International Journal of Computer Vision (2022) 130:56–75
https://doi.org/10.1007/s11263-021-01525-0

Learning a Robust Part-Aware Monocular 3D Human Pose Estimator via Neural Architecture Search

Zerui Chen[1,2] · Yan Huang[1] · Hongyuan Yu[1,2] · Liang Wang[1,2,3,4]

Received: 23 October 2020 / Accepted: 8 September 2021 / Published online: 26 October 2021

Abstract
Even though most existing monocular 3D human pose estimation methods achieve very competitive performance, they are limited in estimating heterogeneous human body parts with the same decoder architecture. In this work, we present an approach to build a part-aware 3D human pose estimator to better deal with these heterogeneous human body parts. Our proposed method consists of two learning stages: (1) searching suitable decoder architectures for specific parts and (2) training the part-aware 3D human pose estimator built with these optimized neural architectures. Consequently, our searched model is very efficient and compact and can automatically select a suitable decoder architecture to estimate each human body part. In comparison with previous state-of-the-art models built with ResNet-50 network, our method can achieve better performance and reduce 64.4% parameters and 8.5% FLOPs (multiply-adds). We validate the robustness and stability of our searched models by conducting extensive and rigorous ablation experiments. Our method can advance state-of-the-art accuracy on both the single-person and multi-person 3D human pose estimation benchmarks with affordable computational cost.

Keywords Monocular 3D human pose estimation · Heterogeneous human body parts · Neural architecture search

1 Introduction

3D human pose estimation is a very active research field in computer vision with widespread applications in human tracking, human action recognition, human-computer interaction, surveillance, robotics, virtual reality, etc. In the literature of 3D human pose estimation, different methods can be generally classified into two categories: monocular methods (Martinez et al. 2017; Pavlakos et al. 2017a; Mehta et al. 2017; Sun et al. 2018) and multi-view methods (Belagiannis et al. 2014; Qiu et al. 2019; Iskakov et al. 2019). Compared with multi-view methods, monocular 3D human pose estimation does not require multiple carefully calibrated cameras and is more flexible for deployment in outdoor environments. However, given its ill-posed nature, estimating 3D human poses from a single RGB image remains challenging.

In recent years, deep learning based methods (Rogez et al. 2017; Kocabas et al. 2019; Sun et al. 2017; Zhou et al. 2017; Pavlakos et al. 2018) have achieved significant progress in estimation accuracy. Many strong monocular methods are emerging and formulate the problem as joint coordinate regression (Martinez et al. 2017; Sun et al. 2017; Li and Chan 2014) or heat maps learning (Yang et al. 2018; Zhou et al. 2017; Nibali et al. 2019). Recently, many methods (Sun et al. 2018; Pavlakos et al. 2017a; Zhou et al. 2019; Varol et al. 2018; Zheng et al. 2019) have followed a popular paradigm in predicting per voxel likelihood for each human joint and achieved very competitive performance.

Before the deep learning era, many works show that it is effective to exploit part-based models for object detec-

Communicated by Gregory Rogez.

✉ Liang Wang
wangliang@nlpr.ia.ac.cn

Zerui Chen
zerui.chen@cripac.ia.ac.cn

Yan Huang
yhuang@nlpr.ia.ac.cn

Hongyuan Yu
hongyuan.yu@cripac.ia.ac.cn

1 Center for Research on Intelligent Perception and Computing, NLPR, CASIA, Beijing, China

2 School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

3 Center for Excellence in Brain Science and Intelligence Technology, CAS, Beijing, China

4 Artificial Intelligence Research, CAS, Beijing, China

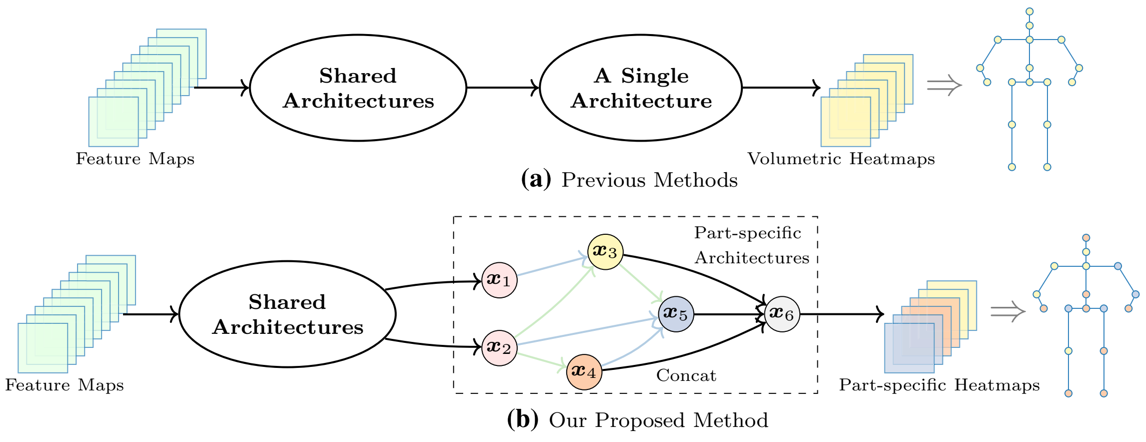


Fig. 1 Motivation. Most previous methods employ a single decoder architecture to deal with intrinsically heterogeneous human body parts (as shown in (**a**)). Instead, we are motivated to search for a suitable network architecture for a group of parts and estimate their 3D locations with part-specific neural architectures (as shown in (**b**)). Part-specific architectures consist of several nodes and edges connecting each pair of nodes. x_1, x_2 are input nodes. x_3, x_4, x_5 are intermediate nodes, which are used to generate part-specific volumetric heatmaps. x_6 is the output node and concatenates all intermediate nodes along the channel dimension. Lines with different colors indicate different transformation operations between each pair of nodes

tion (Felzenszwalb et al. 2009; Divvala et al. 2012) or human pose estimation (Burenius et al. 2013; Belagiannis et al. 2014). However, many current deep learning methods shown in Fig. 1a are limited in using a single decoder network architecture to predict 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. The task of human pose estimation is a multi-task learning problem (Tang et al. 2018; Tang and Wu 2019), and a single neural network architecture might be sub-optimal to deal with intrinsically heterogeneous human 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, distinct receptive fields). A recent effort (Wang et al. 2019) also demonstrates that it is effective to estimate different human body parts by explicitly taking their DOFs into account.

As illustrated in Fig. 1b, we approach the problem from a different angle and propose to estimate different human body parts with part-specific neural network architectures. However, looking for optimal decoder architectures for various human 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) (Liu et al. 2019b; Tan et al. 2019; Guo et al. 2020; Cai et al. 2019a; Howard et al. 2019; Xu et al. 2020) and propose to search part-specific decoder network architectures for different body parts. In fact, the idea of searching network architectures for certain tasks is not new. Specifically, it has been widely applied in semantic segmentation (Chen et al. 2018; Liu et al. 2019a; Zhang et al. 2019) and object detection (Ghiasi et al. 2019; Peng et al. 2019; Chen et al. 2019a).

However, directly applying NAS into 3D human pose estimation is non-trivial because current NAS methods mainly focus on some 2D visual tasks. Different from them, 3D human poses are commonly estimated in a higher-order volumetric space (Sun et al. 2018; Pavlakos et al. 2017a; Tu et al. 2020; Fabbri et al. 2020). 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 neural architecture search and achieve a trade-off between accuracy and computational cost 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 decoder network architectures, and each decoder network consists of a graph of different convolutional 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 decoder network. Considering the symmetry prior of the human body structure, it is inefficient to search a different decoder network for each part. Instead, our method classifies all human body parts into several groups and assigns a part-specific neural network architecture to each group. We propose two strategies to group different body parts. In the first strategy, we follow the default part order in the Human3.6M dataset (Ionescu et al. 2014) and evenly divide all parts into several groups. This strategy could roughly group parts according to their connectivity. In the second one, following (Tang and Wu 2019), we treat the location of each part as a random variable in 3D space and approximate the part distribution by calculating its 3D histogram. Then, we group related body parts based on their mutual information. This strategy can unevenly classify body parts into several groups and make our proposed method more flexible.

In the search stage of our method, all the neural network architectures, including the fusion cell, are optimized by gradient descent. Upon finishing the search stage, we stack these optimized computational cells to construct our part-aware 3D human pose estimator. At test time, our models can select optimized decoder networks encoded in the fusion cell to estimate different groups of human body parts. Benefitting from learning part-specific neural architectures, our models are very efficient and compact. Our small model has 64.4% fewer parameters and 8.5% FLOPs than previous state-of-the-art models and achieves 52.2 mm in Mean Per Joint Position Error (MPJPE) on the Human3.6M benchmark. With 44.0% fewer network parameters, our large model can outperform current state-of-the-art accuracy by 2.8 mm. We also conduct rigorous and extensive ablation experiments to further validate the robustness of our searched models.

A preliminary version (Chen et al. 2020) has been accepted by the 16th European Conference on Computer Vision as a spotlight. We extend it in four ways: (1) we employ another data-driven strategy to group related body parts according to their mutual information and further validate the effectiveness of our method. (2) To make our framework compatible with the data-driven part grouping strategy, we improve our method to search neural architectures for unevenly divided groups of body parts. (3) To search for better architectures, we empower our method to search 3D human pose estimator with a variable number of fusion cells. (4) We conduct extensive ablation experiments to validate the effectiveness of our searched models.

Our contributions can be summarized as follows:

- Our work shows that it is sub-optimal to estimate 3D poses of all human body parts with a single decoder network architecture. To the best of our knowledge, we make the first attempt to search part-specific architectures for estimating 3D poses of different body parts.
- We introduce the fusion cell to generate volumetric heat maps for all body parts efficiently. In the fusion cell, we classify all human body parts into several groups and estimate each group of body parts with a part-specific neural architecture.
- By conducting extensive ablation experiments, we show that our part-aware 3D human 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 estimation benchmarks using fewer parameters and FLOPs.

2 Related Works

3D human pose estimation has been studied widely in the past. In this section, we review some related works and discuss how they differ from our work.

2.1 Multi-view 3D Human Pose Estimation

Multi-view methods can infer 3D human poses from multiple calibrated cameras in good quality and often aim to generate ground-truth annotations for monocular 3D human pose estimation (Pavlakos et al. 2017b; Kocabas et al. 2019; Rhodin et al. 2018; Joo et al. 2015). Some previous works (Burenius et al. 2013; Belagiannis et al. 2014) represent the 3D human body structure as a probabilistic body model and optimize the model parameters to align the projections of the body model with the image features. Current state-of-the-art methods are motivated to combine the multiple-view geometry with popular deep learning systems. Kocabas et al. (2019) propose to generate pseudo 3D human pose labels by triangulating multi-view 2D human poses and train a 3D human pose estimator in a semi-supervised way. Qiu et al. (2019) propose to fuse image features along epipolar lines, leading to more robust 2D human pose estimation results, and present a recursive pictorial structure model to recover the 3D human pose from multi-view 2D human poses. Iskakov et al. (2019) propose an end-to-end differentiable method to aggregate multi-view 2D heat maps into a global volumetric heat map and achieve current state-of-the-art estimation accuracy for multi-view methods. While multi-view methods can always produce high-quality results, they often require multiple cameras commonly set up in indoor environments. Unlike these methods mentioned above, our method falls in the scope of monocular 3D human pose estimation and is more convenient when applied in in-the-wild scenes.

