Pose Neural Fabrics Search

Sen Yang, Wankou Yang, *Member, IEEE* and Zhen Cui, *Member, IEEE*

Abstract—**Neural Architecture Search (NAS) technologies have emerged in many domains to jointly learn the architectures and weights of the neural network. However, most existing NAS works claim they are task-specific and focus only on optimizing a single architecture to replace a human-designed neural network, in fact, their search processes are almost independent of domain knowledge of the tasks. In this paper, we propose Pose Neural Fabrics Search (PoseNFS). We explore a new solution for NAS and human pose estimation task: part-specific neural architecture search, which can be seen as a variant of multi-task learning. Firstly, we design a new neural architecture search space, Cell-based Neural Fabric (CNF), to learn micro as well as macro neural architecture using a differentiable search strategy. Then, we view locating human keypoints as multiple disentangled prediction sub-tasks, and then use prior knowledge of body structure as guidance to search for multiple part-specific neural architectures for different human parts. After search, all these part-specific CNFs have distinct micro and macro architecture parameters. The results show that such knowledge-guided NAS-based architectures have obvious performance improvements to a hand-designed part-based baseline model. The experiments on MPII and MS-COCO datasets demonstrate that PoseNFS[1] can achieve comparable performance to some efficient and state-of-the-art methods.**

Index Terms—**Human pose estimation, neural architecture search, cell-based neural fabrics, micro and macro search space, prior knowledge, part-specific NAS, vector representation**
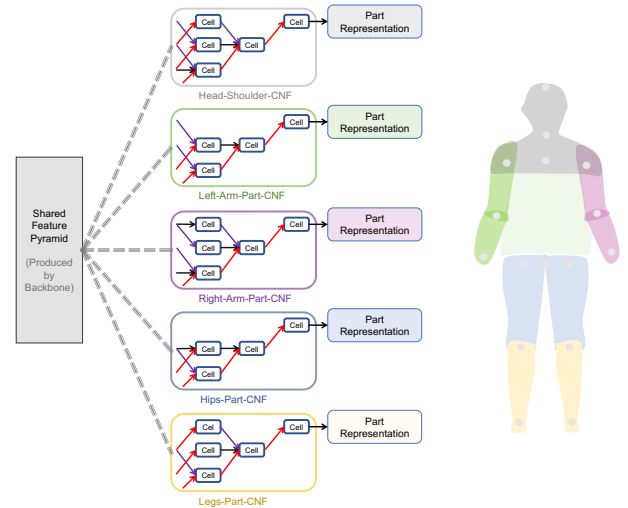

Fig. 1. A schematic diagram of searched architectures: five part-specific CNFs (Cell-based Neural Fabrics) are associated with different body parts according to the prior body structure. They take as input the shared feature pyramid produced from a common CNN backbone or a derived CNF. The weighted operations combinations in the cells are also distinct in the part-specific CNFs.

I. INTRODUCTION

NEURAL Architecture Search (NAS), the process of jointly learning the architecture and weights of the neural network [1]–[9], can play a potential role at designing efficient network architectures automatically. Current NAS methods mainly take image classification as basic task, and only search for a micro "cell" to build a chain-like structure. However, when applying NAS to dense (pixel-wise) prediction tasks such as semantic segmentation and human pose estimation, the micro search space is no longer able to generate more complex architectures. Therefore, it become a necessity to artificially design a macro search space allowing identifying a hierarchical structure upon cells for these tasks. In addition, most existing NAS works such as [1], [3], [9]–[11] optimize a single architecture in the search space, and finally obtain a so-called task-specific architecture to replace a human-designed architecture in the pipeline. Such practice, in fact, decouples the automating architecture engineering from the characteristics of tasks, failing to take advantage of significant

domain knowledge to guide the search process and achieve the expected targets.

For human pose estimation task, we argue that the prior knowledge of human body structure can help discover specific neural architectures in a part-based and automatic way. Thus, we propose a new paradigm: part specific neural architecture search, to search multiple neural architectures for different human parts with guide of prior knowledge, as shown in Fig. 1.

Naturally, the first step to introduce NAS into human pose estimation is to construct an architecture search space that can identify multi-scale, stacked or cascaded neural network. To this end, we propose a general parameterized Cell-based Neural Fabric (CNF), a scalable topology structure to encode micro and macro architecture parameters into cells. The discrete search space is relaxed into continuous search space to make it searchable by gradient descent. This design is motivated by Convolutional Neural Fabrics [12] and DARTS [1], it can be described as a neural fabric architecture woven by cells, as shown in Fig. 2.

In addition, there exists an inconsistency gap between the derived child network (sub-architecture) and the converged parent network (super-network) in DARTS [1]. Many works attempt to eliminate this bias such as SNAS-series works [10], [11]. In our work, we avoid it in a direct and simple way. We do not re-discretize the continuous architecture after searching, which means that all operations are densely preserved with macro and micro architecture parameters in both searching

Sen Yang and Wankou Yang are with the School of Automation of Southeast University, Nanjing 210096, China. E-mail: yangsenius@seu.edu.cn, wkyang@seu.edu.cn. (Corresponding author: Wankou Yang.)

Zhen Cui is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. E-mail: zhen.cui@njust.edu.cn.

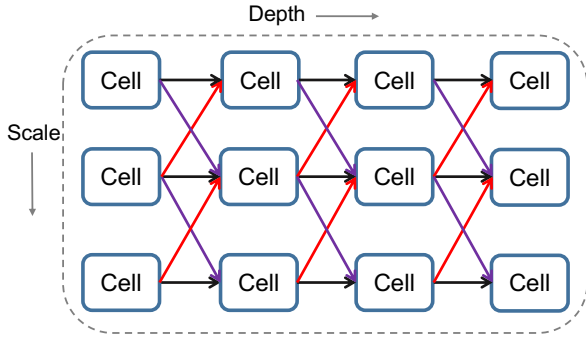[1]Code is available at https://github.com/yangsenius/PoseNFS

Fig. 2. A schematic map of the structure of Cell-based Neural Fabric (CNF). The neural fabric is woven by cells in different scales and layer depths. Black arrow represents identity transformation; purple arrow represents reducing spatial size and doubling the channels of feature maps; red arrow represents increasing spatial size and halving the channels of feature maps.

and evaluation stage. In order to verify such a setting, we test the performances of architectures randomly sampled from the continuous parameter space. Furthermore, we explore a simple yet effective method, gradient-based synchronous optimization, as the major search strategy to reduce the cost of time and computational budgets for the search process.

Designing the neural search space or search strategy is just the beginning for NAS, we believe domain knowledge can help NAS go further. For human pose estimation task, the special human body structure information is significant prior knowledge to be exploited. Current ConvNet-based human pose estimation methods [13]–[20] usually use FCN-like architecture to predict keypoints heatmaps. The global spatial relationship between all keypoints is implicitly learned by shared convolutional blocks and a single linear prediction head. Nevertheless, such learning methods may ignore the explicit knowledge: the spatial distributions of keypoints might be highly correlated in the short-range while weakly in the long-range. Tang et al. [21] have found that some pairs of body parts are weakly related by analyzing their mutual information, and a sharing mechanism using the same network to learn shared feature for all keypoints may cause negative transfer between uncorrelated parts. They therefore view human pose estimation as homogeneous multi-task learning (MTL) [22], [23], and design a structure-shared part-based branching network (PBN) to estimate related parts. Here we ask: since human pose estimation can be viewed as multiple prediction tasks, the internal characteristics of these tasks would be different, are structure-shared branching networks really better for multiple sub-tasks? Whether can NAS search for more specific neural architectures to localize these disentangled human parts?

