Towards Part-Aware Monocular 3D Human Pose Estimation: An Architecture Search Approach

Zerui Chen[1,3](✉), Yan Huang[1], Hongyuan Yu[1,3], Bin Xue[3], Ke Han[1],
Yiru Guo[5], and Liang Wang[1,2,4]

[1] Center for Research on Intelligent Perception and Computing, NLPR, CASIA,
Beijing, China
{zerui.chen,hongyuan.yu,ke.han}@cripac.ia.ac.cn,
{yhuang,wangliang}@nlpr.ia.ac.cn
[2] Center for Excellence in Brain Science and Intelligence Technology, CAS,
Beijing, China
[3] School of Artificial Intelligence, University of Chinese Academy of Sciences,
Beijing, China
xuebin2018@ia.ac.cn
[4] Chinese Academy of Sciences, Artificial Intelligence Research (CAS-AIR),
Beijing, China
[5] School of Astronautics, Beihang University, Beijing, China
guoyiru@buaa.edu.cn

Abstract. Even though most existing monocular 3D pose estimation approaches achieve very competitive results, they ignore the heterogeneity among human body parts by estimating them with the same network architecture. To accurately estimate 3D poses of different body parts, we attempt to build a part-aware 3D pose estimator by searching a set of network architectures. Consequently, our model automatically learns to select a suitable architecture to estimate each body part. Compared to models built on the commonly used ResNet-50 backbone, it reduces 62% parameters and achieves better performance. With roughly the same computational complexity as previous models, our approach achieves state-of-the-art results on both the single-person and multi-person 3D pose estimation benchmarks.

Keywords: 3D pose estimation · Body parts · Neural architecture search

1 Introduction

3D human pose estimation plays a crucial role to unlock widespread applications in human-computer interaction, robotics, surveillance, and virtual reality. Compared with multi-view methods [19,41,43,52,61], monocular 3D human pose estimation is more flexible for deployment in outdoor environments. However, given its ill-posed nature, estimating 3D human poses from a single RGB

A. Vedaldi et al. (Eds.): ECCV 2020, LNCS 12348, pp. 715–732, 2020.
https://doi.org/10.1007/978-3-030-58580-8_42

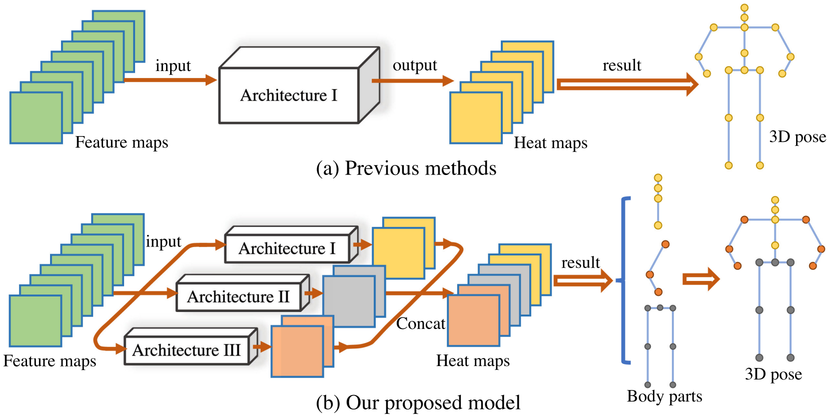


Fig. 1. Motivation. Most of the previous methods employ a single network architecture to deal with intrinsically heterogeneous human body parts (as shown in (a)). Instead, we are motivated to search for a suitable network architecture for a group of parts and estimate their 3D locations with a part-specific architecture (as shown in (b)).

image remains a challenging problem. Thanks to Convolutional Neural Networks (CNNs), many effective approaches are proposed and formulate the problem as joint coordinate regression [28,47] or heat maps learning [57,65]. Recently, many approaches [39,40,48,62] have followed a popular paradigm in predicting per voxel likelihood for each human joint and achieved competitive performance.

In most previous approaches shown in Fig. 1(a), CNNs share the same network architecture for predicting all human body parts with different degrees of freedom (DOFs), ranging from parts with higher DOFs like the wrists to parts with lower DOFs like the torso. However, a single network architecture might be sub-optimal to deal with various body parts. Because different parts might have various movement patterns and shapes, estimating their locations might require different network topologies (*e.g.*, different kernel sizes and distinct receptive fields). A recent effort [54] also demonstrates that it is effective to estimate different body parts by explicitly taking their DOFs into account.

As shown in Fig. 1(b), we approach the problem from a different angle and propose to estimate different body parts with part-specific network architectures. However, looking for optimal architectures for various body parts is an intractable and time-consuming job even for an expert. Therefore, instead of designing them manually, we consult the literature of neural architecture search (NAS) [4,14,17,23,31,49,56] and propose to search part-specific network architectures for different parts. In fact, the idea of searching network architectures for certain tasks is not new. Specifically, it has been applied in semantic segmentation [7,30,60] and object detection [8,13,42].

However, applying NAS into 3D human pose estimation is non-trivial, because current NAS approaches mainly focus on 2D visual tasks. Different from them, 3D human poses are commonly estimated in a higher-order volumetric space [11,40,48,52]. It consists of 2D spatial and depth axes and greatly increases the uncertainty during optimization. More importantly, how to use prior information about the human body structure to facilitate the architecture search and achieve a trade-off between accuracy and complexity is another issue.

To deal with these issues, we introduce the fusion cell in the context of NAS to increase the resolution of feature maps and generate desired volumetric heat maps efficiently. The fusion cell has multiple head networks that are various convolutional architectures, consisting of different kernels and operations. To improve the part-awareness of our model, we attempt to generate the volumetric heat map for each part with a specially optimized head network. Considering the symmetry prior of the human body structure, it is inefficient to search a different head network for each part. Our approach classifies all body parts into several groups and assigns each group with a part-specific architecture. In the search stage of our approach, all the architectures, including the fusion cell, are optimized by gradient descent. Then, we stack these optimized computational cells to construct our part-aware 3D pose estimator. In the evaluation stage, our part-aware 3D human pose estimator can select optimized head networks encoded in the fusion cell to estimate different groups of body parts.

Through extensive experiments, we show that our approach can achieve a good trade-off between complexity and performance. With 62% fewer parameters and 24% fewer FLOPs (multiply-adds), our approach outperforms the model using ResNet-50 backbone and achieves 53.6 mm in Mean Per Joint Position Error (MPJPE). By stacking more computational cells, it can further advance the state-of-the-art accuracy on Human3.6M by 2.3 mm with 41% fewer parameters.

Our contributions can be summarized as follows:

- Our work shows that it might be sub-optimal to estimate 3D poses of all body parts with a single network architecture. To the best of our knowledge, we make the first attempt to search part-specific architectures for different parts.
- We introduce the fusion cell to generate volumetric heat maps efficiently. In the fusion cell, we classify all body parts into several groups and estimate each group of parts with a distinct head network.
- Our part-aware 3D pose estimator is both compact and efficient. It achieves state-of-the-art accuracy on both the single-person and multi-person 3D human pose benchmarks using much fewer parameters and FLOPs.