2.2 3D Human Pose Estimation from Depth Maps

Just like multi-view images, depth maps can also provide partial 3D information about the human structure. Many previous efforts fall in the scope of estimating 3D human poses or 3D hand poses from a single depth image. Ganapathi et al. (2010) propose an efficient filtering framework that combines an accurate generative model with a discriminative model and achieves real-time human motion capture. Shotton et al. (2011) design an intermediate body parts representation that could map the task of pose estimation into a simpler pixel-level classification problem. This formulation allows the classifier to make robust estimations for different body parts. To complete existing datasets with more camera perspectives, shapes and pose variations, Baek et al. (2018) propose to synthesize data in the skeleton space, enabling a more flexible way to manipulate data entries. Then, they synthesize corresponding depth maps from skeleton entries by training a separate hand pose generator. Mueller et al. (2019) propose to learn a dense surface correspondence predictor combined with a parametric mesh model (Romero et al. 2017) and perform real-time hand shape recovery from depth images via optimization. To enhance the generalization ability of pose estimators, Xiong et al. (2019) propose an anchor-based approach to discover informative anchor points towards a certain body part and achieve more superior estimation accuracy. Despite significant progress in these methods, many depth sensors are still not robust in in-the-wild environments. We aim to recover 3D human poses from a single RGB camera in this work since these sensors are ubiquitous.

2.3 3D Human Pose Estimation from 2D Joints

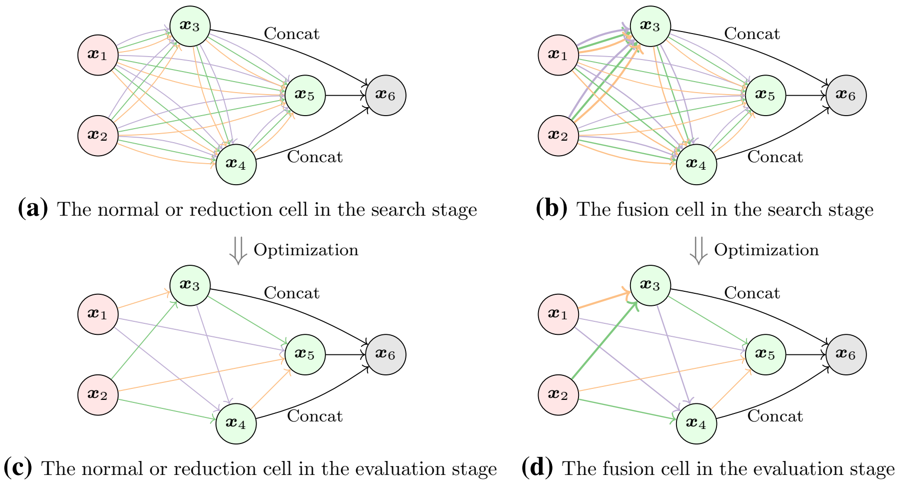
Some methods divide the task of 3D human pose estimation into first predicting 2D joint locations and then back-projecting them to estimate 3D human poses. The practice of inferring 3D human poses from their 2D projections can be traced back to the classic work (Lee and Chen 1985). Given the bone lengths, the problem boils down to a binary decision tree where each branch corresponds to two possible states of a joint concerning its parent. Jiang (2010) generate a set of 3D pose hypothesis using Taylor's algorithm (Taylor 2000) and use them to query an extensive database of motion capture data to find the nearest neighbor. Similarly, the idea of exploiting nearest neighbor queries has been revisited by (Gupta et al. 2014). Chen and Ramanan (2017) also share the idea of using the detected 2D human pose to query a large database of exemplary poses. Some other common methods (Zhou et al. 2016; Bogo et al. 2016) attempt to learn an over-complete dictionary of basis 3D human poses from a large database of motion capture data. Moreno-Noguer (2017) employ the pair-wise distance matrix of 2D joints to learn a distance matrix for 3D joints. Martinez et al. (2017) design a simple fully-connected network to estimate 3D joint locations relative to the pelvis from 2D human poses. Hossain and Little (2018) exploit temporary information to calculate a sequence of 3D human poses from a sequence of 2D joint locations. Ci et al. (2019) propose to combine the advantage of graph convolution network and fully-connected network and equip the model with stronger generalization power. Cai et al. (2019b) introduce a graph-based local-to-global network to recover 3D human poses from 2D human pose sequences. These methods focus on estimating 3D human poses from 2D human poses, and we attempt to estimate 3D poses directly from monocular images.

2.4 Monocular 3D Human Pose Estimation

Recently, many methods have been proposed to estimate 3D human poses from monocular images in an end-to-end fashion. Some previous works (Li and Chan 2014; Park et al. 2016) exploit the 2D human pose information to benefit 3D human pose estimation. Rogez and Schmid (2016); Varol et al. (2017) propose to augment the training data with synthetic images and train CNNs to infer 3D human poses from in-the-wild images. Sun et al. (2017) adopt a reparameterized pose representation using bones instead of joints and achieve superior results. Pavlakos et al. (2017a) extend 2D heat maps to 3D volumetric heat maps and predict per voxel likelihood for each joint. Tome et al. (2017) generalize Convolutional Pose Machine (CPM) (Wei et al. 2016) to the task of monocular 3D human pose estimation. Chen et al. (2019b) propose to decompose the volumetric representation into 2D depth-aware heat maps and joint depth estimation. Mehta et al. (2017) propose to generalize the 3D human pose estimator to in-the-wild scenes through the transfer of learned features. Zhou et al. (2017) 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. (2018) unify heat maps learning and regression learning for human pose estimation. Not limited to estimating root-relative 3D human poses, Moon et al. (2019) introduce the RootNet to recover the absolute depth for each joint and estimate 3D human poses in the camera coordinate system directly. Sárándi et al. (2020) propose metric-scale truncation-robust volumetric heat maps to resolve scale ambiguity in 3D human pose estimation. More recent works (Kanazawa et al. 2018; Omran et al. 2018; Jiang et al. 2020; Kolotouros et al. 2019; Alldieck et al. 2019; Natsume et al. 2019) tend to focus on reconstructing fine-grained 3D human shapes. Nevertheless, all previous works are limited in estimating all human body parts with a single neural network architecture. We attempt to search for suitable neural network architectures for different human body parts.

3 The Proposed Approach

In the literature of NAS, differentiable neural architecture search (DARTS) (Liu et al. 2019a) is a representative method that can search efficient neural network architectures using much fewer computing resources. Therefore, we build our proposed model on DARTS. In the following section, we first introduce some basic knowledge about DARTS. Then, we describe our method that makes it possible to search part-specific neural network architectures for intrinsically heterogeneous human body parts. Finally, we propose two strategies to classify different body parts into several groups and search for suitable architectures for each group.



x_3
x_1
x_2
x_4
x_5
x_6
Concat
Concat
(a) The normal or reduction cell in the search stage
(b) The fusion cell in the search stage
⇓ Optimization
⇓ Optimization
(c) The normal or reduction cell in the evaluation stage
(d) The fusion cell in the evaluation stage

Fig. 2 An illustration of the architecture of a computational cell. x_1, x_2 are input nodes, and x_3, x_4, x_5 are intermediate nodes. x_6 is the output node and concatenates all intermediate nodes. Each edge represents one typical operation between two nodes. Lines with different colors represent different operations. The thickness of a line represents the number of output channels of feature maps

3.1 Preliminaries: Differentiable Architecture Search

The framework of DARTS decomposes the searched neural network architecture into a number of (L) computational cells. In original DARTS, there are two types of computational cells: the normal cell and the reduction cell. The normal cell is used to transform feature maps. The reduction cell has another function to downsample the spatial size of the feature map. Each computational cell can be represented as a directed acyclic graph (DAG), consisting of an ordered sequence of N nodes and edges between each pair of nodes. We denote the set of nodes and the set of edges as:

$$\begin{aligned}\mathcal{N} &= \left\{x^{(i)} | i \in [1, N] \cap \mathbb{N}^+\right\} \\ \mathcal{E} &= \left\{o^{(i,j)} | i \in [1, N] \cap \mathbb{N}^+,\ j \in [1, N] \cap \mathbb{N}^+,\ i < j\right\},\end{aligned} \tag{1}$$

where $x^{(i)}$ denotes a node in a cell and $o^{(i,j)}$ denotes an edge from $x^{(i)}$ to $x^{(j)}$. In our setting, each node $x^{(i)}$ is a hidden representation (*i.e.,* a set of feature maps) and each edge defines how to transform feature maps from $x^{(i)}$ to $x^{(j)}$. Among a total of N nodes in a cell, there are two input nodes (*i.e.*, $x^{(1)}$ and $x^{(2)}$) and one output node $x^{(N)}$. The remaining nodes $x^{(i)}$ $(i \in \{3, ..., N-1\})$ are all called intermediate nodes. Two input nodes are used to transform outputs from previous two cells and prepare inputs for intermediate nodes (e.g., adjust the spatial size and channel dimension of feature maps). Then, prepared input feature maps go through edges in a cell to generate feature maps for all intermediate nodes. The feature map for each intermediate node $x^{(j)}$ is transformed from all previous nodes through their connected edges:

$$x^{(j)} = \sum_{i<j} o^{(i,j)}(x^{(i)}), \tag{2}$$

where the node $x^{(i)}$ is one predecessor of the intermediate node $x^{(j)}$. The output node $x^{(N)}$ is the concatenation of all intermediate nodes along the channel dimension and represents the output for a cell.

Actually, Neural Architecture Search (NAS) is an optimization problem. For each cell, we want to look for the optimal operation between each pair of nodes. Initially, we do not know what the optimal operation between each pair of nodes is. Therefore, as shown in Fig. 2a, we equip each edge with many candidate operations. There is a pre-defined space of operations denoted by $\mathcal{O}$, each element of which is a fixed operation (e.g., identity/skip connection, convolution and pooling with different kernels). Here, our goal is to automatically select one best operation from $\mathcal{O}$ for each edge $o^{(i,j)}$. To this end, some previous methods (Baker et al. 2017; Zoph and Le 2017) employ reinforcement learning to tackle the decision-making problem. However, this kind of method often consumes many computing resources and takes a long time to optimize neural architectures. Instead, the core idea of DARTS is to make the search space continuous and formulate the choice of an operation as a softmax over all candidate operations:

$$\bar{o}^{(i,j)}(x) = \sum_{o \in \mathcal{O}} \frac{\exp\left(\alpha_{i,j}^{o}\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\alpha_{i,j}^{o'}\right)} o(x), \tag{3}$$

where $\alpha_{i,j}^{o}$ 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. Since the softmax function is differentiable, DARTS opens the door to optimize neural architectures using the back-propagation algorithm. The neural architecture of a cell is denoted as:

$$\alpha = \{\alpha_{i,j} | i \in [1, N] \cap \mathbb{N}^+, j \in [1, N] \cap \mathbb{N}^+, i < j\}, \tag{4}$$

where α consists of $\alpha_{i,j}$ for all edges connecting pairs of nodes. The objective function for this optimization problem is to find α to minimize the loss function on the validation set:

$$\min_{\alpha} \; L_{val}(w^{*}(\alpha), \alpha) \tag{5}$$

$$\text{s.t. } w^{*}(a) = \text{argmin}_{w} \; L_{train}(w, \alpha), \tag{6}$$

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. 5 as:

$$\begin{aligned} &\nabla_{\alpha} L_{val}(w^{*}(\alpha), \alpha) \\ &\approx \nabla_{\alpha} L_{val}(w - \xi \nabla_{w} L_{train}(w, \alpha), \alpha), \end{aligned} \tag{7}$$

where w denotes the network weights, $\nabla_{w} L_{train}(w, \alpha)$ is a gradient step of w and ξ is the step's learning rate. When we finish the search stage, we only retain one operation for each edge. Therefore, as shown in Fig. 2c, we extract the operation with the strongest softmax activation from $\alpha_{i,j} \in \mathbb{R}^{|\mathcal{O}|}$ and assign it to the corresponding edge $o^{(i,j)}$. At the evaluation stage, $\alpha_{i,j}$ turns to a sparse one-hot vector, where only the best operation is retained. To avoid constructing neural networks with very complex topologies, we also constrain that each intermediate node only retains its two strongest predecessors following the original DARTS.