As a matter of fact, such problems can be converted into searching multiple task-specific neural architectures for multiple sub-tasks. There also has been a new trend to explore Multi-task Neural Architecture Search such as [24]–[26], due to that MTL could improve generalization by leveraging the domain-specific information contained in the training signals of related tasks [22]. In this work, provided with the proposed search space, we can search multiple architectures to achieve

multiple learning objectives in a single shot, by exploiting simple prior knowledge as guidance. As a result, part-specific CNFs with distinct micro and macro architecture parameters are achieved to adapt to the characteristics of different sub-tasks. As shown in Fig. 3, highly correlated keypoints are grouped into the corresponding part representations that are predicted by searchable CNFs (multi-head predictions). Then each location of keypoint is inferred by the associated part representation. Such NAS-based part-specific architectures extend the carefully designed message passing mechanism, hand-designed branch or information fusion module [21], [27], [28] to fuse feature information between related joints. Besides, we replace scalar value by a vector in each pixel position of heatmaps, because the scalar value representing the existing probability of keypoints is still inadequate to encode local feature. We use the length (ℓ_2 norm) of the vector to represent the existing score of the keypoint and vector space to capture more local component information of the body part. In summary, our main contributions are:

- We propose a novel micro and macro architecture search space: parameterized Cell-based Neural Fabric (CNF).
- With simple prior knowledge as guidance, our method automatically searches part-specific neural architectures to localize disentangled body parts, which extends the traditional part-based methods.
- Such part-specific neural architecture search can be seen as a variant of multi-task learning. It is a novel NAS paradigm as Multi-Task Neural Architecture Search for human pose estimation task.
- The experiments show that NAS-based part-specific architectures have obvious performance improvements to a hand-designed part-based baseline model. And our models achieve comparable performances to some efficient and state-of-the-art methods.

II. RELATED WORK

a) Neural Architecture Search: Our work is motivated by convolutional neural fabrics [12] and neural architecture search methods [1], [4], [8], [9]. Liu et al. [1] propose the continuous relaxation for architecture representation to search architectures using differentiable strategy. Chen et al. [4] adopt random search to search Dense Prediction Cell architecture for dense image prediction. Ghiasi et al. [3] use Reinforce Learning to explore more connection possibilities in different scales of feature pyramid network. Xie et al. [8] explore randomly wired networks for image classification and proposed the concept of *network generator*. Liu et al. [9] propose Auto-DeepLab, searching for a hierarchical neural network as backbone to replace original human-designed network in a common pipeline (DeepLab) for semantic segmentation. It aims to search a cell structure and a better path in multiple branches. In contrast, our architecture space construction method is one-step, unlike the two-step construction scheme for architecture search space in AutoDeepLab, the whole architecture at macro and micro level is totally parameterized as each cell's parameters and not pruned into sparsely connected architecture. Our work also can be viewed as Multi-task Neural Architecture Search. There are similar works like [24]–[26].
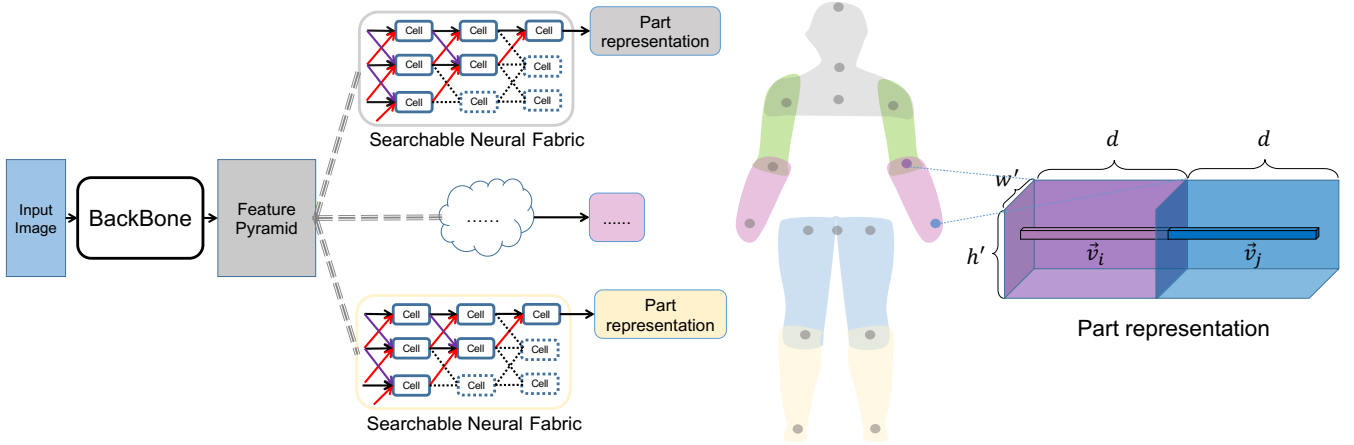
Fig. 3. Pose neural fabrics search framework. **Left:** Given an input image, feature pyramid representing multi-scale feature maps will be produced from backbone network. Then there will be P CNFs receiving the same feature pyramid and predicting P part representations, here two CNFs shown for simplification. The final cell in the highest scale produces the part representation. Dashed lines mean unused connections and cells. **Mid:** The whole body is divided into multiple parts associated with keypoints. **Right:** For instance, right lower arm part representation is associated with the wrist and elbow keypoints. $\vec{v}_i$ and $\vec{v}_j$ mean two d-dim vectors respectively for wrist and elbow keypoints at location (x, y) of part representation feature map, and it's shape is $2 \times w' \times h' \times d$.

b) Human pose estimation and part-based model: Top-down [13], [16], [18]–[20], [29] and bottom-up [14], [30]–[33] human pose estimation approaches have been proven extremely successful in learning global or long-range dependencies relationships of body pose. However, parts occlusions, viewpoint variations, crowed scene, and background interference etc. are still tough problems. Compositional structure models or part-based models [21], [34]–[44] attempt to overcome aforementioned problems by representing the human body as a hierarchy of parts and subparts. Early part-based models [42], [45], pictorial structure models [38]–[40] and some of the current models [21], [27], [35], [46]–[48] are usually based on hand-crafted feature (e.g. HOG, SIFT), hand-designed convolutional neural networks or hand-designed information fusion modules. Our method also exploits the compositionality of body pose to separately predict related keypoints, but further develops it by employing NAS to learn distinct CNFs for different keypoints groups.

c) Vector Representation: The vector in pixel method is motivated by embedding and vector representation methods [14], [15], [33], [49]–[51]. Newell et al. [33] propose associative embedding to group body keypoints. Papandreou et al. [49] use geometric embedding representation to predict offset vectors of keypoints. Cao et al. [14] use part affinity vector field to supervise the part prediction. In addition, Hinton et al. [50] use matrix with extra scalar to represent an entity. Sabour et al. [51] propose Activity Vector, its length can represent existing probability and its orientation represents the instantiation parameters. In this work, we view each type of keypoint as a type of entity in the image and use activity vectors to locate keypoints to estimate human pose.

III. POSE NEURAL FABRICS SEARCH

A. Overview

Based on top-down method, 2D human pose estimation aims to locate all K keypoints coordinates of body joints(e.g.,
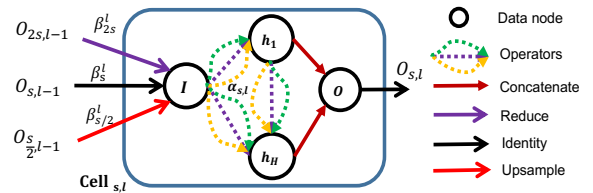

Fig. 4. An overview of inner structure of a cell in scale s and l layer. To simply present the inner structure of cell, we set the number of hidden nodes H as 2. Actually, its number of hidden nodes can be 1 or more than 2, each hidden node h_H is densely connected from its previous nodes $\{h_0, h_1, ..., h_H\}$.

shoulder, wrist, knee, etc) in a single pose. Let $S = \{(x_i, y_i) | x, y \in \mathbb{R}^+, i = 1, 2, ..., K\}$ denote the set consisting of all keypoints coordinates. Considering human body skeleton structures, we disentangle the whole body pose into P part representations as P prediction sub-tasks. P subnetworks are constructed from CNFs sharing backbone to predict related keypoints subset s ($s \subseteq S$) whose element is associated with the corresponding part. The vector in pixel method is introduced to capture keypoint's feature in a relaxed vector space and the prediction of specified keypoint's location is determined by the location of vector $\vec{v}$ whose length is the largest. The entire framework is shown in Fig. 3.