2 Related Work

3D Human pose estimation has been studied widely in the past. In this section, we only focus on previous works that can be relevant to our work.

Estimate 3D poses from 2D Joints: Some approaches divide the task of 3D human pose estimation into first predicting 2D joint locations and then back-projecting them to estimate 3D human poses. The practice of inferring 3D human poses from their 2D projections can be traced back to the classic work [27]. Given the bone lengths, the problem boils down to a binary decision tree where each branch corresponds to two possible states of a joint concerning its parent. Jiang *et al.* [20] generate a set of hypothesis of 3D poses using Taylor's algorithm [50] and use them to query a large database of motion capture data to find the nearest neighbor. Similarly, the idea of exploiting nearest neighbor queries has been revisited by [15]. Chen *et al.* [6] also share the idea of using the detected 2D pose to query a large database of exemplary poses. Another common approach [3,63] is to learn an over-complete dictionary of basis 3D poses from a large database of motion capture data. Moreno-Noguer *et al.* [36] employ the pair-wise distance matrix of 2D joints to learn a distance matrix for 3D joints. Martinez *et al.* [32] design a fully-connected network to estimate 3D joint locations relative to the pelvis from 2D poses. Hossain *et al.* [16] exploit temporary information to calculate a sequence of 3D poses from a sequence of 2D joint locations. Ci *et al.* [10] combine the advantage of graph convolution network and fully-connected network and equip the model with strong generalization power. Cai *et al.* [5] introduce a graph-based local-to-global network to recover 3D poses from 2D pose sequences. These methods focus on estimating 3D poses from 2D poses, and we attempt to estimate 3D poses from monocular images.

Estimate 3D poses from Monocular Images: Recently, many approaches have been proposed to estimate 3D poses from monocular images in an end-to-end fashion. Li *et al.* [28] and Park *et al.* [38] exploit the 2D pose information to benefit 3D pose estimation. Rogez *et al.* [44] and Varol *et al.* [53] augment the training data with synthetic images and train CNNs to predict 3D poses from real images. Sun *et al.* [47] adopt a reparameterized pose representation using bones instead of joints. Pavlakos *et al.* [40] extend 2D heat maps to 3D volumetric heat maps and predict per voxel likelihood for each joint. Tome *et al.* [51] generalize Convolutional Pose Machine (CPM) [55] to the task of monocular 3D human pose estimation. Chen *et al.* [9] propose to decompose the volumetric representation into 2D depth-aware heat maps and joint depth estimation. Zhou *et al.* [65] propose a weakly-supervised transfer learning method that uses mixed 2D and 3D labels in a unified deep neural network. By introducing a simple integral operation, Sun *et al.* [48] unify heat maps learning and regression learning for pose estimation. Kocabas *et al.* [25] propose to train the 3D pose estimator with the multi-view triangulation in a self-supervised manner. Instead of estimating root-relative 3D poses, Moon *et al.* [35] propose to estimate 3D poses in the camera coordinate system directly. More recent works [1,21,22,26,37] tend to focus on reconstructing fine-grained 3D human shapes. Nevertheless, all works are limited in estimating all body parts with a single head network, and we attempt to search part-specific head networks for different body parts.

3 The Proposed Approach

In the literature of NAS, differential architecture search (DARTS) [30] is a representative method that can search effective network architectures using fewer computing resources. Therefore, we build our model on DARTS. First, we introduce some basic knowledge about DARTS. Then, we describe our approach to search part-specific head networks for intrinsically heterogeneous body parts.

3.1 Preliminaries: Differential Architecture Search (DARTS)

The framework of DARTS decomposes the searched network architecture into a number of (L) computational cells. There are two types of cells: the normal cell and the reduction cell. Both of them have typical convolution architectures to transform feature maps. Additionally, the reduction cell has another function to downsample the feature map. Each computational cell can be represented as a directed acyclic graph (DAG), consisting of an ordered sequence of N nodes ($\mathcal{N} = \{x^{(i)}|i = 1, ..., N\}$). In the DAG, each node $x^{(i)}$ ($i \in \{1, ..., N\}$) is a hidden representation (*i.e.,* feature map), and each edge $o^{(i,j)}(\cdot)$ denotes the transformation from $x^{(i)}$ to $x^{(j)}$ and is associated with an operation (*i.e.*, pooling and convolution). In each cell, there are two input nodes (*i.e.*, $x^{(1)}$ and $x^{(2)}$ receive outputs from the previous two cells) and one output node $x^{(N)}$ (*i.e.*, the concatenation of all intermediate nodes ($x^{(3)}, x^{(4)}, ..., x^{(N-1)}$)). The output of an intermediate node $x^{(j)}$ is computed as:

$$x^{(j)} = \sum_{i<j} o^{(i,j)}(x^{(i)}) \tag{1}$$

Where the node $x^{(i)}$ is one predecessor of the node $x^{(j)}$. There is a pre-defined space of operations denoted by $\mathcal{O}$, each element of which is a fixed operation (*e.g.,* identity connection, convolution and max pooling). In the search stage, our goal is to automatically select one operation from $\mathcal{O}$ and assign the operation to $o^{(i,j)}(\cdot)$ for each pair of nodes.

The core idea of DARTS is to make the search space continuous, and formulate the choice of an operation as a softmax over all possible operations:

$$\bar{o}^{(i,j)}(x) = \sum_{o\in\mathcal{O}} \frac{exp(\alpha^{o}_{i,j})}{\sum_{o'\in\mathcal{O}} exp(\alpha^{o'}_{i,j})} o(x) \tag{2}$$

Where $\alpha^{o}_{i,j}$ denotes the learnable score of the operation $o(\cdot)$ on the edge from $x^{(i)}$ to $x^{(j)}$. $\alpha_{i,j} \in \mathbb{R}^{|\mathcal{O}|}$ represents the scores of all candidate operations over the edge. The architecture of a cell is denoted as $\alpha = \{\alpha_{i,j}\}$, consisting of $\alpha_{i,j}$ for all edges connecting pairs of nodes. Then, DARTS formulates architecture search as finding α to minimize the loss function on the validation set:

$$\min_{\alpha} \; L_{val}(w^*(\alpha), \alpha) \tag{3}$$

$$\text{s.t. } w^*(a) = \text{argmin}_{\text{w}} \; L_{train}(w, \alpha) \tag{4}$$

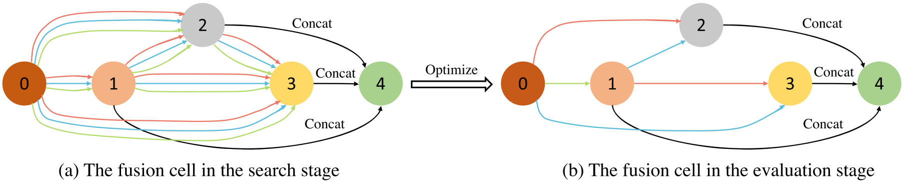
Where $w^*(\alpha)$ denotes the network weights associated with the architecture α, which is optimized on the training set. The architecture parameter α can be optimized via gradient descent by approximating Eq. 3 as:

$$\nabla_{\alpha} L_{val}(w^*(\alpha), \alpha) \approx \nabla_{\alpha} L_{val}(w - \xi \nabla_{w} L_{train}(w, \alpha), \alpha) \tag{5}$$

Where w denotes the current network weights, $\nabla_w L_{train}(w, \alpha)$ is the a gradient step of w and ξ is the step's learning rate. When we finish optimizing α in the search stage, we assign $o^{(i,j)}(\cdot)$ with the most likely operation candidate according to $\alpha^{(i,j)}$. For each intermediate node in a computational cell, DARTS retains its two strongest predecessors.