3.2 DARTS for Monocular 3D Pose Estimation

Since the original DARTS is designed for the task of 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 human 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 human body parts. To this end, as shown in Fig. 2b, 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. Another advantage of the fusion cell is that it could control the output channels of feature maps for each intermediate node. As shown in Fig. 2a, c, the thickness of the edge is the same in normal cells or reduction cells, indicating that intermediate nodes in these cells have the same output channels. To make our learning framework more flexible, as shown in Fig. 2b, d, we give fusion cells the ability to dynamically add 1×1 convolution between nodes to adjust output channels of feature maps according to our needs. In addition to effectively controlling the information flow in a cell and better fusing multi-scale features, this design also makes it possible to group body parts unevenly.

After upsampling feature maps at input nodes, fusion cells connect two nodes with different operations (*i.e.*, convolution, pooling, skip connection, etc.) to transform upsampled feature maps and produce volumetric heat maps for all parts at the output node. As shown in Fig. 2b, d, 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 human body parts. Through intermediate nodes in the fusion cell, we automatically divide all body parts into several groups. Benefiting from the design that different intermediate nodes in fusion cells can have different output channels, our method can unevenly divide body parts into several groups. In Fig. 2b, d, the number of groups is equal to the number of intermediate nodes in the fusion cell. The thickness of the line indicates how many parts are estimated by an intermediate node. As shown in Fig. 2b, there exist many candidate operations between nodes in the search stage, and we obtain the optimized architecture upon finishing the search stage. In the optimized architecture shown in Fig. 2d, we can observe that each intermediate node has been transformed by a different set of operations. In other words, our method can learn part-specific neural architectures in the search stage and employ these optimized architectures to estimate different groups of human body parts.

In the implementation, we follow a popular baseline Sun et al. (2018) to build our part-aware 3D human 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 (He et al. 2016) 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 computational 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:

$$P_r^i = \text{floor}\left(\frac{N_c - N_f}{N_r + 1}\right) \times i + 1, \tag{8}$$

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. To reduce the GPU memory, we resize images to 128×128 during the search stage. 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

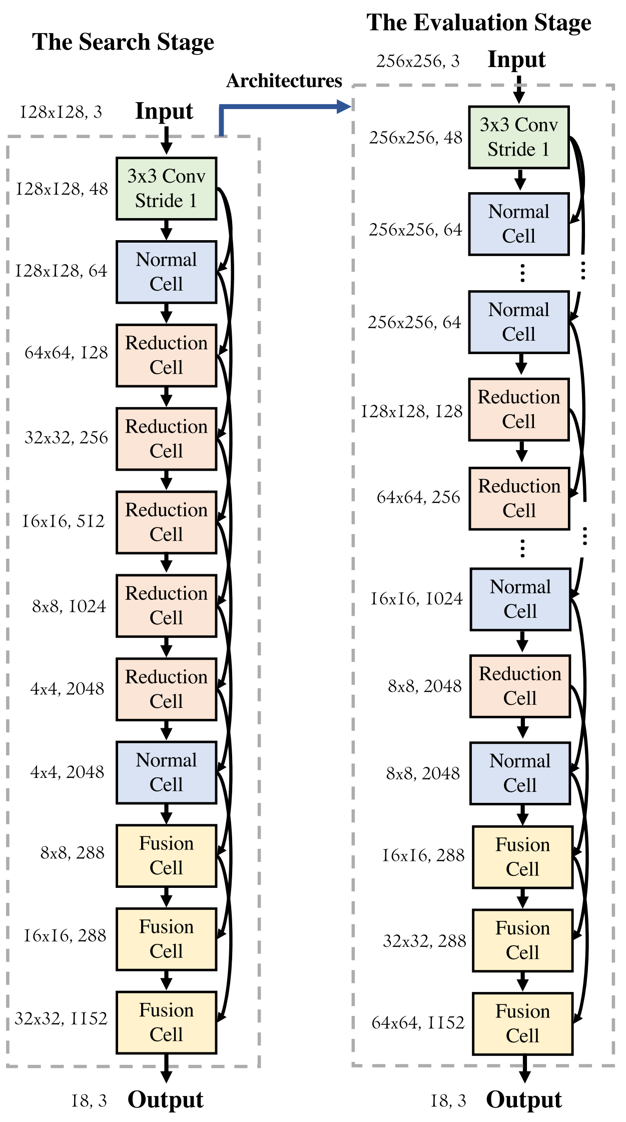


Fig. 3 An illustration of our proposed method. In the search stage, our model consists of ten computational cells: two normal cells, five reduction cells, and three fusion cells. The neural architecture of all types of cells is optimized in the search stage. In the evaluation stage, according to Eq. 8, we stack optimized cells to build our model. In our models, each cell receives inputs from the outputs of the previous two cells. In the search stage, we resize input images to 128 × 128 to save GPU memories

consecutively behind them to generate volumetric heat maps for all body parts. We employ L1 loss to supervise estimated 3D human 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 stage, we obtain the optimized normal cell, reduction cell, and fusion cell, as in Fig. 2d. According to Eq. 8, we stack these optimized cells to build our 3D human pose estimator. To evaluate the effectiveness of our searched neural network 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 the remaining fusion cells to 288. Three consecutive fusion cells upsample the feature map with a total stride of 2^3 and generate the volumetric heat map of size $64 \times 64 \times 1152$ for all human body parts. We extract the 3D coordinate from the corresponding volumetric heat map for each part via a differentiable soft-argmax operation:

$$J = \sum_{p_z=1}^{D} \sum_{p_y=1}^{H} \sum_{p_x=1}^{W} \boldsymbol{p} \cdot \frac{\exp(V(\boldsymbol{p}))}{\int_{\boldsymbol{q} \in \Omega} \exp(V(\boldsymbol{q}))}, \tag{9}$$

where $V(\boldsymbol{p})$ represents the estimated volumetric heat map, and Ω is its domain. We first normalize $V(\boldsymbol{p})$ via softmax, making all elements of $V(\boldsymbol{p})$ non-negative and sum to one. Then, the 3D joint coordinate J is the integration of all locations $\boldsymbol{p}$ in the domain, weighted by their probabilities. The spatial size of the volumetric heat map on depth, height, width is denoted as D, H, and W. In our case, they are all equal to 64. Finally, for all human body parts, we obtain our estimated result as a 18×3 vector. As we do in the search stage, we still employ the L1 loss to train our part-aware 3D human pose estimator.

3.3 Grouping of Related Body Parts

How to group related body parts also plays an important role in our method. Here, we propose two strategies to group related parts. In the first strategy, we follow the default Human3.6M part order[1], which can briefly reflect the connectivity of different parts. Following this order, we try to group body parts as evenly as possible according to a given number of groups N_g. The grouping results using this strategy are shown in Table 1. We use the first strategy as a baseline to test the performance of our part-aware 3D human pose estimator.

[1] The default order: pelvis, right hip, right knee, right ankle, left hip, left knee, left ankle, torso, neck, nose, head, left shoulder, left elbow, left wrist, right shoulder, right elbow, right wrist. We also manually add the thorax part to achieve an alignment with MPII dataset (Andriluka et al. 2014). When we perform model evaluation, we always exclude the thorax part.

Table 1 The results of using different strategies to group body parts. N_g is the number of groups we divide all body parts into

N_g	Results of grouping body parts using strategy I
Strategy I, $N_g = 1$	(pelv rhip rkne rank lhip lkne lank tors neck nose head lsho lelb lwri rsho relb rwri thor)
Strategy I, $N_g = 2$	(pelv rhip rkne rank lhip lkne lank tors neck) (nose head lsho lelb lwri rsho relb rwri thor)
Strategy I, $N_g = 3$	(pelv rhip rkne rank lhip lkne) (lank tors neck nose head lsho) (lelb lwri rsho relb rwri thor)
Strategy I, $N_g = 4$	(pelv rhip rkne rank) (lhip lkne lank tors) (neck nose head lsho lelb) (lwri rsho relb rwri thor)
Strategy I, $N_g = 6$	(pelv rhip rkne) (rank lhip lkne) (lank tors neck) (nose head lsho) (lelb lwri rsho) (relb rwri thor)
N_g	Results of grouping body parts using strategy II
Strategy II, $N_g = 1$	(pelv rhip rkne rank lhip lkne lank tors neck nose head lsho lelb lwri rsho relb rwri thor)
Strategy II, $N_g = 2$	(rhip lhip lkne tors neck nose lshoder lelb rshoder relb thor) (pelv rkne rank lank head lwri rwri)
Strategy II, $N_g = 3$	(lkne neck nose lsho lelb thor) (pelv rkne rank lank tors head lwri rwri) (rhip lhip rsho relb)
Strategy II, $N_g = 4$	(lkne nose lsho lelb thor) (rkne tors neck head lwri rsho relb) (rhip rank lhip) (pelv lank rwri)
Strategy II, $N_g = 6$	(nose lsho lelb thor) (rkne tors neck head relb) (rhip rank lhip lkne) (pelv lank) (lwri lsho) (rwri)

In the table, the prefix 'r' and 'l' stands for right and left. 'pelv', 'kne', 'ank', 'tors', 'sho', 'elb', 'wri', 'thor' are abbreviations for 'pelvis', 'knee', 'ankle', 'torso', 'shoulder', 'elbow', 'wrist', 'thorax', respectively

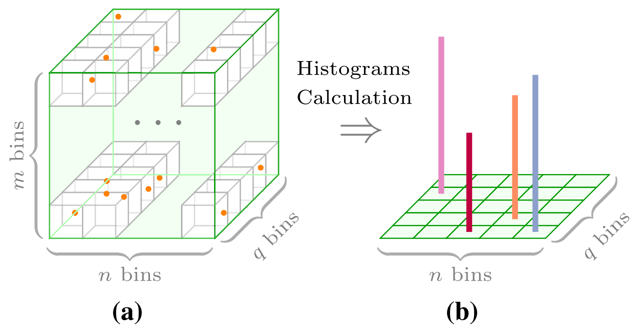


Fig. 4 An illustration of calculating the 3D histogram for a body part. As shown in (**a**), we first voxelize the 3D space into many bins, and the orange point represents 3D coordinates for a certain body part in Human3.6M dataset. Then, as shown in (**b**), we count how many 3D coordinates fall in each 3D cube and calculate the 3D histogram for a given body part. The more points a bin contains, the higher the pillar is shown in (**b**). For simplicity, we only visualize the calculation process at the bottom face of the 3D voxel space in (**b**)

The second grouping strategy is data-driven and treats each body part as a random variable in the 3D space. We attempt to calculate the mutual information between each pair of parts to measure their relatedness. By using camera parameters (*i.e.*, intrinsic matrix and extrinsic matrix), we first transform all 3D coordinates for a certain body joint to a global coordinate system, followed by another canonical transformation to ensure that the origin is located at the pelvis joint. As shown in Fig. 4a, we voxelize the 3D space, and every 3D coordinate falls in a 3D bin. Then, as shown in Fig. 4b, we approximate the distribution by calculating the 3D histogram for each body part. When we have the 3D histogram for all body parts, we calculate the mutual information (MacKay and Mac Kay 2003) to measure the relatedness between two body parts:

$$I\left(p_i, p_j\right) = \sum_{p_i \in P} \sum_{p_j \in P} H(p_i, p_j) \log\left(\frac{H\left(p_i, p_j\right)}{H\left(p_i\right) H\left(p_j\right)}\right), \tag{10}$$

where p_i indicates the i^{th} body part, and P contains all body parts. $H(\cdot)$ and $H(\cdot, \cdot)$ represent marginal and joint probability distributions. Here, we approximate distributions using 3D histograms, which count how many 3D points fall in each 3D bin. Once we finish the counting process, 3D histograms can represent the distribution for each kind of body part. Instead of using the Pearson correlation as the criterion, mutual information can measure not only the linear association but also the nonlinear association between two body parts, which is more suitable for our experimental setting.