In Section III-B, III-C, III-F, we demonstrate how to design and employ parameterized CNFs as the choice of subnetworks and backbone. Then, we describe how to randomly sample architectures and optimize the models by synchronous optimization. In Section III-E, we describe what prior structures of human body are employed to guide neural architecture search. In Section III-D, we demonstrate how to utilize the vector representation to estimate keypoints' locations.

B. Neural Architecture Search Space

Micro structure. Cell is a repeatable unit across different layers and scales of the whole architecture. Illustrated in
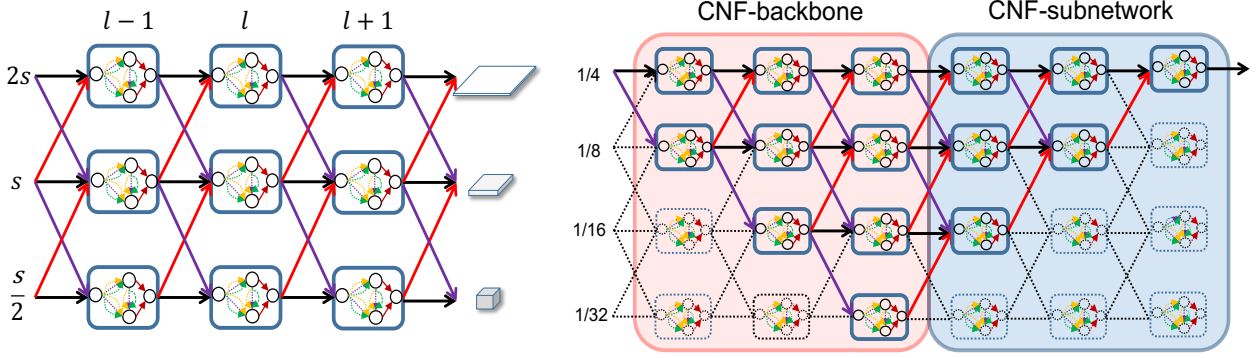
Fig. 5. An overview of the CNF neural search space. **Left:** The homogeneous local connectivity between cells in a neural fabric. **Right**: Examples of constructing a CNF-backbone (red box) or a CNF-subnetwork (blue box) from CNF. Dashed lines mean unused connections and cells.

Fig. 4, it receives outputs from previous layer's cells as its single input node I and it has H nodes as it's hidden states. Each hidden node h_j is connected by a directed edge with each element of candidate nodes set $\{h_0, h_1, h_2, ..., h_{j-1}\}$ ($h_0 = I, j = 1, 2, ..., H$). Continuous Relaxation [1] method is adopted to represent each directed edge with mixed operations. For each h_j is computed by:

$$h_j = \sum_{i=0}^{j-1} \sum_{o \in \mathcal{O}} \frac{\exp\left(\alpha_o^{(i,j)}\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\alpha_{o'}^{(i,j)}\right)} o(h_i), \quad (1)$$

where $\mathcal{O}$ is the set of candidate operations. We choose 6 types of basic operations o, consisting of: zero, skip connection, 3×3 depthwise separable convolution, 3×3 dilated convolution with 2 rate, 3×3 average pooling, 3×3 max pooling. $\alpha_o^{(i,j)}$ means the associated weight for each operation $o \in \mathcal{O}$ in edge $h_i \rightarrow h_j$. For a specified $Cell_{s,l}$ in scale s and layer l in CNF, the continuous search space at micro level is: $\alpha_{s,l} = \left\{\alpha_o^{(i,j)} | \forall o \in \mathcal{O}, \forall j \in \{1, 2, ..., H\}, \forall i \in \{0, 1, ..., j-1\}\right\}$, $\alpha_{s,l} \in \mathbb{R}^{\|\mathcal{O}\| \times \frac{H(H+1)}{2}}$. All hidden nodes $\{h_1, h_2, ..., h_H\}$ are concatenated together and reduced in channels by a 1×1 conv to achieve an independent output node $O_{s,l}$.

Macro Structure. For $Cell_{s,l}$, it receives the sum of outputs from cells in the previous layer: $O_{2s,l-1}$, $O_{\frac{s}{2},l-1}$ and $O_{s,l-1}$. They are associated with macro architecture parameters $\beta_{s,l} = \left\{\beta_{2s}^l, \beta_s^l, \beta_{\frac{s}{2}}^l\right\} \in \mathbb{R}^3$, which are normalized to control different information reception level from previous cells in different scales. All $\beta_{s,l}$ in all cells of the fabric construct the macro continuous search space. For each $Cell_{s,l}$, its I is computed by:

$$I = h_0 = \sum_{(O,\beta) \in Z_{s,l}} \frac{\exp(\beta)}{\sum_{\beta' \in \beta_{s,l}} \exp(\beta')} \mathcal{T}(O), \quad (2)$$

where $Z_{s,l} = \left\{(O_{2s,l-1}, \beta_{2s}^l), (O_{s,l-1}, \beta_s^l), (O_{\frac{s}{2},l-1}, \beta_{\frac{s}{2}}^l)\right\}$ and $\mathcal{T}(\cdot)$ is scale transformation operation. In particular, $\mathcal{T}(O_{2s,l-1})$ is downsampling operation via Conv-BN-ReLU mode with 2 stride (meanwhile doubling the channels of data node), $\mathcal{T}(O_{\frac{s}{2},l-1})$ is upsampling operation via bilinear interpolation (meanwhile halving the channels of data node by 1×1 conv) and $\mathcal{T}(O_{s,l-1})$ means identity transformation.

Parametric Form. In summary, we parameterize the form of cell in the l-th layer and s-th scale of CNF in such a pattern:

$$O_{s,l} = \text{Cell}_{s,l}\left(O_{2s,l-1}, O_{s,l-1}, O_{\frac{s}{2},l-1}; w_{s,l}, \alpha_{s,l}, \beta_{s,l}\right), \quad (3)$$

where $w_{s,l}$ represents the weights of all operations in each cell, $\alpha_{s,l}$ and $\beta_{s,l}$ encode architecture search space inside the cell. The hyperparameter of each cell is $\theta_{s,l} = (H, C, \mathcal{O})$. H is the number of hidden nodes in each cell. C is the channel factor for each node to control the model capacity, i.e., the number of channels of its data node is $C \times \frac{1}{s}$. $\mathcal{O}$ is the set of candidate operations. This form also can be seen as a scalable topology structure or a network generator [8], which can map from a parameter space Θ to a space of neural network architectures $\mathcal{N}$, formulated as $g : \Theta \longmapsto \mathcal{N}$.

C. Constructing Subnetworks or Backbone

Benefit from its local homogeneous connectivity pattern as shown in the left of Fig. 5, cell-based fabric is very flexible and easy to extend into different layers and scales for high and low resource use cases. It is determined by a group of hyperparameters $\Theta = (L, \{1/1, 1/2, ..., 1/2^b\}, H, C, \mathcal{O})$ where L is total layers and $1/2^b$ is the smallest scale. Illustrated in Fig. 5, fabric backbone can be constructed by reserving the first m layers and discarding the latter $L - m$ layers to produce feature pyramid in multiple scales. Likewise, fabric subnetwork can be constructed by reserving the latter n layers and discarding the first $L - n$ layers to receive feature pyramid from backbone. Note that our backbone is not restricted to the proposed architecture, and we do not use 3×3 average pooling and 3×3 max pooling as candidate operations for subnetworks as they are empirically more suitable for extracting low-level visual feature rather high-level feature.

Following common practice for pose estimation, the smallest scale is set to 1/32. We use a two-layer convolutional stem structure to firstly reduce the resolution to 1/4 scale, and consecutively weave the whole CNF [2]. In order to achieve a higher resolution feature map to locate keypoints' coordinates, we just use the final cell's output in 1/4 scale as the part representation.

---

[2] For cells in first layer, they only receive the stem's output. And for cells in 1/32 scale or 1/4 scale of its current layer, it may have only have two outputs from previous cells. In this case, we will copy one of candidate inputs.