3.2 DARTS for Monocular 3D Human Pose Estimation

Since the framework of DARTS is originally designed for image classification, neither the normal cell nor the reduction cell can increase the resolution of feature maps. However, it is a common practice for 3D pose estimators to upsample feature maps from the size of 8×8 to the size of 64×64 consecutively and generate volumetric heat maps for all body parts. To this end, as shown in Fig. 2, we propose to introduce another type of cell, namely fusion cell, in the context of DARTS. It can upsample and transform feature maps propagated from previous cells. Just like the reduction cell performs downsampling at input nodes, the fusion cell also upsamples feature maps at input nodes as a preprocessing step. Then, we employ edges between two nodes (*i.e.*, convolution, pooling, *etc.*) to transform upsampled feature maps and produce volumetric heat maps for all parts at the output node. As shown in Fig. 2, it is interesting to note that the output node is the concatenation of all intermediate nodes and each intermediate node represents volumetric heat maps for a certain group of body parts. Through intermediate nodes in the fusion cell, we automatically divide all body parts into several groups, and the number of groups is equal to the number of intermediate nodes in the fusion cell. As shown in Fig. 2(a), there exist many candidate operations between nodes in the search stage, and we obtain the optimized architecture upon finishing the search process. In the optimized architecture shown in Fig. 2(b), we can observe that each intermediate node has been transformed by a different set of operations. In other words, we learn part-specific architectures in the search stage and employ them to estimate different groups of body parts in the evaluation stage.



(a) The fusion cell in the search stage

(b) The fusion cell in the evaluation stage

Fig. 2. An illustration of the fusion cell. Node 0 is the input node, and Node 1, 2, 3 are intermediate nodes. Node 4 is the output node and concatenates all intermediate nodes. Each edge represents one operation between two nodes. For simplicity, we only draw one input node here instead of two.

We follow a popular baseline [48] to build our part-aware 3D pose estimator. It predicts per voxel likelihood for each part and uses the soft-argmax operator to extract the 3D coordinate from the volumetric heat map. Instead of using ResNet-50 backbone and deconvolution layers, we search the whole network architecture. In the search stage, we stack the normal cell, the reduction cell, the fusion cell to construct our model with a total of N_c cells. We fix the number of reduction cells and fusion cells to N_r and N_f, respectively. Because the fusion cell is designed to generate volumetric heat maps at last, we first interweave $(N_c - N_r - N_f)$ normal cells and N_r reduction cells. Following the original DARTS, we organize the position of the reduction cell as:

$$P_r^i = \text{floor}(\frac{N_c - N_f}{N_r + 1}) \times i + 1 \tag{6}$$

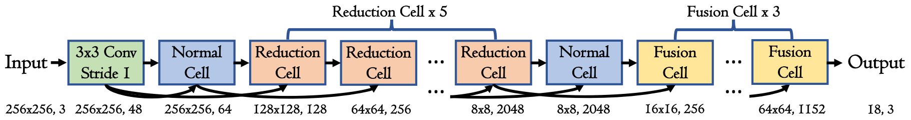


Fig. 3. An overview of our network architecture. We take the 256 × 256 input image as an example. It consists of ten computation cells: two normal cells, five reduction cells, and three fusion cells. The architecture of all types of cells are optimized in the search stage, and each cell receives inputs from the outputs of the previous two cells.

Where $i \in \{1, 2, ..., N_r\}$ denotes the i^{th} reduction cell. P_r^i denotes the position of the i^{th} reduction cell. floor$(\cdot)$ represents the function that discards the decimal point of a given number. After arranging normal cells and reduction cells, we append N_f fusion cells behind them. In the search stage, our model has a total of ten cells. We set N_r and N_f as 5 and 3, respectively. As illustrated in Fig. 3, out of the top seven cells, we interweave two normal cells and five reduction cells. Then, we append three fusion cells consecutively behind them to generate volumetric heat maps for all parts. We employ L1 loss to supervise estimated 3D poses and update network parameters w on the training set and architectures for all types of cells α on the validation set alternately.

When we finish the search process, we obtain the optimized normal cell, reduction cell, and fusion cell, as in Fig. 2(b). To evaluate the effectiveness of our searched architectures, we re-train our model constructed with these optimized cells. When our model is built with ten computational cells, the overview of its architecture is the same as what it was in the search stage. As shown in Fig. 3, given an input image, it first goes through a 3×3 convolution layer and a normal cell to generate the feature map. Then, we append five consecutive reduction cells to downsample the feature map and double its channel with a total stride of 2^5. After a series of reduction cells, the feature map is $8 \times 8 \times 2048$ in size, and we use a normal cell to refine it further. To generate the volumetric heat map, we use the proposed fusion cell to upsample the feature map. Except for the last one, we set the output channel of remaining fusion cells to 256 as a common practice. Three consecutive fusion cells upsample the feature map with a total stride of 2^3 and generate the volumetric heat map of size $64 \times 64 \times 64$ for all body parts. For each part, we extract its 3D coordinate from the corresponding volumetric heat map via the differential soft-argmax operation [48]. As we do in the search stage, we still employ L1 loss to train our model.

4 Experimental Evaluation

In this section, we present a detailed evaluation of our proposed approach. First, we introduce main benchmarks and present our experimental settings. Then, we conduct rigorous ablation analysis about our approach. Finally, we build our strongest part-aware estimator upon the knowledge obtained in ablation studies and compare it with state-of-the-art performance.

4.1 Main Benchmarks and Evaluation Metrics

Human3.6M Dataset [18]**:** It is captured in a calibrated multi-view studio and consists of 3.6 millions of video frames. Eleven subjects are recorded from four camera viewpoints, performing 15 activities. Previous works widely use two evaluation metrics. The first one is mean per joint position error (MPJPE), which first aligns the pelvis joint between estimated and ground-truth 3D poses and computes the average joint error among all human joints. The second metric uses Procrustes Analysis (PA) to align MPJPE further, and it is called PA MPJPE.