In implementation, we calculate 3D histograms for different body parts on Human3.6M dataset (Ionescu et al. 2014) since the dataset contains a wide-range of 3D human poses and a large number of training samples. First, we define a 3D space, which can circumscribe the human body and is centered on the pelvis joint. Then, we voxelize this 3D space into 64 × 64 × 64 bins to ensure that each bin does not have too many or too few 3D points. In Fig. 5, we show the mutual information between each pair of body parts. Because the pelvis lies at the origin of the coordinate system, it has no associations with other parts. We can also observe that some parts, e.g., right hip and left hip have more associations than the others, e.g., right hip and left wrist. Based on the score matrix we obtain as shown in Fig. 5, we construct the affinity matrix via a Gaussian kernel to obtain well-behaved similarities for the spectral clustering algorithm:

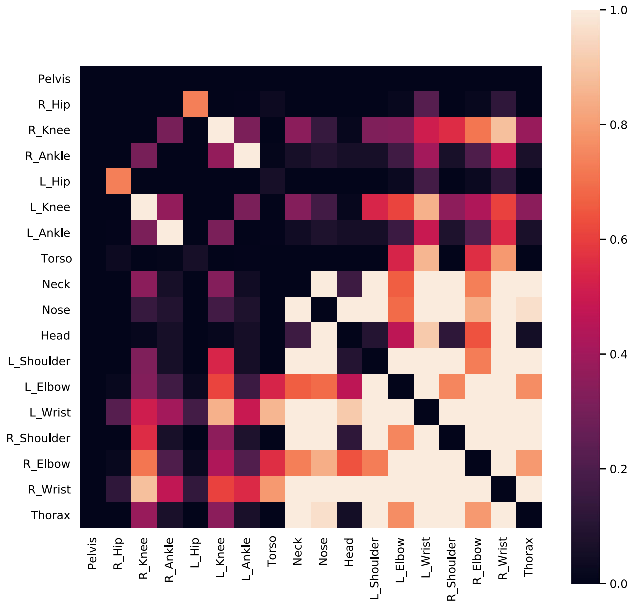


Fig. 5 We calculate the normalized mutual information between each pair of body parts. The brighter the color, the stronger the association between the two body parts

$$A\left(p_i, p_j\right) = \exp\left(-\gamma * \frac{I\left(p_i, p_j\right)^2}{2 * \delta^2}\right), \tag{11}$$

where we empirically set γ to 1 and δ to 0.5, which results in more balanced clustering results. Then, we employ the spectral clustering algorithm to divide all body parts into N_g groups. The spectral clustering solves the normalized cuts problem based on the given affinity matrix during the clustering process. Here, we employ the algorithm implemented by scikit-learn (Pedregosa et al. 2011) and obtain clustering results. As shown in Table 1, we can observe that part pairs with larger mutual information are often classified into the same group, which is in line with our intuition and facilitates us to search suitable neural architectures for different groups of body parts. In the next section, we will investigate which strategy and how many groups we divide can benefit our proposed method most.

4 Experimental Evaluation

In this section, we present a detailed evaluation of our proposed method. First, we introduce the main benchmarks and present our experimental settings. Then, we conduct a rigorous ablation analysis of our method. Finally, we build our strongest part-aware 3D human pose 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 (Ionescu et al. 2014): It is captured in a calibrated multi-view studio and consists of 3.6 million video frames. Eleven subjects are recorded from four camera viewpoints, performing 15 activities. Previous works widely use two evaluation metrics. The first one is the 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 (Mehta et al. 2018): These datasets are designed for multi-person 3D human pose estimation. The training set is the MuCo-3DHP dataset, and it is generated by compositing the MPI-INF-3DHP dataset (Mehta et al. 2017). 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).

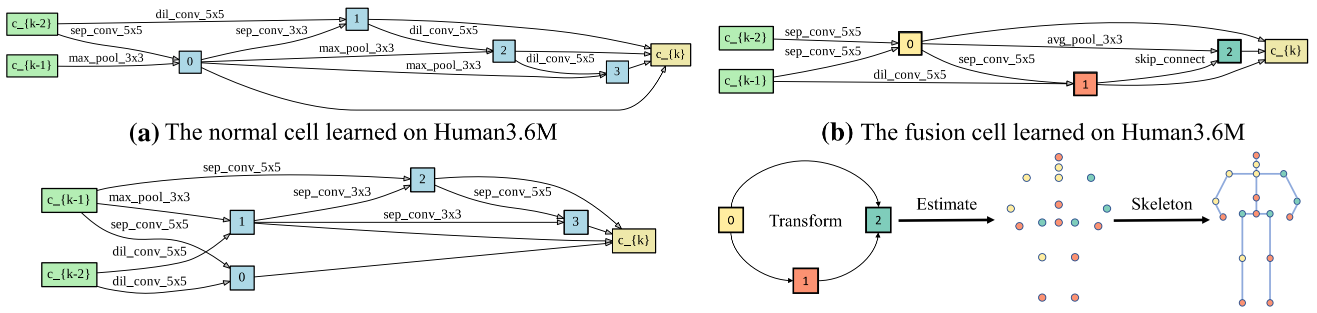
4.2 Experimental Settings and Implementation Details

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 (Andriluka et al. 2014) 2D human 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 (Lin et al. 2014) 2D pose data during training.

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 AdamW (Loshchilov and Hutter 2019) optimizer with a batch size of 40. The initial learning rate is 1×10^{-3} and



(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 different groups of parts with different nodes

Fig. 6 Cells optimized on Human3.6M dataset when we set N_g to three. Our model employs three intermediate nodes encoded in fusion cells to estimate different groups of human body parts

reduced by a factor of 10 at the 15th and the 20th 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 $64 \times 64 \times 64$, respectively. The total epoch is 20. We train our network with Adam (Kingma and Ba 2014) 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 ($\pm 30°$), horizontal flip, color jittering, and synthetic occlusion (Sárándi et al. 2018). To achieve an alignment between datasets, following previous works (Moon et al. 2019; Sun et al. 2018), we manually add the thorax joint and predict eighteen joints in the training process. At test time, we only evaluate the original seventeen joints.

4.3 Ablation Experiments

4.3.1 The Effect of Part Grouping Strategies

In this set of experiments, by using part grouping strategies summarized in Table 1, we are motivated to explore how to group different body parts could be an optimal choice for our method. In the search stage, we optimize neural architectures using different grouping strategies and a different number of groups. As we explained in Sect. 3, the number of groups we divide all body parts into is equal to the number of intermediate nodes in the fusion cell. In our setting, the fusion cell can have $N_g \in \{1, 2, 3, 4, 6\}$ intermediate nodes. As shown in Fig. 3, following original DARTS settings, our model has a total of ten computational cells. To be consistent with previous 3D human pose estimators (Sun et al. 2018), our model has three fusion cells at last to consecutively upsample feature maps at a stride of eight. Upon finishing the search stage, we train our model built with these optimized architectures. We summarize the performance of our models using different grouping strategies in Table 2.

According to Table 1, Strategy I follows the default Human3.6M part order and evenly divides parts into groups. Since it does not need additional 1×1 convolution layers to dynamically adjust channels of feature maps in fusion cells (widely used in Strategy II), searched models can have relatively small parameters and FLOPs. By using Strategy I, our model can achieve the best performance when N_g equals two. In this setting, fusion cells roughly divide body parts into two groups: low-degree-of-freedom parts (e.g., torso, hip) and high-degree-of-freedom parts (e.g., wrist, elbow). This grouping result could reduce the difficulty of searching for suitable neural architectures.

As shown in Table 1, Strategy II groups body parts in a data-driven fashion and can unevenly divide them into groups. As shown in Table 2, except for N_g equals two, models using Strategy II outperforms counterparts using Strategy I. When it divides all body parts into three groups, our model can outperform all other models on most actions and achieve an overall performance of 52.2 mm in MPJPE. In this setting, we visualize grouping results and searched neural architectures in Fig. 6. As shown in Fig. 6d, when N_g equals three, Strategy II divides body parts into three groups: yellow group, red group, cyan-blue group. Our method can search for suitable architectures for these three groups. As shown in Fig. 6b, fusion cells can employ three part-specific architectures to estimate different groups of body parts. Node 0 is transformed from depth-wise convolution layers and is used to estimate some left parts of the human body. The fusion cell further transforms Node 0 via depth-wise convolutions and the second input node with dilated convolutions and generates Node 1 to estimate some middle body parts. Node 2 has a connection with Node 0 via pooling layers and a skip connection with Node 1. It is used to estimate some right and middle parts of the human body. As shown in Fig 6a, the normal cell consists of many dilated convolutional layers, which significantly increase the receptive field of our model and are beneficial for performance improvement. In Fig 6c, the reduction cell employs many depth-wise separable convolution layers to fuse multi-scale features efficiently. When

Table 2 Quantitative evaluation of the number of intermediate nodes within each fusion cell on Human3.6M using Protocol 2

Methods	Params	FLOPs	Dire.	Dis.	Eat	Gre.	Pho.	Pose	Pur.	Sit	SD.	Smo.	Pho.	Wait	Walk	WD.	WT.	Ave.
S1, $N_g = 1$	15.1M	23.6G	52.6	59.9	51.2	53.4	58.3	51.6	53.0	68.2	68.4	55.9	61.3	54.1	42.6	57.1	48.1	56.4
S1, $N_g = 2$	11.7M	9.4G	49.5	$\underline{55.7}$	$\underline{47.7}$	50.5	$\underline{54.8}$	$\underline{47.4}$	51.6	65.1	**62.6**	53.8	$\underline{58.4}$	49.4	**40.3**	57.1	$\underline{46.2}$	$\underline{53.2}$
S1, $N_g = 3$	11.1M	9.1G	50.8	57.8	49.3	51.9	56.4	48.9	53.0	65.3	70.5	54.4	61.0	50.3	41.7	57.1	47.8	54.9
S1, $N_g = 4$	**8.0M**	$\underline{7.1G}$	47.8	56.8	50.7	51.3	58.9	48.9	53.8	66.7	70.5	56.6	61.1	50.0	43.4	59.2	48.5	55.6
S1, $N_g = 6$	$\underline{8.7M}$	**5.1G**	48.0	57.3	49.9	51.2	55.6	47.8	55.5	$\underline{64.5}$	67.1	54.0	60.5	49.5	42.4	58.3	47.4	54.4
S2, $N_g = 1$	15.1M	23.6G	52.6	59.9	51.2	53.4	58.3	51.6	53.0	68.2	68.4	55.9	61.3	54.1	42.6	57.1	48.1	56.4
S2, $N_g = 2$	13.3M	17.9G	52.1	59.1	49.6	53.3	56.2	51.2	53.1	65.7	65.3	55.7	61.2	51.1	43.4	59.2	48.7	55.5
S2, $N_g = 3$	12.2M	13.2G	**46.4**	**55.3**	**46.9**	**48.3**	**54.8**	**45.4**	$\underline{50.8}$	**64.2**	$\underline{62.8}$	**53.0**	**57.9**	**46.8**	$\underline{40.9}$	**55.5**	**44.9**	**52.2**
S2, $N_g = 4$	9.7M	9.0G	49.8	56.5	49.6	52.2	55.9	48.1	52.5	65.5	67.1	$\underline{53.8}$	58.6	48.6	41.4	56.5	46.9	54.1
S2, $N_g = 6$	8.8M	8.1G	$\underline{47.6}$	56.0	48.6	$\underline{50.2}$	55.9	48.7	**50.5**	65.6	70.0	54.1	59.9	$\underline{48.4}$	41.4	$\underline{56.2}$	46.4	53.9

N_g denotes the number of intermediate nodes within each fusion cell. S1 and S2 represents part grouping Strategy I and Strategy II, respectively. Lower is better, best in bold, second-best underlined

using Strategy II, the part grouping process and the architecture search process are both done automatically, which can fully exploit the strength of our method. Since our model can achieve the leading performance when we use Strategy II to divide body parts into three groups, we will keep this setting as the default in the following experiments.