Finally, we use P subnetworks (multi-head) sharing backbone to produce P part representations, as shown in Fig. 3. Thus, the local information and pairwise relationships between highly correlated keypoints are combined in each part representation and predicted by each subnetwork. The global information and constraint relationships among all parts are implicitly learned by the shared feature and predicted by the backbone. The long-range and short-range constraint relationships of the human pose are enforced by the whole architecture in an end-to-end learning method.

D. Body Part Representation with Vector in Pixel

Based on the top-down method of pose estimation, we estimate human pose with single person proposal. Given a input image $I \in \mathbb{R}^{H \times W \times 3}$ centered at a person proposal, there will be P part representations $T_1, T_2, ..., T_P$ to be predicted. Let $T_p \in \mathbb{R}^{J \times h' \times w' \times d}$ denote its p-th body part representation, where J is the number of keypoints belonging to this part (see the assignment in Tab I), i.e. a part what we mean here may associate several keypoints. h' and w' are the height and width of part representation; in our down-sampling setting, $h'/H = w'/W = 1/4$. d is the dimension of vector in each pixel position. For the i-th keypoints of T_p, vector in position (x, y) is denoted as $\vec{v}_{i,x,y} = T_p^i(x,y) \in \mathbb{R}^d$, simplified as $\vec{v}$. Note that the dimension d of vector is set to 8 by default and choice for dimension is discussed in Section IV .

We relax scalar value into a latent vector as keypoint entity in each image pixel, expecting it to implicitly capture the redundant and local feature information around the keypoint position, please see more explanation in Appendix A.

Besides inherent to the characteristics of encoding locations, $\vec{v}$ can represent existing probability of keypoints by using Squashing Function $f_S(\cdot)$ [51] to normalize its ℓ_2 norm to $[0, 1)$. Formally, for i-th keypoint of p-th part, the non-linear Squash function will compute the squashed vector $\vec{v}_s$ in each position (x, y) of T_p^i by:

$$\vec{v}_s = \frac{\|\vec{v}\|^2}{1 + \|\vec{v}\|^2} \frac{\vec{v}}{\|\vec{v}\|}, \tag{4}$$

$$\|\vec{v}_s\| = f_S(\vec{v}) = \frac{\|\vec{v}\|^2}{1 + \|\vec{v}\|^2}, \tag{5}$$

where $\|\vec{v}_s\|$ exactly represents i-th keypoint's existing score in position (x, y). In inference, the position $(\bar{x}, \bar{y})$ of the longest (max-norm) $\vec{v}$ will be regarded as keypoint location. Predicted score maps are collected from all part presentations. Commonly, groundtruth score map H_k^{gt} is generated from the groundtruth of $k-$th keypoint's position by applying 2D Gaussian with deviation of σ where the peak value equals 1 and σ controls the spread of the peak. Train loss $\mathcal{L}_{train}$ is computed by Mean Square Error (MSE) between the predicted score maps and groundtruth score maps for all P part representations.

Note that the i-th keypoint of the part T_p is some type of all keypoints, and several parts may contain the same type of keypoint, e.g. elbow maybe fall into the upper arm part and the lower arm part. Thus we make a indicator function

$\mathbb{1}(k, p, i)$ equal 1 if the local index i-th keypoint of the part T_p is the global index k-th type of all keypoints otherwise 0. And the final position $(\bar{x}, \bar{y})$ will be inferred by the prediction summed from these parts (same effect by averaging). We finally formulate the training loss as:

$$\mathcal{L}_{train} = \frac{1}{K} \sum_{k=1}^{K} \sum_x \sum_y \|\hat{H}_k(x,y) - H_k^{gt}(x,y)\|_2^2, \tag{6}$$

$$\hat{H}_k(x,y) = \sum_{p,i} f_S\left(T_p^i(x,y)\right) \cdot \mathbb{1}(k,p,i), \tag{7}$$

where $\hat{H}_k$ is the prediction map for each keypoint, (x, y) is each location of T_p^i, $\hat{H}_k$ and H_k^{gt}. In practice, we will mask out the keypoints without position annotation in the training process.

E. Prior Knowledge of human body structure

The intuition behind this work is that using part-specific neural networks to separately capture each pairwise spatial relationship in local part is possibly better than using a shared neural network to model the global relationship. Therefore, we consider the human body structures and adopt four types of grouping strategy to make comparison. Specially, $P = 1$ means that we model long-range dependencies relationships of pose and global relationship of all joints is learned by a shared neural network. $P = 3$ means that body pose is predefined into 3 parts associated with 3 CNFs: head part, upper limb part and lower limb part. $P = 8$ means that body pose is predefined into 8 parts associated with 8 CNFs: head-shoulder, left upper arm, left lower arm, right upper arm, right lower arm, thigh, left lower leg and right lower leg. In addition, we also adopt the data-driven grouping strategy of [21] by setting $P = 5$: head-shoulder, left lower arm, right lower arm, thigh, lower limb part. In Tab I, all skeleton keypoints are associated with the corresponding part according to the prior body structure.[3]

F. Optimization

One-shot Search and Part-specific architectures for Different Parts. Given a hyperparameter Θ for a CNF, the weights $w_o = \{w_{s,l}\}$ and architecture $\alpha_o = \{\alpha_{s,l}\}, \beta_o = \{\beta_{s,l}\}$ are optimized. Following the principle of one-shot architecture search [6], we assume that $\alpha_{s,l}$ share same weights across a single CNF and $\beta_{s,l}$ is cell-wise in each CNF. Supposing that there are P CNFs, their weights of operations are $w = \{w_0, ..., w_P\}$ and the total architecture parameters are $\alpha = \{\alpha_1, ..., \alpha_P\}, \beta = \{\beta_1, ..., \beta_P\}$. We search for part-specific CNF to adapt each prediction sub-task, which means that the architecture parameters $\alpha_1, ..., \alpha_P$ are totally distinct and so are $\beta_1, ..., \beta_P$. In the Section IV-B0b, we make contrastive experiments to study the differences on different keypoints groups strategies.

---

[3]For MPII [52] dataset, we set indices of head top, upper neck, thorax, l-shoulder, r-shoulder, l-elbow, r-elbow, l-wrist, r-wrist, l-hip, r-hip, l-knee, r-knee, l-ankle, r-ankle, pelvis keypoints to 0-15 orderly. For COCO [53] dataset, we set indices of nose, l-eye, r-eye, l-ear, r-ear, l-shoulder, r-shoulder, l-elbow, r-elbow, l-wrist, r-wrist, l-hip, r-hip, l-knee, r-knee, l-ankle and r-ankle to 0-16 orderly.

TABLE I
ACCORDING TO THE PRIOR KNOWLEDGE OF HUMAN BODY STRUCTURE, THERE ARE DIFFERENT GROUPING TYPES OF BODY PART REPRESENTATIONS.

Representation Mode	Group Name	Index	
		MPII	COCO
$P = 1$	all keypoints	0-15	0-16
$P = 3$	head part	0-2	0-4
	upper limb part	3-8	5-10
	lower limb part	9-15	11-16
$P = 5$	head-shoulder	0-4	0-6
	left lower arm	5,7	7,9
	right lower arm	6,8	8,10
	thigh	9,10,15	11,12
	lower limb part	11-14	13-16
$P = 8$	head-shoulder	0-4	0-6
	left upper arm	3,5	5,7
	left lower arm	5,7	7,9
	right upper arm	4,6	6,8
	right lower arm	6,8	8,10
	thigh	9-12,15	11-14
	left lower leg	11,13	13,15
	right lower leg	12,14	14,16

Random Sampling. Random search can be seen as a powerful baseline for neural architecture search or hyperparameter optimization [8], [54], [55]. It is conducted in [1] as well and has a competitive result compared with the gradient-based method. In this work, we randomly initialize values of α, β by standard normal distribution and make them fixed in the whole training process. From another point of view, this also can be viewed as a *stochastic* network generator like [8]. The sampled architecture parameters make no assumption about the structures, and only the weights of neural networks are optimized. Therefore, we conduct it to validate the design of CNF, each random experiment also can represent the performance of CNF without any neural architecture search strategies.