MuCo-3DHP and MuPoTS-3D Datasets [34]**:** These datasets are designed for multi-person 3D pose estimation. The training set is the MuCo-3DHP dataset, and it is generated by compositing the MPI-INF-3DHP dataset [33]. MuPoTS-3D dataset acts as the test set and contains 20 in-the-wild scenes. The evaluation metric is the 3D percentage of correct keypoints (3DPCK).

4.2 Experimental Settings and Implementation Details

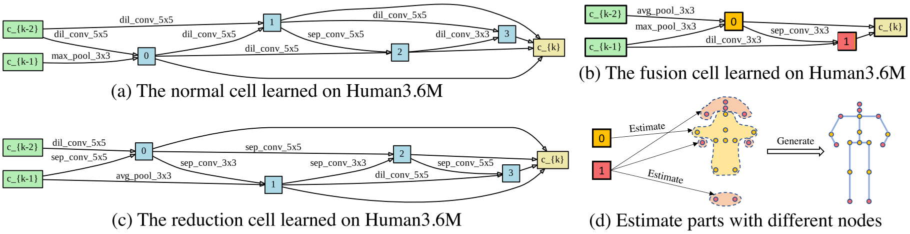
Human3.6M Dataset: Two evaluation protocols are widely used. Protocol 1 uses six subjects (S1, S5, S6, S7, S8, S9) in training and reports the evaluation result on every 64^{th} frame of Subject 11's videos using PA MPJPE. Protocol 2 uses six subjects (S1, S5, S6, S7, S8) in training and reports the evaluation result on every 64^{th} frame of two subjects (S9, S11) using MPJPE. In the evaluation stage of our approach, we use additional MPII [2] 2D pose data during training.

In the search stage, we train the network only with Human3.6M data. We split three subjects (S1, S5, S6) as the training set to update the network parameter w and use two subjects (S7, S8) as the validation set to update the network architecture α. We include following eight operations in the pre-defined space $\mathcal{O}$: 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, identity and *zero*.

MuCo-3DHP and MuPoTS-3D Datasets: We create 400K composite frames of the MuCo-3DHP dataset, of which half are without appearance augmentation. We use additional COCO [29] 2D pose data during training.

Table 1. Quantitative evaluation of the number of intermediate nodes within each fusion cell on Human3.6M using Protocol 2. N_i denotes the number of intermediate nodes within each fusion cell. Lower is better, best in bold, second-best underlined.

Methods	Search space		Params	FLOPs	Direct.	Dicuss	Eating	Greet	Phone	Pose
	dil. conv.	sep. conv.								
Ours, $N_i = 1$	✓	✓	14.7M	22.9G	52.6	60.9	50.8	54.3	62.0	53.4
Ours, $N_i = 2$	✓	✓	13.0M	10.7G	**46.3**	**55.3**	**47.2**	**49.0**	**55.0**	**48.2**
Ours, $N_i = 3$	✓	✓	9.9M	**7.8G**	53.5	62.2	54.1	56.5	62.7	55.5
Ours, $N_i = 4$	✓	✓	**9.9M**	7.9G	50.8	60.0	53.2	53.3	60.7	50.8
Ours, $N_i = 2$	–	✓	15.9M	12.8G	55.8	61.3	52.5	55.3	63.0	54.6
Ours, $N_i = 2$	✓	–	10.4M	8.7G	52.0	60.0	51.4	53.9	61.5	52.6
Methods	Purch.	Sitting	SitD.	Smoke	Photo	Wait	Walk	WalkD.	WalkT.	Avg
Ours, $N_i = 1$	56.0	68.8	76.7	60.0	65.8	53.8	44.6	62.7	51.8	58.9
Ours, $N_i = 2$	**52.6**	**64.6**	**70.8**	**54.4**	**60.0**	**48.8**	**40.9**	**58.3**	**46.7**	**53.6**
Ours, $N_i = 3$	59.0	73.1	81.5	60.7	66.9	55.8	46.9	63.7	53.3	60.9
Ours, $N_i = 4$	55.9	69.8	74.3	58.7	64.8	53.0	43.4	61.2	49.5	58.0
Ours, $N_i = 2$	57.0	69.9	76.8	61.3	67.6	54.4	45.9	64.2	52.9	59.9
Ours, $N_i = 2$	56.6	68.8	76.7	59.9	66.3	53.5	44.8	62.6	51.6	58.7



(a) The normal cell learned on Human3.6M

(b) The fusion cell learned on Human3.6M

(c) The reduction cell learned on Human3.6M

(d) Estimate parts with different nodes

Fig. 4. Cells found on Human3.6M dataset when we set N_i to 2. Our model uses two intermediate nodes encoded in the fusion cell to estimate different groups of body parts.

Implementation Details: In the search stage, to save GPU memory, we set the size of the input image and the volumetric heat map to 128 × 128 and $32 \times 32 \times 32$, respectively. The total training epoch is 25, and the parameter w is updated by the Adam optimizer [24] with a batch size of 40. The initial learning rate is 1×10^{-3} and reduced by a factor of 10 at the 15^{th} and the 20^{th} epoch. We start to optimize the network architecture α at the 8^{th} epoch. Its learning rate and weight decay are 8×10^{-4} and 3×10^{-4}, respectively. The search process lasts two days on a single NVIDIA TITAN RTX GPU. In the evaluation stage, the size of the input image and the volumetric heat map are 256 × 256 and 64 × 64 × 64, respectively. The total epoch is 20. We train our network with Adam with a batch size of 64. The initial learning rate is 1×10^{-3} and reduced by ten at the 12^{th} and the 16^{th} epoch. Training samples are augmented via rotation (±30°), horizontal flip, color jittering, and synthetic occlusion [46]. The training process takes two days on four NVIDIA P100 GPUs. We run each experiment three times with different random seeds, and the confidence interval is about ±0.3 mm.

4.3 Ablation Experiments

The Number of Intermediate Nodes in the Fusion Cell
As we explain in Sect. 3, the number of intermediate nodes in the fusion cell is equal to the number of groups that we divide all body parts into. In this set of experiments, by adjusting the number of intermediate nodes, we are motivated to explore how many groups all body parts are divided into is an optimal choice. In the search stage, we optimize the network architecture where the fusion cell can have $N_i \in \{1, 2, 3, 4\}$ intermediate nodes, and the model has a total of ten computational cells, as in Fig. 3. In Table 1, we can observe that the model with two intermediate nodes outperforms all the others on every action. Compared to dividing all parts into more or fewer groups, it achieves a better trade-off between performance and computational complexity. With only 13.0M parameters and 10.7G FLOPs, it encouragingly reduces MJPJE to 53.6 mm.