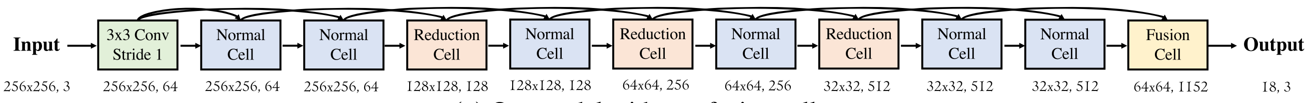
4.3.2 The Order of Different Groups of Parts

As shown in Fig. 6b, d, Node 0 only depends on input features to estimate left parts of the human body. However, Node 2 also unilaterally depends on Node 0 and Node 1 to make predictions. Due to the unilateral dependence between different nodes, different orders of groups can result in different neural architectures. Here, we attempt to investigate how the order of groups affect models' performance. To this end, we swap the order of different groups and summarize our results in Table 3. We can observe that our model can achieve the best performance when it follows the original order. On some hard actions (e.g., Sitting Down), it surpasses other models by more than 4 mm. To investigate the reason for this phenomenon, we take a closer look at the fusion cell searched shown in Fig. 6d. We can observe that different parts are roughly grouped according to their degrees of freedom. Node 1 estimates torso parts of the human body, which have the lowest degrees of freedom. Node 0 is employed to estimate some more flexible parts, including left elbow, left shoulder. Since most people are right-handed, right parts move more frequently than their left counterparts. These parts with the highest degrees of freedom are predicted by Node 2. Since the model searched with the original order can achieve the most competitive performance, we will keep this setting for following experiments.

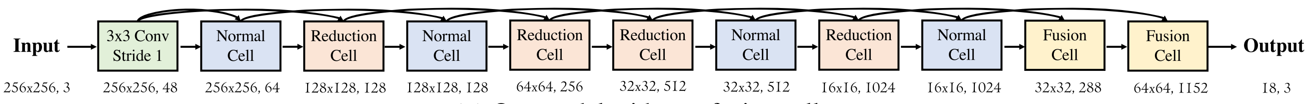
4.3.3 The Number of Fusion Cells

In the standard setting for 3D human pose estimators, we feed input images into our model and downsample them with a total stride of 2^5. Then, we use three fusion cells to upsample these feature maps consecutively with a total stride of 2^3. In this set of experiments, we want to make our framework more flexible and investigate how the number of fusion cells influences the performance of our models. To this end, we construct part-aware models built with $N_f \in \{1, 2, 3, 4\}$ fusion cells and illustrate their architectures in Fig. 7. In this set of experiments, based on results obtained from the experiments shown above, we still use Strategy II to group body parts and set N_g to three.

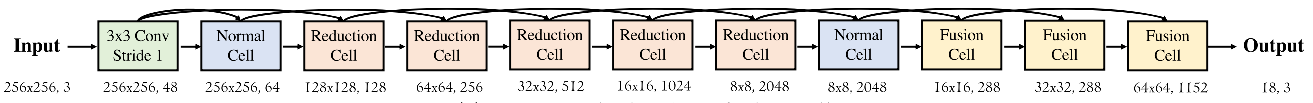
We summarize our experimental results with regard to the number of fusion cells in Table 4. We build our models with different architectures with one to four fusion cells. As shown in Fig. 7, all these models consist of ten computational cells. From Table 4, we can observe that most of



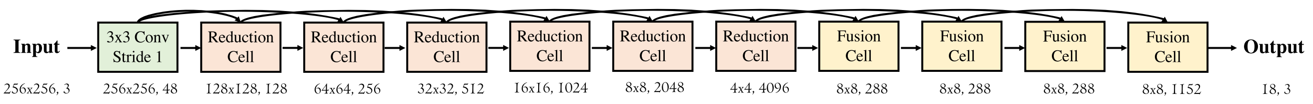
(a) Our model with one fusion cell



(b) Our model with two fusion cells



(c) Our model with three fusion cells



(d) Our model with four fusion cells

Fig. 7 Illustration of our models built with $N_f \in \{1, 2, 3, 4\}$ fusion cells. All our models consist of ten computational cells

Table 3 Quantitative evaluation of the group order on Human3.6M using Protocol 2

Methods	Results of Grouping Body Parts Using Strategy II
Original	(lkne neck nose lsho lelb thor) (pelv rkne rank lank tors head lwri rwri) (rhip lhip rsho relb)
Swapped #1	(rhip lhip rsho relb) (pelv rkne rank lank tors head lwri rwri) (lkne neck nose lsho lelb thor)
Swapped #2	(pelv rkne rank lank tors head lwri rwri) (lkne neck nose lsho lelb thor) (rhip lhip rsho relb)
Swapped #3	(lkne neck nose lsho lelb thor) (rhip lhip rsho relb) (pelv rkne rank lank tors head lwri rwri)

Methods	Dire.	Dis.	Eat	Gre.	Pho.	Pose	Pur.	Sit	SD.	Smo.	Pho.	Wait	Walk	WD.	WT.	Ave.
Original	46.4	55.3	**46.9**	**48.3**	54.8	**45.4**	**50.8**	64.2	**62.8**	53.0	**57.9**	**46.8**	**40.9**	**55.5**	**44.9**	**52.2**
Swapped #1	47.4	55.7	47.4	50.3	**54.4**	47.1	51.1	62.8	68.4	53.4	58.7	48.3	41.0	57.0	47.0	53.2
Swapped #2	**45.6**	**55.1**	47.4	49.6	54.6	47.0	52.9	**62.5**	70.1	**52.6**	58.6	47.7	42.2	58.8	46.8	53.1
Swapped #3	48.3	56.1	48.0	50.5	55.9	47.8	51.7	63.5	66.9	53.2	59.3	49.4	41.9	57.6	47.4	53.7

We set N_c and N_g to ten and three, respectively. We compute part-wise MPJPE to report performance. Lower is better, best in bold, second-best underlined

the network parameters are consumed in backbone architectures (*i.e.*, normal cells and reduction cells) for our models with two to four fusion cells. However, in all of our models, most of the computation is done in fusion cells, which also highlights the importance of our proposed fusion cells. As shown in Table 4, when our model has three fusion cells, it can achieve the best performance, especially on some hard actions (e.g., Sitting Down). In human pose estimation, high-resolution representation (Sun et al. 2019) is very important to obtain good performance. Our model with four fusion cells employs too many reduction cells to downsample feature maps and loses too many discriminative features during this process, contributing to its poor overall performance. From our experimental results, we think that three fusion cells can efficiently fuse multi-scale features and be a good choice for our model. Therefore, in our following experiments, we choose to set N_f to three as the default setting for our method.

4.3.4 The Importance of Search Space

As we know, the search space plays a crucial role in NAS. To further validate the effectiveness of our method, we add two more operations in our search space $\mathcal{O}$: 7×7 depth-wise convolutions, 7×1 and 1×7 convolutions. In this set of experiments, we use Strategy I and Strategy II to group body parts and set N_f to three. As shown in Table 5, in most cases, our models using Strategy II achieve better performance than their counterparts using Strategy I. When we use Strategy I to group body parts, our model can achieve the best performance when N_g equals one. However, when

Table 4 Quantitative evaluation of the number of fusion cells in our model on Human3.6M using Protocol 2

Methods	Params	FLOPs	FP	FF	Dire.	Dis.	Eat	Gre.	Pho.	Pose	Pur.	Sit	SD.	Smo.	Pho.	Wait	Walk	WD.	WT.	Ave.
$N_f = 1$	**3.8M**	15.1G	**74.7%**	76.8%	**45.2**	54.5	49.5	**47.9**	54.9	**43.7**	52.2	65.0	75.0	53.4	57.6	48.9	**40.7**	56.6	45.0	53.2
$N_f = 2$	4.8M	**11.5G**	45.8%	70.8%	48.0	**53.7**	50.8	49.9	**53.7**	46.3	**47.5**	**63.7**	70.8	**52.4**	**57.3**	48.4	41.4	55.7	46.0	52.9
$N_f = 3$	12.2M	13.2G	26.3%	76.2%	46.4	55.3	**46.9**	48.3	54.8	45.4	50.8	64.2	**62.8**	53.0	57.9	46.8	40.9	**55.5**	**44.9**	**52.2**
$N_f = 4$	16.3M	11.2G	22.0%	**84.1%**	53.3	61.4	52.4	54.8	59.9	54.9	57.7	66.4	75.2	57.8	65.7	55.5	46.5	61.9	51.9	58.8

All our models consist of ten computational cells. N_f denotes the number of fusion cells in our model. FP and FF indicates the percentage of parameters and FLOPs in fusion cells. In each fusion cell, there are three intermediate nodes. Lower is better, best in bold, second-best underlined

we use Strategy II to divide body parts into three parts, our model can achieve the most competitive performance. This phenomenon suggests that different grouping strategies can yield different results. In our experimental settings, Strategy II can group body parts in a data-driven fashion and can help search for more efficient neural architectures By using Strategy II, as we increase the number of operations in $\mathcal{O}$, we can observe a consistent improvement in performance. When we add 7×7 depth-wise convolutions into our search space, our model can become more lightweight and achieve better performance. When we add another 7×1 and 1×7 convolutions, our model becomes more competitive but also has more parameters and FLOPs. This set of experiments validate that our work method is flexible and can become more lightweight if we put corresponding operations (e.g., convolutions with small kernels) into its search space.

4.3.5 The Number of Computational Cells

Instead of only stacking ten computation cells, we attempt to construct a deeper part-aware 3D human pose estimator, according to Eq. 8. Following previous experiments, we use Strategy II to group body parts and set N_g and N_f to three. As a common practice in many human pose estimation methods (Newell et al. 2016; Wu et al. 2018; Sun et al. 2019), we employ intermediate supervision on multi-scale feature maps to train our large models with fifteen or twenty computational cells. We still search our models in the original space of eight operations to achieve a balance between performance and computational complexity. As shown in Table 6, as we increase the number of computational cells, our model becomes better in performance but has more parameters and FLOPs. When we set N_c to twenty, our model achieves the best performance, 46.8 mm in MPJPE. As we increase N_c from ten to twenty, the gain in network parameters (from 12.2M to 19.2M) and FLOPs (from 12.9G to 16.0G) does not compromise the gain in performance (from 52.2 to 46.8 mm). This phenomenon also demonstrates that the network architecture optimized during the search process is very computationally efficient.