Synchronous Optimization. In DARTS [1], the architecture search problem is regarded as a bilevel optimization problem. An extra subset val of original train set is held out serving as performance validation to produce the gradient w.r.t. architecture parameters α, β excluding the weights w. However, the training of the second-order gradient-based method is still time-consuming and restricted by GPU memory due to the high resolution intermediate representation for pose estimation. Moreover, the training for parent continuous network and the derived net are inconsistent in DARTS, final pruned network needs to be trained again with all training samples. There are works attempting to eliminate this problem, such as SNAS [10] introducing Gumbel-Softmax trick. Instead, we explore a more simple way as our major optimization strategy, benefit from the parametric form of cell. The α and β are registered to model's parameters, synchronously optimized with the weights w, i.e. the w, α and β are updated by $\nabla_{w,\alpha,\beta}\mathcal{L}_{train}$ in a single step of gradient descent. Without extra validation phase, the final continuous α, β and the weights w have seen all training samples, thus it do not need to be pruned into discrete architecture by $argmax$ operation and trained again from scratch.

IV. EXPERIMENTS

A series of experiments have been conducted on two datasets MPII Human Pose Dataset [52] and COCO Keypoint [53]. In Section IV-A and Section IV-B, we show the implementation details and ablation study for the effectiveness of the purposed optimization strategies, part-based pose representation and the vector in pixel method. Then we compare our model with a part-based baseline model and an efficient light-weight model, in Section IV-C. Finally, in Section IV-D, we present the results compared with the state-of-the-art.

A. Implementation Details

As for most subnetworks in ablation study, we set C=10 and total final layers is 3 (discarding first 3 layers of CNF architecture with $L = 6$), and the total number of cells is 6 as a basic configuration. Backbone with fabric can be constructed as described in Section III-C. To make fair comparison with methods using model pretrained on ImageNet [56], we take Fabric-1, Fabric-2, Fabric-3, pretrained Mobilenet-V2 [57], ResNet-50 [58] and HRNet-W32-Stem-Stage3 [20] feature blocks (5.8M, 6.8M, 10.5M, 1.3M, 23.5M, 8.1M parameters respectively) as choices for backbone to provide feature pyramid to subnetworks, see configuration details in Appendix B-A and B-B.

We implement our work by PyTorch [59] and each experiment is conducted on a single NVIDIA Titan Xp GPU. Training epoch is 200 and batchsize is set to 24 (not fixed). We use Adam [60] optimizer to update the weights and architecture parameters with 0.001 initial learning rate, decay at epoch 90, 120, 150 with 0.25 factor by default. Data augmentation strategies are used with random rotation range in $[-45°, 45°]$, random scale range in $[0.7, 1.3]$ and random flipping with 0.5 probability. Flip test is used in inference. In practice, one quarter offset from the peak to the secondary peak is introduced to reduce the quantization error. Strategies mentioned above are adopted in all ablation experiments.

B. Ablation Study

Dataset and Evaluation. We conduct ablation study on MPII Human Pose Dataset [52] which is a benchmark for evaluation of pose estimation. The dataset consist of around 25K images containing over 40K people with annotated body joints. All models in ablation study experiments are trained on a subset MPII training set and evaluate on a held validation set of 2958 images following [19]. The standard PCKh metric (head-normalized probability of correct keypoint) is used for MPII. PCKh@0.5 means that a predicted joint is correct if its position is within 50% of the length groudtruth head box from its groundtruth location. Evaluation procedure reports the PCKh@0.5 of head, shoulder, elbow, wrist, hip, knee, ankle, mean PCKh@0.5 and mean PCKh@0.1.

a) Optimization Strategies: For random search strategy, we conduct 5 experiments with different pseudo random seeds, each experiment costs 0.8 days for a single GPU. Result shows that completely random architecture parameters can perform well. As shown in Tab II, synchronous optimization is effective as well as random search under the same configuration

TABLE II
OPTIMIZATION STRATEGIES (SEARCH METHOD). WE CHOOSE
MOBILENET-V2 AS THE BACKBONE OF MODEL.
$P = 3, H = 1, C = 10, d = 8$, EACH SUBNETWORK HAS SIX CELLS AND
TOTAL PARAMETERS OF MODEL IS 3.3M, THE MADDS OF MODEL
INFERENCE COMPLEXITY FOR SINGLE INPUT SAMPLE IS 1.2 GFLOPS.

Search Method	Search Time	Mean	Mean
(search strategy)	(GPU days)	(PCKh@0.5)	(PCKh@0.1)
Random	0.8	87.1±0.2	35.2±0.4
First-order gradient-based	0.9	87.1	34.7
Synchronous gradient-based	0.8	87.0	34.6

TABLE III
BODY PART REPRESENTATION MODES. WE CHOOSE MOBILENET-V2 AS
THE BACKBONE OF MODEL. $H = 1, C = 10, d = 8$, EACH SUBNETWORK
HAS SIX CELLS.

Representation Mode	Mean(PCKh@0.5)	Mean(PCKh@0.1)
$P = 1$	86.4	33.4
$P = 3$	87.0	34.6
$P = 5$	87.1	35.1
$P = 8$	87.3	35.6

and search time. We observe that the best result of random initialization for architecture surpasses the synchronous optimization, this actually reveals two points: 1) the search space design has more significant impact on the performance of neural network; 2) the parameter initialization becomes important when not leveraging NAS to the proposed CNF structure. In addition, we implement the first-order gradient-based optimization method according to the official code[4] of DARTS [1] for comparison. We hold out half of MPII training data as validation for performance estimation of architecture. Another Adam optimizer is used to update α, β with 0.003 learning rate and 0.001 weight decay, discrete architecture is not derived from continuous architecture for full training. Note that we can not theoretically prove the found architecture is the optimal one, but we find that there is almost no performance difference between the synchronous optimization and the first-order gradient-based optimization proposed by DARTS, which

[4]https://github.com/quark0/darts/blob/master/cnn/architect.py

TABLE IV
DIMENSION CHOICES FOR VECTOR IN PIXEL. WE CHOOSE
MOBILENET-V2 AS THE BACKBONE OF MODEL. $P = 3, H = 1, C = 10$,
EACH SUBNETWORK HAS SIX CELLS AND TOTAL PARAMETERS OF MODEL
IS 3.3M, THE MADDS (FLOPs) OF MODEL INFERENCE COMPLEXITY FOR
SINGLE INPUT SAMPLE IS 1.2 GFLOPS. † IS THE RESULT ACHIEVED BY
RUNNING THE OFFICIAL CODE ON OUR MACHINE, THE OFFICIAL IS 90.3.

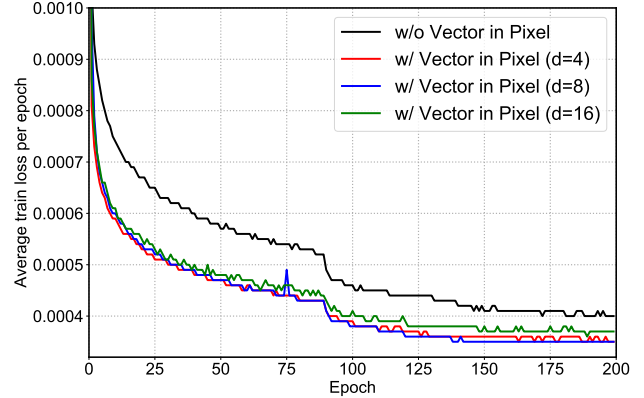| Dimension	Mean(PCKh@0.5)	Mean(PCKh@0.1)	#Params	#Madds(FLOPs)
$d = 1 (scalar)$	86.8	33.5	3.3M	1.1G
$d = 4$	86.9	34.9	3.3M	1.1G
$d = 8$	87.0	34.6	3.3M	1.2G
$d = 16$	86.8	34.9	3.3M	1.2G
SimpleBaseline [19]	88.5	33.9	34.0M	12.0G
+ vector(8-dim)	88.7	34.2	34.0M	12.1G
HRNet [20]	90.1†	37.7	28.5M	9.5G
+ vector(8-dim)	90.2	38.1	28.5M	9.6G