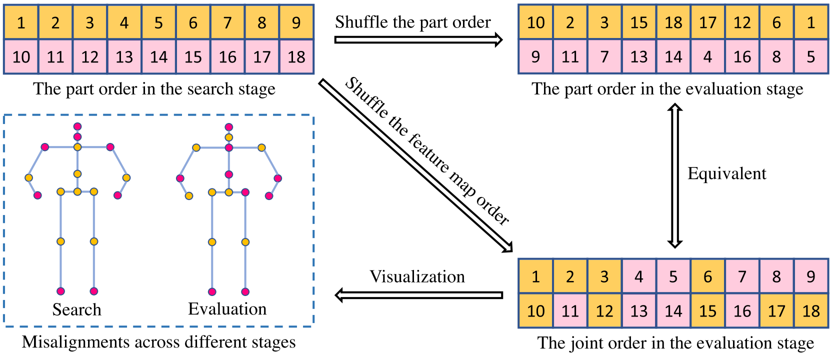


Fig. 5. Illustration of the equivalence between shuffling the part order and shuffling the heat map order. The number in the box denotes the part id. There are a total of eighteen parts. As shown in Fig. 4(d), within the last fusion cell, orange boxes indicate parts estimated by Node 0, and pink boxes indicate ones estimated by Node 1. (Color figure online)

To investigate what makes our architecture efficient when N_i is 2, we visualize searched architectures in Fig. 4. As a comparison, when N_i is 1, our model estimates all body parts with a single head network. It is computationally intensive, having 14.7M parameters and 22.9G FLOPs, but its performance is not satisfactory. Towards a better solution shown in Fig. 4(d), we employ two intermediate nodes encoded in the fusion cell to estimate the torso and limbs, respectively. Specifically, Node 0 is transformed from pooling layers and is robust to estimate parts with relatively low DOFs. On the other side, dilated convolutional layers empower Node 1 to capture long-range context information, which is helpful to estimate parts with higher DOFs, such as the wrist and ankle. The normal cell, shown in Fig. 4(a), consists of many dilated convolutional layers, which greatly increase the receptive field of our model, and are critical to performance improvement. As shown in Table 1, if we remove dilated convolution from our search space $\mathcal{O}$, our searched model has more parameters and FLOPs, and its performance drops from 53.6 mm to 59.9 mm. The reduction cell employs many depth-wise convolution layers to fuse multi-scale features efficiently. Similarly, we validate their importance by removing these operations from $\mathcal{O}$, and it leads to a 5.1 mm decline in performance.

The Part-Awareness of Our Model

We begin to validate the part-awareness of our approach from two perspectives. First, to investigate whether searched head networks are part-specific, we intend to shuffle the order of parts when we re-train our model in the evaluation stage. However, it is a little troublesome to do this since we would have to modify the data augmentation policy according to the shuffled order. Alternatively, as shown in Fig. 5, we propose to shuffle the order of heat maps produced in the last fusion cell. The implementation of the shuffle operation is the same as ShuffleNet [59], which is efficient and GPU-friendly. If our model trained with the shuffled order behaves obviously worse than the original one, we can validate that our optimized head networks are part-aware. We run experiments three times and train our model with different shuffled orders. As shown in Table 2, we observe that all models trained with shuffled orders suffer from a significant drop in performance, more than 3 mm in MPJPE. As we take a closer look, the decline in performance also reflects on every individual part, especially parts with higher DOFs (*e.g.,* ankle, knee), and their estimation accuracy might drop by more than 5 mm. By comparing models trained with shuffled orders, we validate that our approach learns part-specific head networks for specific body parts in the search stage.

In our model, the fusion cell plays a pivotal role in learning part-specific head networks. To evaluate the importance of the fusion cell, we replace them with deconvolution layers and only search the backbone network. The backbone network only consists of normal cells and reduction cells. For a fair comparison, all constructed networks have two normal cells and five reduction cells, and their

Table 2. Quantitative evaluation of the shuffled part order on Human3.6M using Protocol 2. We set N_c and N_i to 10 and 2 respectively. We compute part-wise MPJPE to report performance. Bold values indicate parts estimated by Node 0 and italic values denote ones estimated by Node 1.

Methods	Pelvis	R Hip	R Kn.	R An.	L Hip	L Kn.	L An.	Torso	Neck
Ours, original	**0.0**	**23.2**	**53.4**	**74.4**	**22.6**	**47.3**	**75.5**	**37.2**	**44.7**
Ours, shuffled 1	**0.0**	*24.3*	**56.2**	*78.3*	**23.7**	*50.3*	**78.6**	*38.9*	**48.3**
Ours, shuffled 2	**0.0**	**25.1**	*56.1*	**83.8**	**24.9**	*52.1*	**82.8**	**39.4**	*46.5*
Ours, shuffled 3	**0.0**	**24.3**	**58.2**	*81.7*	*24.2*	*53.6*	**82.7**	**40.4**	**45.4**

Methods	Nose	Head	L Sh.	L El.	L Wr.	R Sh.	R El.	R Wr.	Avg
Ours, original	*46.9*	*50.8*	*51.7*	*72.1*	*92.2*	*50.6*	*76.0*	*93.9*	53.6
Ours, shuffled 1	*53.7*	**57.8**	*57.0*	**74.7**	*95.3*	**55.4**	*80.1*	**98.2**	57.2
Ours, shuffled 2	**50.7**	*55.7*	*54.2*	**74.1**	*93.3*	*53.5*	**79.9**	*96.0*	56.9
Ours, shuffled 3	*49.3*	*53.1*	*55.0*	**75.8**	**95.3**	**54.2**	*80.8*	*97.7*	57.1

Table 3. Quantitative evaluation of the importance of the fusion cell on Human3.6M using Protocol 2. BS and WS denote the backbone search and the whole architecture search, respectively. We compute action-wise MPJPE to report the network performance. Lower is better, best in bold, second-best underlined.

Methods	Backbone	Pretrain	Params	FLOPs	Direct.	Dicuss	Eating	Greet	Phone	Pose
Ours, ResNet	ResNet50	✓	34.3M	14.1G	50.8	**52.3**	54.8	57.9	**52.8**	**47.0**
Ours, BS	Searched	–	20.5M	12.5G	49.0	59.9	49.8	53.5	58.0	51.0
Ours, WS	–	–	**13.0M**	**10.7G**	**46.3**	55.3	**47.2**	**49.0**	55.0	48.2
Methods	Purch.	Sitting	SitD.	Smoke	Photo	Wait	Walk	WalkD.	WalkT.	Avg
Ours, ResNet	**52.1**	**62.0**	73.7	**52.6**	**58.3**	50.4	40.9	**54.1**	**45.1**	53.9
Ours, BS	56.0	65.8	77.5	56.3	63.8	52.9	44.4	62.7	50.0	57.1
Ours, WS	52.6	64.6	**70.8**	54.4	60.0	**48.8**	**40.9**	58.3	46.7	**53.6**

only difference is whether they have fusion cells. In Table 3, compared to the backbone search, searching the whole network architecture improves performance by 3.5 mm and reduces 37% parameters and 14% FLOPs. In comparison with the model built on the commonly used ResNet-50 backbone, we advance estimation accuracy by 0.3 mm with 62% fewer parameters and 24% fewer FLOPs. Through our experiments, we show that fusion cells significantly contribute to the compactness and efficiency of our approach and exhibit more competitive performance over models using the ResNet-50 backbone.

