoulos, Jean Kossaifi, and Maja Pantic. Improved training of binary networks for human pose estimation and image recognition. *arXiv preprint arXiv:1904.05868*, 2019.

[6] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.

[7] 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*, 43(1):172–186, 2019.

[8] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation= 2d pose estimation+ matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7035–7043, 2017.

[9] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7103–7112, 2018.

[10] Zerui Chen, Yan Huang, Hongyuan Yu, Bin Xue, Ke Han, Yiru Guo, and Liang Wang. Towards part-aware monocular 3d human pose estimation: An architecture search approach. Springer, 2020.

[11] Rishabh Dabral, Nitesh B Gundavarapu, Rahul Mitra, Abhishek Sharma, Ganesh Ramakrishnan, and Arjun Jain. Multi-person 3d human pose estimation from monocular images. In *2019 International Conference on 3D Vision (3DV)*, pages 405–414. IEEE, 2019.

[12] Haoshu Fang, Yuanlu Xu, Wenguan Wang, Xiaobai Liu, and Song-Chun Zhu. Learning knowledge-guided pose grammar machine for 3d human pose estimation. In *AAAI*, page 6, 2018.

[13] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1580–1589, 2020.

[14] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4918–4927, 2019.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

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

[17] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

[18] Dong-Hyun Hwang, Suntae Kim, Nicolas Monet, Hideki Koike, and Soonmin Bae. Lightweight 3d human pose estimation network training using teacher-student learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 479–488, 2020.

[19] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer, 2016.

[20] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. Multiposenet: Fast multi-person pose estimation using pose residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 417–433, 2018.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[24] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[25] Chunjie Luo, Xiwen He, Jianfeng Zhan, Lei Wang, Wanling Gao, and Jiahui Dai. Comparison and benchmarking of ai models and frameworks on mobile devices. *arXiv preprint arXiv:2005.05085*, 2020.

[26] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017.

[27] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *2017 International Conference on 3D Vision (3DV)*, pages 506–516, 2017.

[28] Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Mohamed Elgharib, Pascal Fua, Hans-Peter Seidel, Helge Rhodin, Gerard Pons-Moll, and Christian Theobalt. Xnect: Real-time multi-person 3d human pose estimation with a single rgb camera. *arXiv preprint arXiv:1907.00837*, 2019.

[29] Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Srinath Sridhar, Gerard Pons-Moll, and Christian Theobalt. Single-shot multi-person 3d pose estimation from monocular rgb. In *2018 International Conference on 3D Vision (3DV)*, pages 120–130. IEEE, 2018.

[30] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017.

[31] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. Camera distance-aware top-down approach for 3d multi-person pose estimation from a single rgb image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10133–10142, 2019.

[32] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 2277–2287. Curran Associates, Inc., 2017.

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

[34] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[35] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7025–7034, 2017.

[36] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. Lcr-net++: Multi-person 2d and 3d pose detection in natural images. *IEEE transactions on pattern analysis and machine intelligence*, 42(5):1146–1161, 2019.

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[38] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[39] István Sárándi, Timm Linder, Kai O. Arras, and Bastian Leibe. Metric-scale truncation-robust heatmaps for 3D human pose estimation. In *IEEE Int Conf Automatic Face and Gesture Recognition (FG)*, 2020.

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[41] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. Compositional human pose regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2602–2611, 2017.

[42] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 529–545, 2018.

[43] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[44] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[45] Zhiqiang Tang, Xi Peng, Shijie Geng, Lingfei Wu, Shaoting Zhang, and Dimitris Metaxas. Quantized densely connected u-nets for efficient landmark localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 339–354, 2018.

[46] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[47] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[48] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 466–481, 2018.

[49] Tien-Ju Yang, Andrew Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze, and Hartwig Adam. Netadapt: Platform-aware neural network adaptation for mobile applications. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 285–300, 2018.

[50] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5255–5264, 2018.

[51] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3425–3435, 2019.

[52] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13001–13008, 2020.

[53] Daquan Zhou, Qibin Hou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Rethinking bottleneck structure for efficient mobile network design. *ECCV, August*, 2, 2020.