Fig. 6. Train losses of experiments w and w/o vector in pixel method. Detailed configurations are described in Tab IV. The sudden drop at epoch 90 is caused by learning rate decay.

demonstrates its effectiveness compared with the first-order optimization.

b) Body Part Representation Modes: We study these four modes predefined by the prior knowledge and results are shown in Tab III. We choose MobileNet-v2 [57] feature blocks as backbone. We find that multiple part presentations predicted by part-specific CNFs surpasses global whole-body representation predicted by a shared CNF. 8 part representations mode achieve 1% accuracy increase than whole-body representation and mode with $P = 3$ is a trade-off between performance and model capacity.

c) Dimension Choices for Vector in Pixel: We study the effect of choice for dimension d of the vector on performance by setting d with $1, 4, 8, 16$. $d = 1$ represents the common heatmap regression approach without vector in pixel. We find that 8-dim vector has a better performance shown in Tab IV. To validate the generalization of 8-dim vector representation method, we apply it to SimpleBaseline [5] [19] and HRNet [6] [20]. We find that this 8-dim vector representation is effective in these two frameworks. It gains 0.23% increase in PCKh@0.5 and 0.88% increase in PCKh@0.1 than Simple-Baseline official results with little increase of complexity and 1.06% increase in PCKh@0.1 than HRNet official results. Although we find that there is no obvious boost on PCKh@0.5 metric and a little in PCKh@0.1 metric, from Fig. 6 we observe that the losses of training with vector in pixel method converge faster, which implies the fitting between training data and label is more robust.

C. Comparison with Baseline Model and Efficient Model

a) Combination and Comparison with Baseline Model: [21] conducts part-specific feature learning for pose estimation as well. It uses a part-based branch network (PBN) that has a shared representation extracted by a Hourglass Network and is stacked by 8 times. It represents our main competitor. For the sake of fairness, we follow the [21] to construct the Branchnet

[5]https://github.com/microsoft/human-pose-estimation.pytorch
[6]https://github.com/leoxiaobin/deep-high-resolution-net.pytorch

TABLE V
COMPARISONS OF PERFORMANCE, MODEL PARAMETERS AND INFERENCE COMPLEXITY ON MPII TEST SET. THE BACKBONES OF OURS-A AND OURS-B MODELS ARE MOBILENET-V2 (1.3M) AND HRNET-W32-STEM~STAGE3 (8.1M).

Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total	#Params	#FLOPs
Tompson et al. [46]	95.8	90.3	80.5	74.3	77.6	69.7	62.8	79.6	-	-
Belagiannis & Zisserman [34]	97.7	95.0	88.2	83.0	87.9	82.6	78.4	88.1	-	-
Wei et al. [13]	97.8	95.0	88.7	84.0	88.4	82.8	79.4	88.5	-	-
Insafutdinov et al. [30]	96.8	95.2	89.3	84.4	88.4	83.4	78.0	88.5	42.6M	41.2G
Bulat&Tzimiropoulos [61]	97.9	95.1	89.9	85.3	89.4	85.7	81.7	89.7	-	-
Newell et al. [62]	98.2	96.3	91.2	87.1	90.1	87.4	83.6	90.9	25.1M	19.1G
Xiao et al. [19]	98.5	96.6	91.9	87.6	91.1	88.1	84.1	91.5	68.6M	20.9G
Tang et al. [35]	98.4	96.9	92.6	88.7	91.8	89.4	86.2	92.3	15.5M	33.6G
FPD (Knowledge Distillation) [63]	98.3	96.4	91.5	87.4	90.9	87.1	83.7	91.1	3M	9G
Ours-a	97.9	95.6	90.7	86.5	89.8	86.0	81.5	90.2	5.2M	4.6G
Ours-b	98.2	95.9	91.5	87.6	90.1	87.3	83.2	91.0	16.4M	9.4G

TABLE VI
COMPARISON WITH BASELINE MODELS ON MPII VALIDATION SET. WE CHOOSE HRNET-W32-STEM~STAGE3 AS THE BACKBONE, WHICH OUTPUTS SHARED FEATURE PYRAMIDS WITH FOUR LEVELS FO FEATURE MAPS. INPUT SIZE IS 256×256. OUR-C-CNF: $P = 3, H = 2, C = 12$. OUR-B-CNF: $P = 5, H = 1, C = 16$. $\mathcal{O}$ INCLUDES ZERO, SKIP CONNECTION, 3×3 SEPARABLE CONV, 3×3 DILATED CONV WITH 2 RATE. HERE WE USE A 1×1 CONVOLUTION TO FIRST REDUCE THE FEATURE DIMENSION TO $W = 400/296$ AND THEN $D = 8/8$ SUBSEQUENT RESIDUAL BLOCKS FOR THE BRANCHNETS OF BASELINE-1 AND BASELINE-2. † IS THE RESULT ACHIEVED BY RUNNING THE OFFICIAL CODE ON OUR MACHINE, THE OFFICIAL RESULT IS 90.3 AP. ‡ REPRESENTS THE METRIC MEAN PCKH@0.5 OVER TEN HARD JOINTS, REPORTED IN [21].

Method	Backbone	Head	NAS	Part-specific	Mean@0.5	Mean@0.1	#Params	#FLOPs
SimpleBaseline [19]	ResNet-152 (52.2M)	DeConvs (16.4M)	✗	✗	89.6	35.0	68.6M	20.9G
HRNet-W32 [20]	Stem~stage3 (8.1M)	Stage4(19.7M)	✗	✗	90.1†	37.7	27.8M	9.5G
Stacked PBNs [21]	Stacked Hourglass Network	Stacked Branchnets	✗	✓	88.14‡	-	26.7M	-
Baseline-1	Stem~stage3 (8.1M)	Branchnet × 3 (4.7M)	✗	✓	89.0	38.2	12.8M	23.7G
Baseline-2	Stem~stage3 (8.1M)	Branchnet × 5 (4.5M)	✗	✓	89.3	38.7	12.6M	22.8G
Random partition	Stem~stage3 (8.1M)	CNF × 3 (4.8M)	✓	✓	81.0	30.5	12.9M	8.3G
Ours-c	Stem~stage3 (8.1M)	CNF × 3 (4.8M)	✓	✓	89.9	39.5	12.9M	8.3G
Ours-b	Stem~stage3 (8.1M)	CNF × 5 (8.3M)	✓	✓	90.1	39.4	16.4M	9.4G

serving as the subnetworks but not stack the whole module repeatedly. We choose the ImageNet pretrained HRNet-W32 [20] as backbone by replacing blocks of the last stage, which only retains 8.1M parameters. And then we make a comparison between NAS-based CNFs and hand-designed Branchnets. By controlling their parameters amount to be similar, we find that using NAS to learn part-specific CNFs has achieved significantly higher AP (89.9 vs. 89.0) and less FLOPs (8.3G vs. 23.7G) than using a fixed structure for each part, as shown in Tab VI.

With 16.4M model size and 9.4 GFLOPs, our model achieves 90.1/39.4AP accuracy in PCKh@0.5/PCKh@0.1 metrics, in contrast to 90.3/37.7AP reported from [20] by HRNet-W32 (28.5M, 9.5G FLOPs) in single-scale testing. We can see that with obviously fewer parameters (↓58%) compared with the stage4 block of HRNet, the searched CNFs can maintain the high performance at PCKh@0.5 metric and improve PCKh@0.1 metric performance significantly.