The Number of Computational Cells

Instead of stacking only ten computation cells, we attempt to construct a deeper part-aware 3D pose estimator, according to Eq. 6. As shown in Table 4, as we increase the number of computational cells, our model becomes better in performance but has more parameters and FLOPs. When N_c is 20, our model achieves the best performance, 47.3 mm in MPJPE. As we increase N_c from 10 to 20, the gain in network parameters (from 13.0M to 20.4M) and FLOPs (from 10.7G to 14.1G) also leads to an improvement in performance (from 53.6 mm to 47.3 mm). This phenomenon also demonstrates that the network architecture optimized during the search process is computationally efficient.

Table 4. Quantitative evaluation of the number of cells on Human3.6M using Protocol 2. N_c denotes the number of computational cells. We compute action-wise MPJPE to report the network performance. Lower is better, best in bold, second-best underlined.

Methods	Params	FLOPs	Direct.	Discuss	Eating	Greet	Phone	Pose	Purch.
Ours, $N_c = 10$	**13.0M**	**10.7G**	46.3	55.3	47.2	49.0	55.0	48.2	52.6
Ours, $N_c = 15$	14.7M	12.7G	45.8	53.7	43.4	49.4	52.0	46.4	51.4
Ours, $N_c = 20$	20.4M	14.1G	**41.4**	**48.6**	**42.0**	**45.3**	**47.1**	**42.3**	**46.0**
Methods	Sitting	SitD.	Smoke	Photo	Wait	Walk	WalkD.	WalkT.	Avg
Ours, $N_c = 10$	64.6	70.8	54.4	60.0	48.8	40.9	58.3	46.7	53.6
Ours, $N_c = 15$	60.8	63.4	50.9	55.6	45.7	40.8	55.4	44.5	50.9
Ours, $N_c = 20$	**57.9**	**62.1**	**47.8**	**51.2**	**43.6**	**36.1**	**51.1**	**41.5**	**47.3**

Table 5. Comparison with state-of-the-art methods on Human3.6M using Protocol 1. S denotes our small part-aware model with ten cells, and L denotes our large model with twenty cells. Lower is better, best in bold, second-best underlined.

Methods	Dire.	Dis.	Eat	Gre.	Phe.	Pose	Pur.	Sit	SitD.	Smo.	Phot..	Wait	Walk	WD.	WT.	Ave.
Yasin [58]	88.4	72.5	108.5	110.2	97.1	81.6	107.2	119.0	170.8	108.2	142.5	86.9	92.1	165.7	102.0	108.3
Chen [6]	71.6	66.6	74.7	79.1	70.1	67.6	89.3	90.7	195.6	83.5	93.3	71.2	55.7	85.9	62.5	82.7
Moreno [36]	67.4	63.8	87.2	73.9	71.5	69.9	65.1	71.7	98.6	81.3	93.3	74.6	76.5	77.7	74.6	76.5
Zhou [64]	47.9	48.8	52.7	55.0	56.8	49.0	45.5	60.8	81.1	53.7	65.5	51.6	50.4	54.8	55.9	55.3
Sun [47]	42.1	44.3	45.0	45.4	51.5	43.2	41.3	59.3	73.3	51.0	53.0	44.0	38.3	48.0	44.8	48.3
Fang [12]	38.2	41.7	43.7	44.9	48.5	40.2	38.2	54.5	64.4	47.2	55.3	44.3	36.7	47.3	41.7	45.7
Sun [48]	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
Moon [35]	31.9	**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
Ours, S	31.8	33.4	38.9	37.9	36.4	36.6	32.6	36.2	47.8	38.9	43.0	32.6	26.5	39.8	30.8	36.4
Ours, L	**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**

4.4 Comparison with the State-of-the-Art

To demonstrate the effectiveness and the generalization ability of our approach, we conduct our experiments on both single-person and multi-person 3D pose estimation benchmarks. Previous works have different experimental settings, and we summarize comparison results in Tables 5, 6 and 7, respectively. In Fig. 6, we show qualitative results produced by our model with ten cells. It can generalize well for in-the-wild images, even on challenging poses and crowded scenes.

Single-Person 3D Human Pose Estimation: We compare our approach on Human3.6M with state-of-the-art methods in Tables 5 and 6. By reducing about 40% parameters, our large part-aware model advances the-state-of-the-art accuracy by 1.3 mm and 2.3 mm in protocol 1 and protocol 2, respectively. If we add supervision on intermediate feature maps, the performance of our small

Table 6. Comparison with state-of-the-art methods on Human3.6M using Protocol 2. S denotes our small part-aware model with ten cells, and L denotes our large model with twenty cells. Lower is better, best in bold, second-best underlined.

Methods	Dire.	Dis.	Eat	Gre.	Phe.	Pose	Pur.	Sit	SitD.	Smo.	Phot.	Wait	Walk	WD.	WT.	Ave.
Chen [6]	89.9	97.6	90.0	107.9	107.3	93.6	136.1	133.1	240.1	106.7	139.2	106.2	87.0	114.1	90.6	114.2
Tome [51]	65.0	73.5	76.8	86.4	86.3	68.9	74.8	110.2	173.9	85.0	110.7	85.8	71.4	86.3	73.1	88.4
Moreno [36]	69.5	80.2	78.2	87.0	100.8	76.0	69.7	104.7	113.9	89.7	102.7	98.5	79.2	82.4	77.2	87.3
Zhou [64]	68.7	74.8	67.8	76.4	76.3	84.0	70.2	88.0	113.8	78.0	98.4	90.1	62.6	75.1	73.6	79.9
Mehta [33]	57.5	68.6	59.6	67.3	78.1	56.9	69.1	98.0	117.5	69.5	82.4	68.0	55.3	76.5	61.4	72.9
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
Omran [37]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	59.9
Sun [47]	52.8	54.8	54.2	54.3	61.8	53.1	53.6	71.7	86.7	61.5	67.2	53.4	47.1	61.6	63.4	59.1
Kanazawa [47]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	56.8
Moon [35]	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
Sun [48]	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
Ours, S	46.3	55.3	47.2	49.0	55.0	48.2	52.6	64.6	70.8	54.4	60.0	48.8	40.9	58.3	46.7	53.6
Ours, L	**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**

Table 7. Comparison with state-of-the-art methods on MuPoTS-3D using all ground truths. S denotes our small part-aware model with ten cells, and L denotes our large model with twenty cells. Higher is better, best in bold, second-best underlined.