4.3.6 The Part-Awareness of Our Model

We begin to validate the part-awareness of our method from two aspects. First, to investigate whether searched decoder networks are part-specific, we intend to shuffle the order of parts when we re-train our model in the evaluation stage. Suppose that our model trained with the shuffled order behaves worse than the original one. In that case, we can validate that our searched decoder networks are optimized for certain groups of body parts. To this end, we randomly shuffle the part order three times and train networks with these shuffled orders. In this set of experiments, we run experiments

Table 5 Quantitative evaluation of the importance of the search space using Protocol 2

Methods	Search Space		Params	FLOPs	Direct.	Discuss	Eating	Greet	Phone	Pose
	7×7 sep.	7×7 conv.								
$S1, N_g = 1, N_f = 3$	✓	✓	19.4M	19.5G	45.7	54.2	**45.3**	48.3	55.8	45.9
$S1, N_g = 2, N_f = 3$	✓	✓	12.4M	12.2G	51.3	59.8	50.6	52.2	57.3	49.5
$S1, N_g = 3, N_f = 3$	✓	✓	25.2M	13.3G	50.5	58.1	48.6	52.1	57.7	50.9
$S1, N_g = 4, N_f = 3$	✓	✓	20.0M	17.5G	50.7	57.2	48.9	52.6	57.5	50.0
$S1, N_g = 6, N_f = 3$	✓	✓	12.2M	**8.9G**	48.7	55.2	47.9	50.0	54.6	48.3
$S2, N_g = 1, N_f = 3$	✓	✓	24.5M	26.4G	49.1	56.9	47.7	50.8	55.7	48.6
$S2, N_g = 2, N_f = 3$	✓	✓	19.2M	44.0G	50.3	56.9	49.4	52.2	55.8	50.2
$S2, N_g = 3, N_f = 3$	✓	✓	13.8M	23.9G	**45.1**	54.6	47.7	**47.1**	**53.0**	**45.2**
$S2, N_g = 4, N_f = 3$	✓	✓	**10.5M**	15.9G	47.0	55.1	49.5	49.7	57.9	46.5
$S2, N_g = 6, N_f = 3$	✓	✓	18.8M	*10.3G*	46.6	**53.7**	47.7	48.9	55.8	46.2
$S2, N_g = 3, N_f = 3$	–	–	12.2M	13.2G	46.4	55.3	46.9	48.3	54.8	45.4
$S2, N_g = 3, N_f = 3$	✓	–	*11.9M*	12.0G	46.6	56.5	46.3	48.3	54.5	46.3

Methods	Purch.	Sitting	SitD.	Smoke	Photo	Wait	Walk	WalkD.	WalkT.	Ave.
$S1, N_g = 1, N_f = 3$	48.9	62.4	67.5	52.3	57.0	46.5	39.4	54.7	45.0	51.9
$S1, N_g = 2, N_f = 3$	51.1	66.7	73.3	54.7	62.8	50.3	43.0	58.7	48.6	56.0
$S1, N_g = 3, N_f = 3$	51.7	68.8	75.9	55.1	60.6	51.8	42.4	57.7	47.5	55.9
$S1, N_g = 4, N_f = 3$	51.6	67.4	74.7	54.9	59.6	50.6	41.4	57.4	47.0	55.4
$S1, N_g = 6, N_f = 3$	51.7	65.8	73.5	53.3	57.0	48.6	41.5	56.6	45.8	53.7
$S2, N_g = 1, N_f = 3$	52.1	63.2	69.2	54.1	59.0	48.5	41.2	57.4	46.9	54.9
$S2, N_g = 2, N_f = 3$	52.2	64.6	66.8	55.5	59.8	49.9	44.0	59.0	48.7	54.8
$S2, N_g = 3, N_f = 3$	51.1	62.4	**65.0**	**51.0**	**55.7**	46.8	**38.9**	**53.9**	**43.6**	**51.4**
$S2, N_g = 4, N_f = 3$	52.4	65.1	66.8	54.7	59.8	49.2	41.0	58.6	46.2	53.9
$S2, N_g = 6, N_f = 3$	**48.8**	64.0	70.7	53.0	56.7	47.3	40.1	54.2	45.3	52.6
$S2, N_g = 3, N_f = 3$	50.8	64.2	62.8	53.0	57.9	46.8	40.9	55.5	44.9	52.2
$S2, N_g = 3, N_f = 3$	52.1	**62.3**	65.0	51.5	58.0	**46.3**	39.1	55.4	43.9	52.0

S1 and S2 denote that we use Strategy I or II to group body parts. N_g denotes the number of groups we divide body parts into. N_f denotes the number of fusion cells. Here, 7×7 sep. and 7×7 conv. denote depth-wise separable convolution layers and 7×1, 1×7 convolution layers, respectively. Lower is better, best in bold, second-best underlined

Table 6 Quantitative evaluation of the number of computational cells on Human3.6M using Protocol 2

Methods	Params	FLOPs	Dire.	Dis.	Eat	Gre.	Pho.	Pose	Pur.	Sit	SD.	Smo.	Pho.	Wait	Walk	WD.	WT.	Ave.
$N_c = 10$	**12.2M**	**12.9G**	46.4	55.3	46.9	48.3	54.8	45.4	50.8	64.2	**62.8**	53.0	57.9	46.8	40.9	55.5	44.9	52.2
$N_c = 15$	*13.8M*	*14.6G*	44.2	52.7	45.3	46.2	51.5	44.0	47.6	61.7	65.6	50.5	55.4	45.9	39.7	54.9	44.2	50.5
$N_c = 20$	19.2M	16.0G	**42.5**	**49.4**	**43.0**	**44.6**	**50.2**	**40.9**	**46.8**	**58.5**	63.4	**48.3**	**51.9**	**42.0**	**36.1**	**49.8**	**40.5**	**46.8**

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

Table 7 Quantitative evaluation of the shuffled part order on Human3.6M using Protocol 2

Methods	pelv	rhip	rkne	rank	lhip	lkne	lank	tors	neck	nose	head	lsho	lelb	lwri	rsho	relb	rwri	ave.
Original	0.0	**22.4**	**51.7**	77.2	**22.6**	**46.0**	77.2	**37.2**	**42.6**	**49.7**	**52.6**	**51.0**	66.1	**84.5**	**50.8**	70.1	87.3	**52.2**
Shuffled #1	0.0	24.9	53.5	73.2	25.5	47.3	**74.0**	41.3	48.4	52.7	56.2	54.4	67.4	84.7	54.7	72.0	88.5	54.1
Shuffled #2	0.0	24.4	51.7	76.5	24.4	48.6	76.4	41.3	47.3	50.9	54.7	52.3	**65.7**	84.5	51.5	**69.1**	**86.8**	53.3
Shuffled #3	0.0	26.3	54.8	**72.5**	26.4	50.0	74.2	44.4	48.7	51.9	53.4	55.0	66.7	83.6	54.8	72.0	88.8	54.4

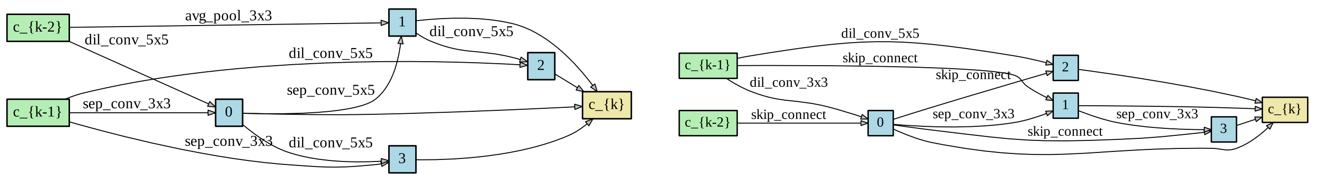
We set N_c and N_g to ten and three, respectively. We compute part-wise MPJPE to report performance. Lower is better, best in bold, second-best underlined

Table 8 Quantitative evaluation of the importance of the fusion cell on Human3.6M using Protocol 2

Methods	Backbone	Pretrain	Params	FLOPs	Direct.	Discuss	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	–	20.5*M*	**12.5G**	49.0	59.9	49.8	53.5	58.0	51.0
Ours, WS	–	–	**12.2M**	12.9*G*	**46.4**	55.3	**46.9**	**48.3**	54.8	**45.4**

Methods	Purch.	Sitting	SitD.	Smoke	Photo	Wait	Walk	WalkD.	WalkT.	Ave.
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	**50.8**	64.2	**62.8**	53.0	**57.9**	**46.8**	**40.9**	55.5	**44.9**	**52.2**

BS and WS denote the backbone search and the whole architecture search, respectively. To search the whole architecture, we use Strategy II to group body parts and divide them into three groups. We compute action-wise MPJPE to report the network performance. Lower is better, best in bold, second-best underlined



(a) The normal cell learned in backbone search **(b)** The reduction cell learned in backbone search

Fig. 8 Cells optimized on Human3.6M dataset when we only search the backbone network

Table 9 Comparison with state-of-the-art methods on Human3.6M using Protocol 1

Methods	Dire.	Dis.	Eat	Gre.	Phe.	Pose	Pur.	Sit	SitD.	Smo.	Phot.	Wait	Walk	WD.	WT.	Avg
Yasin et al. (2016)	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 and Ramanan (2017)	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-Noguer (2017)	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 et al. (2018)	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 et al. (2017)	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 et al. (2018)	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 et al. (2018)	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 et al. (2019)	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, small model	29.4	31.3	39.6	34.9	34.5	31.5	**30.0**	35.5	46.8	35.7	39.4	29.3	25.0	35.8	28.6	34.2
Ours, large model	**26.1**	**30.3**	**36.3**	**32.2**	**32.5**	**27.9**	34.1	**33.1**	44.4	**33.0**	38.4	**27.5**	**21.5**	37.0	**26.3**	**32.2**

Our small model has ten computational cells, and our large model is built with twenty cells. Lower is better, best in bold, second-best underlined

three times and train our model with different shuffled orders. As shown in Table 7, we observe that all models trained with shuffled orders suffer from a noticeable drop in performance, more than 1 mm in MPJPE. As we take a closer look, the decline in performance also reflects on every individual part, especially parts with lower DOFs (e.g., torso, neck), and their estimation accuracy might drop by more than 4 mm. By comparing models trained with shuffled orders, we validate that our approach learns part-specific decoder networks (e.g., topologies, kernel sizes, receptive fields) for specific body parts in the search stage.

Within our model, the fusion cells play a pivotal role in learning part-specific decoder network architectures, and most of the computation is done in them according to Table 4. 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 only difference is whether they have fusion cells. As shown in Fig. 8, we visualize optimized neural architectures when we only search the backbone network. In Table 8, compared to the backbone search, searching the whole network

Table 10 Comparison with state-of-the-art methods on Human3.6M using Protocol 2

Methods	Dire.	Dis.	Eat	Gre.	Phe.	Pose	Pur.	Sit	SitD.	Smo.	Phot.	Wait	Walk	WD.	WT.	Avg
Chen and Ramanan (2017)	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 et al. (2017)	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-Noguer (2017)	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 et al. (2018)	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 et al. (2017)	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 et al. (2018)	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 et al. (2018)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	59.9
Sun et al. (2017)	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
Sun et al. (2017)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	56.8
Moon et al. (2019)	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 et al. (2018)	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, small model	46.4	55.3	46.9	48.3	54.8	45.4	50.8	64.2	**62.8**	53.0	57.9	46.8	40.9	55.5	44.9	52.2
Ours, large model	**42.5**	49.4	**43.0**	**44.6**	**50.2**	**40.9**	46.8	**58.5**	63.4	**48.3**	**51.9**	**42.0**	**36.1**	49.8	**40.5**	**46.8**

Our small model has ten computational cells, and our large model is built with twenty cells. Lower is better, best in bold, second-best underlined

architecture improves performance by 4.9 mm and reduces 40.5% parameters. Though we use more complex operations to upsample feature maps in fusion cells, it only results in a negligible increase in FLOPs when compared with backbone search. In comparison with the model built on the commonly used ResNet-50 backbone, we advance estimation accuracy by 1.7 mm with 64.4% fewer parameters and 8.5% fewer FLOPs. It is also worth noting that our models do not require any pretraining on the ImageNet dataset (Deng et al. 2009). The reason why backbone search does not outperform ResNet50 baseline might lie in the gap between the backbone architecture and the head network architecture. The backbone part of the model consists of very complex topologies, which is not fully compatible with the manually designed head network. Besides, from the search stage to the evaluation stage, the backbone network changes from Fig. 2a–c, while the head network remains the same in both stages. Therefore, in such a case, it is more difficult for DARTS to search suitable neural architectures for the evaluation stage. To improve the performance of the backbone search, perhaps we need better training strategies and better architecture search algorithms. Through this set of experiments, we show that fusion cells significantly contribute to the efficiency of our method and exhibit an advantage over models using the ResNet-50 backbone.