To validate whether the prior knowledge is significant to help NAS search part-specific neural architectures, we make a contrastive experiment by randomly partitioning keypoints into three groups: 1) l-shoulder, l-ankle, l-elbow, pelvis, r-wrist, head-top; 2) upper-neck, r-knee, r-elbow, r-hip, l-wrist, throax; 3) l-hip, r-shoulder, l-knee, r-ankle. Such partitioned groups can be seen as wrong knowledge of human body structure or chaotic training signal. We use the same architecture configurations as Ours-c model, with 12.9M parameters and 9.3 GFLOPs. The best accuracy in the training process is only 81.0 AP and the model collapses at 120 epoch with 41.7 AP, as shown in Tab. VI. This result indicate that the guidance of correct prior knowledge can help search higher-performances neural architectures, but incorrect targets also make NAS degenerate. We believe that the intervention of appropriate domain knowledge would be significant for NAS to search task-specific architectures.

b) Comparison with the Efficient Model: Fast human pose distillation (FPD) [63] is an ideal efficient model to compare. Though it uses knowledge distillation technique rather than NAS, but to obtain the light-weight models is the same goal for both methods. Specifically, it first trains a large Teacher network, Houglass Network, to achieve a high accuracy performance on the task-dataset. Then a small Student model also taking Hourglass network as backbone is trained by knowledge distillation. We show the trade-off
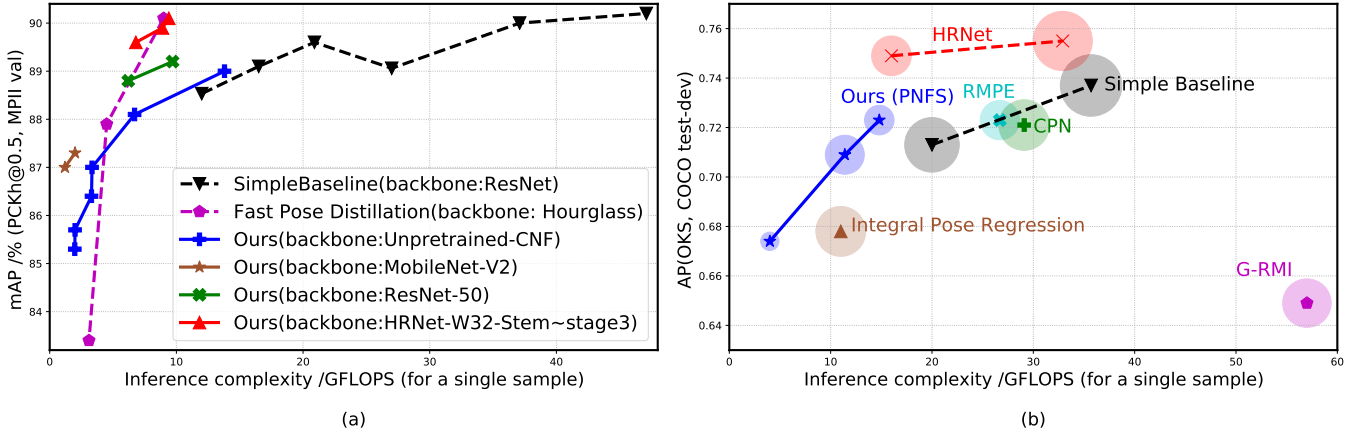
Fig. 7. **(a)**: The mAP of PCKh@0.5 metric vs. model inference complexity (GFLOPs) on MPII val set. **(b)**: The AP of OKS@0.5:0.95 c vs. model inference complexity (GFLOPs) on COCO test-dev2017 dataset. Model parameters and FLOPs of detecting persons in COCO dataset are not included. The areas of the circles are linearly relative with the amounts of models' parameters.

TABLE VII
COMPARISONS OF PERFORMANCE, MODEL PARAMETERS AND INFERENCE COMPLEXITY ON COCO TEST-DEV SET. MODEL PARAMETERS AND FLOPS OF DETECTING PERSONS ARE NOT INCLUDED. THE BACKBONES OF OURS-1, OURS-2 AND OURS-3 MODELS ARE MOBILENET-V2 (1.3M), RESNET-50 (23.5M) AND HRNET-W32-STEM~STAGE3 (8.1M).

	AP	AP^{50}	AP^{75}	AP^M	AP^L	AR	AR^{50}	AR^{75}	AR^M	AR^L	#Params	#FLOPs
CMU-Pose [14]	0.618	0.849	0.675	0.571	0.682	-	-	-	-	-	-	-
Mask-RCNN [29]	0.631	0.873	0.687	0.578	0.714	-	-	-	-	-	-	-
Associative Embedding [33]	0.655	0.868	0.723	0.606	0.726	0.702	0.895	0.760	0.646	0.78	-	-
Integral Pose Regression [64]	0.678	0.882	0.748	0.639	0.74	-	-	-	-	-	45.0M	11.0G
SJTU [16]	0.680	0.867	0.747	0.633	0.750	0.735	0.908	0.795	0.686	0.804	-	-
G-RMI [15]	0.685	0.871	0.755	0.658	0.733	0.733	0.901	0.795	0.681	0.804	42.6M	57.0G
PersonLab [49]	0.687	0.890	0.754	0.641	0.755	0.754	0.927	0.812	0.697	0.830	-	-
MultiPoseNet [32]	0.696	0.863	0.766	0.650	0.763	0.735	0.881	0.795	0.686	0.803	-	-
CPN [18]	0.721	0.914	0.800	0.687	0.772	0.785	0.951	0.853	0.742	0.843	-	-
SimpleBaseline(ResNet-50) [19]	0.702	0.909	0.783	0.671	0.759	0.758	-	-	-	-	34M	8.9G
SimpleBaseline(ResNet-152) [19]	0.737	0.919	0.811	0.703	0.800	0.790	-	-	-	-	68.6M	35.6G
HRNet-W32 [20]	0.749	0.925	0.828	0.713	0.809	0.801	-	-	-	-	28.5M	16.0G
Ours-1	0.674	0.890	0.737	0.633	0.743	0.731	0.928	0.791	0.681	0.800	6.1M	4.0G
Ours-2	0.709	0.904	0.777	0.667	0.782	0.766	0.941	0.829	0.715	0.836	27.5M	11.4G
Ours-3	0.723	0.909	0.795	0.684	0.792	0.779	0.945	0.844	0.731	0.845	15.8M	14.8G

curves between AP and FLOPs in the (a) of Fig. 7. The results show that both methods can achieve competitive performance with less computing complexity for MPII dataset. Our smaller models have a slight advantage over the smaller models of FPD.

Pruning Useless Structures. In practice, we find that some α, β architecture parameters of the final searched architectures might be zero values. That is, cells whose outputs are multiplied by the zero β parameter have no computing contributions for the cells in the next layers; operations whose outputs are multiplied by the zero α parameter have no computing contributions for the next hidden node. To further reduce the parameters and computing complexity, we use empty cells and zero operations without parameters to replace those useless cells and operations. By this way, the architectures only retain the structures associated with non-zero architecture parameters. This method does not affect the precision and also does not need to retrain the architecture. Note that our NAS method cannot ensure there are always existing zero

values in the micro or macro architecture parameters. When we cancel the learning rate decay of optimizing the architecture parameters so that they remain sensitive to be optimized in the late stage of the search process, we find that some architecture parameters converge to zero values, i.e. some types of operation or cells have no contributions to the forward computing in some searched architectures. We expect that future works can introduce sparse constraint to optimize the architecture parameters, to guarantee the architecture could be pruned by this way.