Methods	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Rogez [45]	67.7	49.8	53.4	59.1	67.5	22.8	43.7	49.9	31.1	78.1
Mehta [34]	81.0	60.9	64.4	63.0	69.1	30.3	65.0	59.6	64.1	83.9
Moon [35]	94.4	77.5	79.0	81.9	85.3	72.8	81.9	75.7	90.2	90.4
Ours, S	93.1	76.7	79.9	78.2	83.6	64.6	79.0	72.5	87.6	88.3
Ours, L	**95.8**	**80.2**	**81.3**	**84.6**	**87.1**	**74.5**	**82.7**	**79.4**	**91.2**	**93.3**
Methods	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
Rogez [45]	50.2	51.0	51.6	49.3	56.2	66.5	65.2	62.9	66.1	59.1
Mehta [34]	68.0	68.6	62.3	59.2	70.1	80.0	79.6	67.3	66.6	67.2
Moon [35]	79.2	79.9	75.1	72.7	81.1	89.9	89.6	81.8	81.7	76.2
Ours, S	76.1	79.4	71.1	70.6	77.7	86.6	87.1	80.3	79.5	72.0
Ours, L	**83.4**	**82.0**	**78.6**	**76.5**	**84.3**	**92.1**	**91.1**	**85.3**	**82.4**	**77.8**

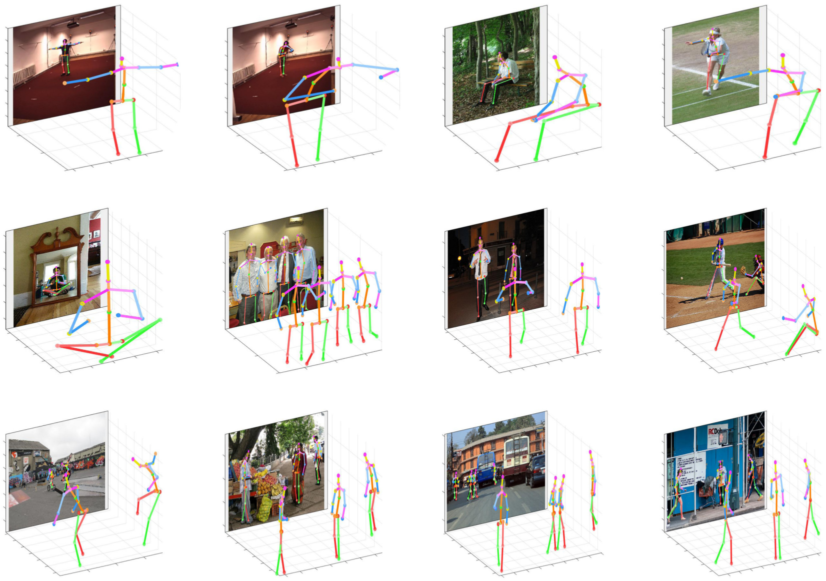


Fig. 6. Qualitative results on different datasets. Our small model produces convincing results even on challenging poses and crowded scenes.

model can be significantly improved, achieving 50.4 mm in Protocol 2. Moreover, our method is also compatible with some efficient learning frameworks [19, 25, 62].

Multi-person 3D Human Pose Estimation: For multi-person 3D pose estimation, we use RootNet [35] to estimate absolute depth for the root joint of each person. As shown in Table 7, we compare our model with previous state-of-the-art multi-person pose estimation methods on MuPoTS-3D, and our large part-aware 3D pose estimator achieves more superior performance on every sequence.

5 Conclusion and Future Works

In this work, we propose to estimate 3D poses of different parts with part-specific neural architectures. In the search stage, we optimize the architectures of different types of cells via gradient descent. Then, we interweave optimized computational cells to construct our part-aware 3D pose estimator, which is compact and efficient. Our model advances the state-of-the-art accuracy on both the single-person and multi-person 3D human pose estimation benchmarks. In the future, we attempt to explore other NAS methods to search 3D pose estimators in a larger space, which may open up the possibility for a global optimization.

Acknowledgements. This work is jointly supported by National Key Research and Development Program of China (2016YFB1001000), Key Research Program of Frontier Sciences, CAS (ZDBS-LY-JSC032), National Natural Science Foundation of China (61525306, 61633021, 61721004, 61806194, U1803261, 61976132), Shandong Provincial Key Research and Development Program (2019JZZY010119), HW2019SOW01, and CAS-AIR.

References

1. Alldieck, T., Pons-Moll, G., Theobalt, C., Magnor, M.: Tex2shape: detailed full human body geometry from a single image. In: ICCV (2019)
2. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: new benchmark and state of the art analysis. In: CVPR (2014)
3. Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., Black, M.J.: Keep it SMPL: automatic estimation of 3D human pose and shape from a single image. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 561–578. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_34
4. Cai, H., Zhu, L., Han, S.: ProxylessNAS: direct neural architecture search on target task and hardware. In: ICLR (2019)
5. Cai, Y., et al.: Exploiting spatial-temporal relationships for 3D pose estimation via graph convolutional networks. In: ICCV (2019)
6. Chen, C.H., Ramanan, D.: 3D human pose estimation = 2D pose estimation + matching. In: CVPR (2017)
7. Chen, L.C., et al.: Searching for efficient multi-scale architectures for dense image prediction. In: NeurIPS (2018)
8. Chen, Y., Yang, T., Zhang, X., Meng, G., Xiao, X., Sun, J.: DetNAS: backbone search for object detection. In: NeurIPS (2019)
9. Chen, Z., Guo, Y., Huang, Y., Liang, W.: Learning depth-aware heatmaps for 3D human pose estimation in the wild. In: BMVC (2019)
10. Ci, H., Wang, C., Ma, X., Wang, Y.: Optimizing network structure for 3D human pose estimation. In: ICCV (2019)
11. Fabbri, M., Lanzi, F., Calderara, S., Alletto, S., Cucchiara, R.: Compressed volumetric heatmaps for multi-person 3D pose estimation. In: CVPR (2020)
12. Fang, H., Xu, Y., Wang, W., Liu, X., Zhu, S.C.: Learning knowledge-guided pose grammar machine for 3D human pose estimation. In: AAAI (2018)
13. Ghiasi, G., Lin, T.Y., Le, Q.V.: NAS-FPN: learning scalable feature pyramid architecture for object detection. In: CVPR (2019)