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 9, 10 and 11, respectively. In Fig. 9, we show qualitative results produced by our small model with ten cells. It can generalize well for in-the-wild images, even on challenging poses and crowded scenes. All our models have three fusion cells and three intermediate nodes in each fusion cell. We run our models three times, and the variances of our small and large models are about ±0.8 mm and ±1.2 mm, respectively.

Single-person 3D human pose estimation: We compare our approach on Human3.6M with state-of-the-art methods in Tables 9, 10. By reducing about 40% parameters, our large part-aware model advances the-state-of-the-art accuracy by 1.8 mm and 2.8 mm in protocol 1 and protocol 2, respectively.

Multi-person 3D human pose estimation: We also extend our work to perform multi-person 3D human pose estimation. We follow the top-down pipeline for multi-person pose estimation. First, we detect each human with bounding boxes (He et al. 2017) and crop images with these bounding boxes. Then, we employ RootNet (Moon et al. 2019) to estimate absolute depth for the pelvis part of the person in each bounding box and use our model to perform single-person 3D

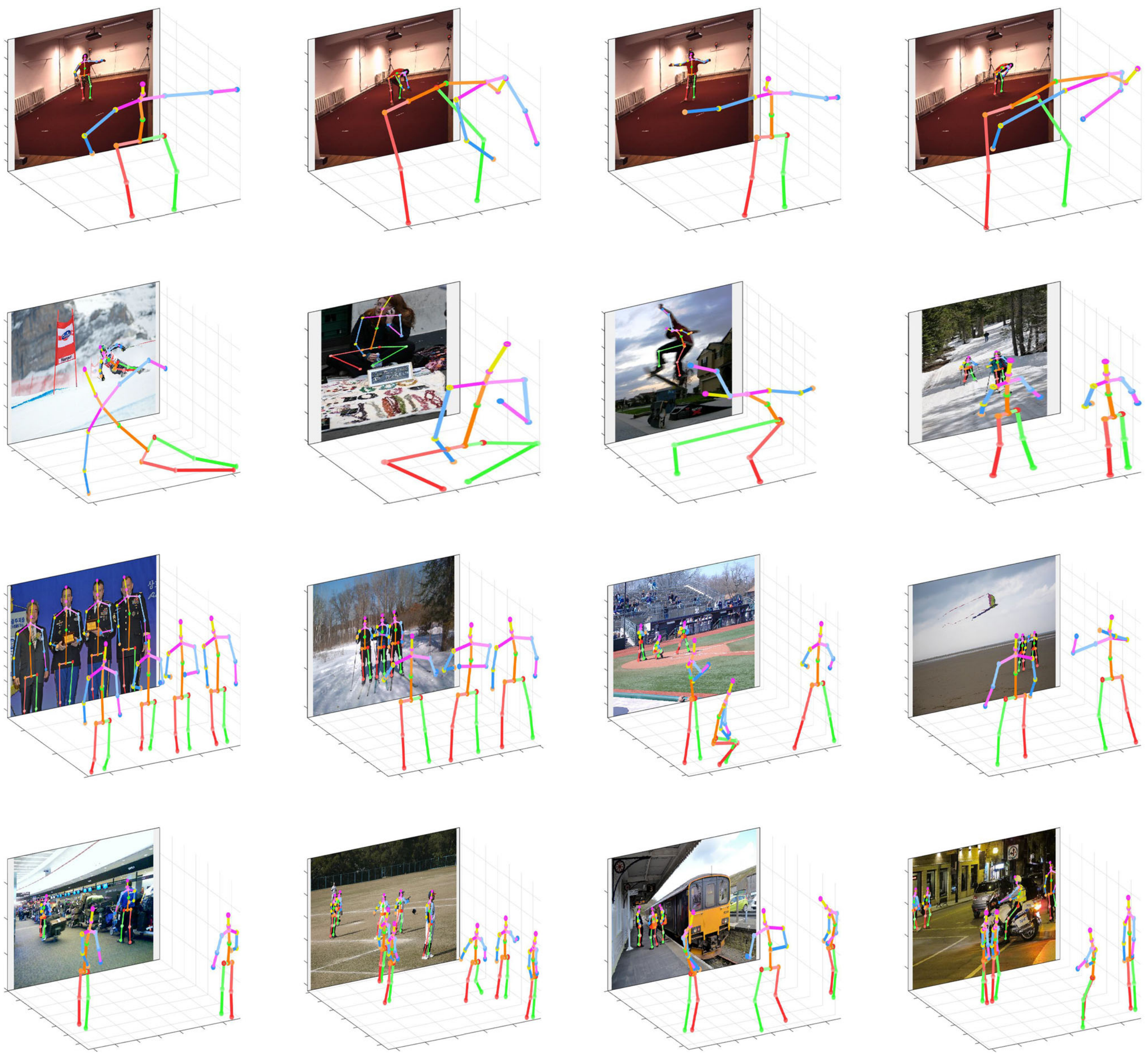


Fig. 9 Qualitative results on different datasets. Our model can produce convincing results on some challenging cases

pose estimation from cropped images. As shown in Table 11, 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 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 neural 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. Through extensive ablation experiments, we validate the effectiveness and robustness of our proposed method. As a result, our model advances the state-of-the-art accuracy on both the single-person and multi-person 3D human pose estimation benchmarks. Though our method shows promising performance, it also has two major limitations: (1) the organization of computational cells is manually defined, and (2) the number of groups is determined via the grid search. In future works towards a more global optimization for our part-aware

Table 11 Comparison with state-of-the-art methods on MuPoTS-3D using all ground truths

Methods	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Rogez et al. (2017)	67.7	49.8	53.4	59.1	67.5	22.8	43.7	49.9	31.1	78.1
Mehta et al. (2018)	81.0	60.9	64.4	63.0	69.1	30.3	65.0	59.6	64.1	83.9
Moon et al. (2019)	<u>94.4</u>	77.5	79.0	<u>81.9</u>	<u>85.3</u>	<u>72.8</u>	<u>81.9</u>	<u>75.7</u>	<u>90.2</u>	<u>90.4</u>
Ours, small model	94.2	<u>76.5</u>	<u>80.3</u>	77.6	84.0	64.8	78.8	73.6	87.5	88.4
Ours, large model	**95.6**	**80.7**	**81.0**	**84.8**	**86.8**	**75.1**	**82.1**	**79.2**	**91.4**	**93.5**
Methods	**S11**	**S12**	**S13**	**S14**	**S15**	**S16**	**S17**	**S18**	**S19**	**S20**
Rogez et al. (2017)	50.2	51.0	51.6	49.3	56.2	66.5	65.2	62.9	66.1	59.1
Mehta et al. (2018)	68.0	68.6	62.3	59.2	70.1	80.0	79.6	67.3	66.6	67.2
Moon et al. (2019)	<u>79.2</u>	<u>79.9</u>	<u>75.1</u>	<u>72.7</u>	<u>81.1</u>	<u>89.9</u>	<u>89.6</u>	<u>81.8</u>	<u>81.7</u>	<u>76.2</u>
Ours, small model	75.7	79.7	72.1	71.4	77.9	86.6	87.2	80.5	79.8	72.0
Ours, large model	**83.7**	**82.2**	**78.1**	**76.3**	**84.3**	**92.5**	**91.6**	**85.4**	**82.1**	**77.8**

Our small model has ten computational cells, and our large model is built with twenty cells. Higher is better, best in bold, second-best underlined

3D human pose estimators, we attempt to explore other NAS methods (Guo et al. 2020; Cai et al. 2019a) to obtain more efficient models.

Acknowledgements This work was jointly supported by National Key Research and Development Program of China Grant No. 2018AAA0100400, National Natural Science Foundation of China (61633021, 61721004, 61806194, U1803261, and 61976132), Beijing Nova Program (Z201100006820079), Shandong Provincial Key Research and Development Program (2019JZZY010119), Key Research Program of Frontier Sciences CAS Grant No.ZDBS-LY-JSC032, and CAS-AIR.

References

Alldieck, T,. Pons-Moll, G., Theobalt, C., & Magnor, M. (2019). Tex2shape: Detailed full human body geometry from a single image. In *IEEE international conference on computer vision (ICCV)*.

Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Baek, S., Kim, K. I., & Kim, T. K. (2018). Augmented skeleton space transfer for depth-based hand pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Baker, B., Gupta, O., Naik, N., & Raskar, R. (2017). Designing neural network architectures using reinforcement learning. In *International conference on learning representations (ICLR)*.

Belagiannis, V., Amin, S., Andriluka, M., Schiele, B., Navab, N., & Ilic, S. (2014). 3d pictorial structures for multiple human pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., & Black, M. J. (2016). Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *European conference on computer vision (ECCV)*.

Burenius, M., Sullivan, J., & Carlsson, S. (2013). 3d pictorial structures for multiple view articulated pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Cai, Y., Ge, L., Liu, J., Cai, J., Cham, T. J., Yuan, J., & Thalmann, N. M. (2019b). Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In *IEEE international conference on computer vision (ICCV)*.

Cai, H., Zhu, L., & Han, S. (2019a). Proxylessnas: Direct neural architecture search on target task and hardware. In *International conference on learning representations (ICLR)*.

Chen, C. H., & Ramanan, D. (2017). 3d human pose estimation = 2d pose estimation + matching. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Chen, L. C., Collins, M., Zhu, Y., Papandreou, G., Zoph, B., Schroff, F., Adam, H., & Shlens, J. (2018). Searching for efficient multi-scale architectures for dense image prediction. In *Conference on neural information processing systems (NeurIPS)*.

Chen, Z., Guo, Y., Huang, Y., & Liang, W. (2019b). Learning depth-aware heatmaps for 3d human pose estimation in the wild. In *British machine vision conference (BMVC)*.

Chen, Z., Huang, Y., Yu, H., Xue, B., Han, K., Guo, Y., & Wang, L. (2020). Towards part-aware monocular 3d human pose estimation: An architecture search approach. In *European conference on computer vision (ECCV)*.

Chen, Y., Yang, T., Zhang, X., Meng, G., Xiao, X., & Sun, J. (2019a). Detnas: Backbone search for object detection. In *Conference on neural information processing systems (NeurIPS)*.

Ci, H., Wang, C., Ma, X., & Wang, Y. (2019). Optimizing network structure for 3d human pose estimation. In *IEEE international conference on computer vision (ICCV)*.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Divvala, S. K., Efros, A. A., & Hebert, M. (2012). How important are "deformable parts" in the deformable parts model? In *European conference on computer vision (ECCV)*.

Fabbri, M., Lanzi, F., Calderara, S., Alletto, S., & Cucchiara, R. (2020). Compressed volumetric heatmaps for multi-person 3d pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Fang, H., Xu, Y., Wang, W., Liu, X., & Zhu, S. C. (2018). Learning knowledge-guided pose grammar machine for 3d human pose estimation. In *AAAI conference on artificial intelligence (AAAI)*.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2009). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 32,* 1627–1645.

Ganapathi, V., Plagemann, C., Koller, D., & Thrun, S. (2010). Real time motion capture using a single time-of-flight camera. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Ghiasi, G., Lin, T. Y., & Le, Q. V. (2019). Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Guo, Z., Zhang, X., Mu, H., Heng, W., Liu, Z., Wei, Y., & Sun, J. (2020). Single path one-shot neural architecture search with uniform sampling. In *European conference on computer vision (ECCV).*

Gupta, A., Martinez, J., Little. J. J., & Woodham, R. J. (2014). 3d pose from motion for cross-view action recognition via non-linear circulant temporal encoding. In *IEEE conference on computer vision and pattern recognition (CVPR).*

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In *IEEE international conference on computer vision (ICCV).*

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Hossain, M. R. I., & Little, J. J. (2018). Exploiting temporal information for 3d human pose estimation. In *European conference on computer vision (ECCV).*

Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V. (2019). Searching for mobilenetv3. In *IEEE international conference on computer vision (ICCV).*

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2014). Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. In *IEEE transactions on pattern analysis and machine intelligence (TPAMI).*

Iskakov, K., Burkov, E., Lempitsky, V., & Malkov, Y. (2019). Learnable triangulation of human pose. In *IEEE international conference on computer vision (ICCV).*

Jiang, H. (2010). 3d human pose reconstruction using millions of exemplars. In *International conference on pattern recognition (ICPR)*

Jiang, W., Kolotouros, N., Pavlakos, G., Zhou, X., & Daniilidis, K. (2020). Coherent reconstruction of multiple humans from a single image. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Joo, H., Liu, H., Tan, L., Gui, L., Nabbe, B., Matthews, I., Kanade, T., Nobuhara, S., & Sheikh, Y. (2015). Panoptic studio: A massively multiview system for social motion capture. In *IEEE international conference on computer vision (ICCV).*

Kanazawa, A., Black, M. J., Jacobs, D. W., & Malik, J. (2018). End-to-end recovery of human shape and pose. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. In *International conference on learning representations (ICLR).*

Kocabas, M., Karagoz, S., & Akbas, E. (2019). Self-supervised learning of 3d human pose using multi-view geometry. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Kolotouros, N., Pavlakos, G., & Daniilidis, K. (2019). Convolutional mesh regression for single-image human shape reconstruction. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Lee, H. J., & Chen, Z. (1985). Determination of 3d human body postures from a single view. In *Computer vision, graphics, and image processing (CVGIP).*

Li, S., & Chan, A. B. (2014). 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian conference on computer vision (ACCV).*

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV).*

Liu, C., Chen, L. C., Schroff, F., Adam, H., Hua, W., Yuille, A. L., & Fei-Fei, L. (2019a). Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Liu, H., Simonyan, K., & Yang, Y. (2019b). Darts: Differentiable architecture search. In *International conference on learning representations (ICLR).*

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International conference on learning representations (ICLR).*

MacKay, D. J., & Mac Kay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge University Press.

Martinez, J., Hossain, R., Romero, J., & Little, J. J. (2017). A simple yet effective baseline for 3d human pose estimation. In *IEEE international conference on computer vision (ICCV).*

Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., & Theobalt, C. (2017). Monocular 3d human pose estimation in the wild using improved cnn supervision. In *International conference on 3D vision (3DV).*

Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Sridhar, S., Pons-Moll, G., & Theobalt, C. (2018). Single-shot multi-person 3d pose estimation from monocular rgb. In *International conference on 3D vision (3DV).*

Moon, G., Chang, J. Y., & Lee, K. M. (2019). Camera distance-aware top-down approach for 3d multi-person pose estimation from a single rgb image. In *IEEE international conference on computer vision (ICCV)*

Moreno-Noguer, F. (2017). 3d human pose estimation from a single image via distance matrix regression. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Mueller, F., Davis, M., Bernard, F., Sotnychenko, O., Verschoor, M., Otaduy, M. A., et al. (2019). Real-time pose and shape reconstruction of two interacting hands with a single depth camera. *ACM Transactions on Graphics (TOG) 38,* 1–13.

Natsume, R., Saito, S., Huang, Z., Chen, W., Ma, C., Li, H., & Morishima, S. (2019). Siclope: Silhouette-based clothed people. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. In *European conference on computer vision (ECCV).*

Nibali, A., He, Z., Morgan, S., & Prendergast, L. (2019). 3d human pose estimation with 2d marginal heatmaps. In *IEEE winter conference on applications of computer vision (WACV).*

Omran, M., Lassner, C., Pons-Moll, G., Gehler, P., & Schiele, B. (2018). Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In *International conference on 3D vision (3DV).*

Park, S., Hwang, J., & Kwak, N. (2016). 3d human pose estimation using convolutional neural networks with 2d pose information. In *European conference on computer vision (ECCV).*

Pavlakos, G., Zhou, X., & Daniilidis, K. (2018). Ordinal depth supervision for 3d human pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Pavlakos, G., Zhou, X., Derpanis, K. G., & Daniilidis, K. (2017a). Coarse-to-fine volumetric prediction for single-image 3d human pose. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Pavlakos, G., Zhou, X., Derpanis, K. G., & Daniilidis, K. (2017b). Harvesting multiple views for marker-less 3d human pose annotations. In *IEEE conference on computer vision and pattern recognition (CVPR).*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR) 12,* 2825–2830.

Peng, J., Sun, M., Zhang, Z., Tan, T., & Yan, J. (2019). Efficient neural architecture transformation search in channel-level for object detection. In *Conference on neural information processing systems (NeurIPS)*.

Qiu, H., Wang, C., Wang, J., Wang, N., & Zeng, W. (2019). Cross view fusion for 3d human pose estimation. In *IEEE international conference on computer vision (ICCV)*.

Rhodin, H., Spörri, J., Katircioglu, I., Constantin, V., Meyer, F., Müller, E., Salzmann, M., & Fua, P. (2018). Learning monocular 3d human pose estimation from multi-view images. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Rogez, G., & Schmid, C. (2016). Mocap-guided data augmentation for 3d pose estimation in the wild. In *Conference on neural information processing systems (NeurIPS)*.

Rogez, G., Weinzaepfel, P., & Schmid, C. (2017). Lcr-net: Localization-classification-regression for human pose. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Romero, J., Tzionas, D., & Black, M. J. (2017). Embodied hands: Modeling and capturing hands and bodies together. In *ACM transactions on graphics (TOG)*.

Sárándi, I., Linder, T., Arras, K. O., & Leibe, B. (2018). Synthetic occlusion augmentation with volumetric heatmaps for the 2018 eccv posetrack challenge on 3d human pose estimation. In *Workshop at european conference on computer vision (ECCVW)*.

Sárándi, I., Linder, T., Arras, K., & Leibe, B. (2020). Metric-scale truncation-robust heatmaps for 3d human pose estimation. In *IEEE international conference on automatic face and gesture recognition (FG)*.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., & Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Sun, X., Shang, J., Liang, S., & Wei, Y. (2017). Compositional human pose regression. In *IEEE international conference on computer vision (ICCV)*.

Sun, K., Xiao, B., Liu, D., & Wang, J. (2019). Deep high-resolution representation learning for human pose estimation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Sun, X., Xiao, B., Wei, F., Liang, S., & Wei, Y. (2018). Integral human pose regression. In *European conference on computer vision (ECCV)*.

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V. (2019). Mnasnet: Platform-aware neural architecture search for mobile. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Tang, W., & Wu, Y. (2019). Does learning specific features for related parts help human pose estimation? In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Tang, W., Yu, P., & Wu, Y. (2018). Deeply learned compositional models for human pose estimation. In *European conference on computer vision (ECCV)*.

Taylor, C. J. (2000). Reconstruction of articulated objects from point correspondences in a single uncalibrated image. In *Computer vision and image understanding (CVIU)*.

Tome, D., Russell, C., & Agapito, L. (2017). Lifting from the deep: Convolutional 3d pose estimation from a single image. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Tu, H., Wang, C., & Zeng, W. (2020). Voxelpose: Towards multi-camera 3d human pose estimation in wild environment. In *European conference on computer vision (ECCV)*.

Varol, G., Ceylan, D., Russell, B., Yang, J., Yumer, E., Laptev, I., & Schmid, C. (2018). Bodynet: Volumetric inference of 3d human body shapes. In *European conference on computer vision (ECCV)*.

Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M. J., Laptev, I., & Schmid, C. (2017). Learning from synthetic humans. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Wang, J., Huang, S., Wang, X., & Tao, D. (2019). Not all parts are created equal: 3d pose estimation by modeling bi-directional dependencies of body parts. In *IEEE international conference on computer vision (ICCV)*.

Wei, S. E., Ramakrishna, V., Kanade, T., & Sheikh, Y. (2016). Convolutional pose machines. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Wu, X., Finnegan, D., O'Neill, E., & Yang. Y. L. (2018). Handmap: Robust hand pose estimation via intermediate dense guidance map supervision. In *European conference on computer vision (ECCV)*.

Xiong, F., Zhang, B., Xiao, Y., Cao, Z., Yu, T., Zhou, J. T., & Yuan, J. (2019). A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image. In *IEEE international conference on computer vision (ICCV)*

Xu, Y., Xie, L., Zhang, X., Chen, X., Qi, G. J., Tian, Q., & Xiong, H. (2020). Pc-darts: Partial channel connections for memory-efficient architecture search. In *International Conference on Learning Representations (ICLR)*.

Yang, W., Ouyang, W., Wang, X., Ren, J., Li, H., & Wang, X. (2018). 3d human pose estimation in the wild by adversarial learning. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Yasin, H., Iqbal, U., Kruger, B., Weber, A., & Gall, J. (2016). A dual-source approach for 3d pose estimation from a single image. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhang, Y., Qiu, Z., Liu, J., Yao, T., Liu, D., & Mei, T. (2019). Customizable architecture search for semantic segmentation. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zheng, Z., Yu, T., Wei, Y., Dai, Q., & Liu, Y. (2019). Deephuman: 3d human reconstruction from a single image. In *IEEE international conference on computer vision (ICCV)*.

Zhou, K., Han, X., Jiang, N., Jia, K., & Lu, J. (2019). Hemlets pose: Learning part-centric heatmap triplets for accurate 3d human pose estimation. In *IEEE international conference on computer vision (ICCV)*.

Zhou, X., Huang, Q., Sun, X., Xue, X., & Wei, Y. (2017). Towards 3d human pose estimation in the wild: a weakly-supervised approach. In *IEEE international conference on computer vision (ICCV)*.

Zhou, X., Zhu, M., Leonardos, S., Derpanis, K. G., & Daniilidis, K. (2016). Sparseness meets deepness: 3d human pose estimation from monocular video. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhou, X., Zhu, M., Pavlakos, G., Leonardos, S., Derpanis, K. G., & Daniilidis, K. (2018). Monocap: Monocular human motion capture using a cnn coupled with a geometric prior. In *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*.

Zoph, B., & Le, Q. V. (2017). Neural architecture search with reinforcement learning. In *International Conference on Learning Representations (ICLR)*.