14. Guo, Z., et al.: Single path one-shot neural architecture search with uniform sampling. In: NeurIPS (2019)
15. Gupta, A., Martinez, J., Little, J.J., Woodham, R.J.: 3D pose from motion for cross-view action recognition via non-linear circulant temporal encoding. In: CVPR (2014)
16. Hossain, M.R.I., Little, J.J.: Exploiting temporal information for 3D human pose estimation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11214, pp. 69–86. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01249-6_5
17. Howard, A., et al.: Searching for MobileNetV3. In: ICCV (2019)
18. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6M: large scale datasets and predictive methods for 3D human sensing in natural environments. In: TPAMI (2014)
19. Iskakov, K., Burkov, E., Lempitsky, V., Malkov, Y.: Learnable triangulation of human pose. In: ICCV (2019)
20. Jiang, H.: 3D human pose reconstruction using millions of exemplars. In: ICPR (2010)
21. Jiang, W., Kolotouros, N., Pavlakos, G., Zhou, X., Daniilidis, K.: Coherent reconstruction of multiple humans from a single image. In: CVPR (2020)
22. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: CVPR (2018)
23. Kasim, M., et al.: Up to two billion times acceleration of scientific simulations with deep neural architecture search. arXiv preprint arXiv:2001.08055 (2020)
24. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2014)
25. Kocabas, M., Karagoz, S., Akbas, E.: Self-supervised learning of 3D human pose using multi-view geometry. In: CVPR (2019)
26. Kolotouros, N., Pavlakos, G., Daniilidis, K.: Convolutional mesh regression for single-image human shape reconstruction. In: CVPR (2019)
27. Lee, H.J., Chen, Z.: Determination of 3D human body postures from a single view. In: CVGIP (1985)
28. Li, S., Chan, A.B.: 3D human pose estimation from monocular images with deep convolutional neural network. In: ACCV (2014)
29. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
30. Liu, C., et al.: Auto-DeepLab: hierarchical neural architecture search for semantic image segmentation. In: CVPR (2019)
31. Liu, H., Simonyan, K., Yang, Y.: DARTS: differentiable architecture search. In: ICLR (2019)
32. Martinez, J., Hossain, R., Romero, J., Little, J.J.: A simple yet effective baseline for 3D human pose estimation. In: ICCV (2017)
33. Mehta, D., et al.: Monocular 3D human pose estimation in the wild using improved CNN supervision. In: 3DV (2017)
34. Mehta, D., et al.: Single-shot multi-person 3D pose estimation from monocular RGB. In: 3DV (2018)
35. Moon, G., Chang, J.Y., Lee, K.M.: Camera distance-aware top-down approach for 3D multi-person pose estimation from a single RGB image. In: ICCV (2019)
36. Moreno-Noguer, F.: 3D human pose estimation from a single image via distance matrix regression. In: CVPR (2017)

37. Omran, M., Lassner, C., Pons-Moll, G., Gehler, P., Schiele, B.: Neural body fitting: unifying deep learning and model based human pose and shape estimation. In: 3DV (2018)
38. Park, S., Hwang, J., Kwak, N.: 3D human pose estimation using convolutional neural networks with 2D pose information. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 156–169. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_15
39. Pavlakos, G., Zhou, X., Daniilidis, K.: Ordinal depth supervision for 3D human pose estimation. In: CVPR (2018)
40. Pavlakos, G., Zhou, X., Derpanis, K.G., Daniilidis, K.: Coarse-to-fine volumetric prediction for single-image 3D human pose. In: CVPR (2017)
41. Pavlakos, G., Zhou, X., Derpanis, K.G., Daniilidis, K.: Harvesting multiple views for marker-less 3D human pose annotations. In: CVPR (2017)
42. Peng, J., Sun, M., ZHANG, Z.X., Tan, T., Yan, J.: Efficient neural architecture transformation search in channel-level for object detection. In: NeurIPS (2019)
43. Qiu, H., Wang, C., Wang, J., Wang, N., Zeng, W.: Cross view fusion for 3D human pose estimation. In: ICCV (2019)
44. Rogez, G., Schmid, C.: Mocap-guided data augmentation for 3D pose estimation in the wild. In: NeurIPS (2016)
45. Rogez, G., Weinzaepfel, P., Schmid, C.: LCR-Net: localization-classification-regression for human pose. In: CVPR (2017)
46. Sárándi, I., Linder, T., Arras, K.O., Leibe, B.: Synthetic occlusion augmentation with volumetric heatmaps for the 2018 ECCV PoseTrack challenge on 3D human pose estimation. In: ECCVW (2018)
47. Sun, X., Shang, J., Liang, S., Wei, Y.: Compositional human pose regression. In: ICCV (2017)
48. Sun, X., Xiao, B., Wei, F., Liang, S., Wei, Y.: Integral human pose regression. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11210, pp. 536–553. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01231-1_33
49. Tan, M., et al.: MnasNet: platform-aware neural architecture search for mobile. In: CVPR (2019)
50. Taylor, C.J.: Reconstruction of articulated objects from point correspondences in a single uncalibrated image. In: CVIU (2000)
51. Tome, D., Russell, C., Agapito, L.: Lifting from the deep: convolutional 3D pose estimation from a single image. In: CVPR (2017)
52. Tu, H., Wang, C., Zeng, W.: VoxelPose: towards multi-camera 3D human pose estimation in wild environment. In: Vedaldi A., Bischof H., Brox T., Frahm JM. (eds) ECCV, pp. 197–212 . Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_12
53. Varol, G., et al.: Learning from synthetic humans. In: CVPR (2017)
54. Wang, J., Huang, S., Wang, X., Tao, D.: Not all parts are created equal: 3D pose estimation by modeling bi-directional dependencies of body parts. In: ICCV (2019)
55. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR (2016)
56. Xu, Y., et al.: PC-DARTS: partial channel connections for memory-efficient architecture search. In: ICLR (2020)
57. Yang, W., Ouyang, W., Wang, X., Ren, J., Li, H., Wang, X.: 3D human pose estimation in the wild by adversarial learning. In: CVPR (2018)
58. Yasin, H., Iqbal, U., Kruger, B., Weber, A., Gall, J.: A dual-source approach for 3D pose estimation from a single image. In: CVPR (2016)

59. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: an extremely efficient convolutional neural network for mobile devices. In: CVPR (2018)
60. Zhang, Y., Qiu, Z., Liu, J., Yao, T., Liu, D., Mei, T.: Customizable architecture search for semantic segmentation. In: CVPR (2019)
61. Zhang, Z., Wang, C., Qin, W., Zeng, W.: Fusing wearable IMUs with multi-view images for human pose estimation: a geometric approach. In: CVPR (2020)
62. Zhou, K., Han, X., Jiang, N., Jia, K., Lu, J.: HEMlets pose: learning part-centric heatmap triplets for accurate 3D human pose estimation. In: ICCV (2019)
63. Zhou, X., Zhu, M., Leonardos, S., Derpanis, K.G., Daniilidis, K.: Sparseness meets deepness: 3D human pose estimation from monocular video. In: CVPR (2016)
64. Zhou, X., Zhu, M., Pavlakos, G., Leonardos, S., Derpanis, K.G., Daniilidis, K.: MonoCap: monocular human motion capture using a CNN coupled with a geometric prior. In: TPAMI (2018)
65. Zhou, X., Huang, Q., Sun, X., Xue, X., Wei, Y.: Towards 3D human pose estimation in the wild: a weakly-supervised approach. In: ICCV (2017)