D. Comparison with the state-of-the-art

a) Testing on MPII Single Person Pose Estimation: To further evaluate our pose estimation method on test set of MPII [52], we train the model on all samples of MPII train set with early-stopping strategy. All input images are resized to 384×384 pixels, data augmentation is the same as mentioned above. We use pretrained MobileNet-v2 and HRNet-W32-Stem~stage3 feature blocks as backbone. The
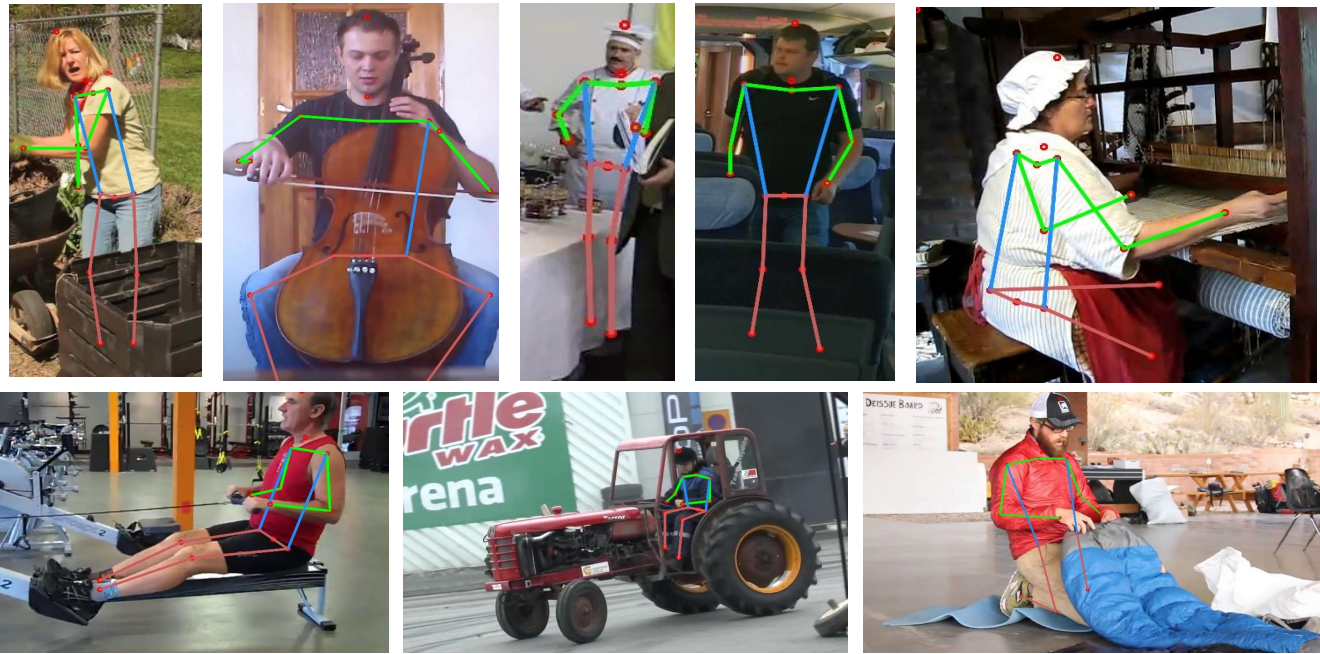
Fig. 8. Qualitative pose estimation results on MPII val set for single person pose estimation. We show the cropped image regions containing human body.

Fig. 9. Qualitative pose estimation results on COCO val2017 set. Estimation is conducted on bounding boxes detected by Faster-RCNN [65]. It is worth noting that our method works well in some heavily partial occluded hard samples (such as left two images in first row and the fourth in second row).

hyperparameters are: $P = 3/5, H = 2/1, d = 8/8, C = 10/16$ and $\mathcal{O}$ includes zero, skip connection, 3×3 separable conv, 3×3 dilated conv with 2 rate. The optimization method is synchronous optimization. As shown in Tab V, we achieve a comparable result (91.0 vs. 91.1 AP and 9.4G vs. 9.0G FLOPs) compared with FPD [63] that exploits knowledge distillation to achieve a lightweight and efficient model.

b) COCO Keypoint Detection Task: MS-COCO [53] dataset contains more than 200k images and 250k person instances with keypoints label. We use COCO train2017 as our training set, it consists of 57k images and 150k person instances. Val2017 set contains 5k images and test-dev2017 consists of 20k images. It is worth mentioning that some invisible keypoints are labeled on train set and statistics

show that around 11.3 % of annotated keypoints are invisible according to train2017 annotations. Object keypoint similarity (OKS) is the standard evaluation metric for keypoints locating accuracy. More detailed information is available in COCO official website [7].

COCO keypoint detection task involves detecting bodies and localizing their keypoints. Based on top-down method, we focus on single pose estimation, therefore we use the detected bounding boxes detected by Faster-RCNN [65] with 60.9 AP persons detection results on COCO test-dev2017 dataset. We respectively use pretrained MobileNet-v2 [57], ResNet-50 [58] and HRNet-W32-Stem~stage3 [20] feature blocks as

---

[7]http://cocodataset.org/

backbone and train two models only on train2017 set. The hyperparameters are: $P = 3/3/5, H = 1/1/1, d = 8/8/8$ and $\mathcal{O}$ includes zero, skip connection, 3×3 separable conv, 3×3 dilated conv with 2 rate. $C = 16/10/16$ for ours-1/ours-2/ours-3 models. The optimization method is synchronous optimization, we cancel the learning rate decay for searching the architecture parameters of ours-3 model, and prune it as described in SectionIV-C0b. The input size is 384×288 pixels and the OKS-NMS algorithm [15] is utilized to suppress redundant detected bounding boxes . We report average precision (AP) and average recall (AR) on COCO test-dev2017 set. With fewer parameters and low computational complexity, we can achieve comparable results with some state-of-the-art methods without using any extra data, ensemble models or other training tricks, 76.2 AP (gt bbox) and 73.0 AP (detected bbox) on COCO validation set and 72.3 AP on COCO test-dev2017 set, as shown in Tab VII. This model runs at ~ 20 FPS on a single GPU. The tendency of the trade-off curves between the average precision (AP) and computational cost (FLOPs) shown in the (b) of Fig. 7. can illustrate our model can achieve comparable performances with many other state-of-the-art methods [16], [18], [19]. Prediction results on some partial occluded hard samples can be seen in Fig. 9.

In fact, our searched models are still over-parameterized, the sizes of them also have a potential space to be further reduced by pruning and other model compressing techniques. Large-capacity models (e.g. >30M, >20GFLOPs) for COCO dataset are not searched, limited by its huge resource consumption caused by the gradient-based NAS search strategy. More advanced neural search strategies can be explored to overcome this problem in the future.

V. Conclusion

In this work, we presented a new paradigm - part-specific neural architecture search for human pose estimation, in which we made the first attempt to exploit prior knowledge of human body structure to search part-specific neural architectures automatically. Such paradigm develops the part-based method and gives a new example as Multi-task Neural Architecture Search. Experiment results showed that our light-weight models achieved comparable results on MPII dataset with fewer parameters and lower computational complexity than some state-of-the-art methods. For more challenging COCO keypoint detection task, our light-weight model attained comparable results to some state-of-the-art methods with fewer parameters. In addition, we empirically demonstrate the effectiveness of representing the human body keypoints as vector entities at image locations. We hope that these ideas may be helpful to adequately leverage NAS to practical application for human pose estimation task or other domains.

References

[1] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[2] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2017.

[3] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[4] Liang-Chieh Chen, Maxwell Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jon Shlens. Searching for efficient multi-scale architectures for dense image prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8699–8710, 2018.

[5] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8697–8710, 2018.

[6] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *arXiv preprint arXiv:1808.05377*, 2018.

[7] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. *arXiv preprint arXiv:1802.01548*, 2018.

[8] Saining Xie, Alexander Kirillov, Ross Girshick, and Kaiming He. Exploring randomly wired neural networks for image recognition. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[9] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. *arXiv preprint arXiv:1901.02985*, 2019.

[10] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018.

[11] Shoukang Hu, Sirui Xie, Hehui Zheng, Chunxiao Liu, Jianping Shi, Xunying Liu, and Dahua Lin. Dsnas: Direct neural architecture search without parameter retraining, 2020.

[12] Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4053–4061, 2016.

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

[14] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7291–7299, 2017.

[15] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4903–4911, 2017.

[16] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2334–2343, 2017.

[17] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1281–1290, 2017.

[18] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7103–7112, 2018.

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

[20] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[21] Wei Tang and Ying Wu. Does learning specific features for related parts help human pose estimation? In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[22] Rich Caruana. Multitask learning. In *Encyclopedia of Machine Learning and Data Mining*, 1998.

[23] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[24] Jason Liang, Elliot Meyerson, and Risto Miikkulainen. Evolutionary architecture search for deep multitask networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 466–473, 2018.

[25] Alejandro Newell, Lu Jiang, Chong Wang, Li-Jia Li, and Jia Deng. Feature partitioning for efficient multi-task architectures. In *arXiv preprint arXiv: 1908.04339*, 2019.

[26] Yuan Gao, Haoping Bai, Zequn Jie, Jiayi Ma, Kui Jia, and Wei Liu. Mtl-nas: Task-agnostic neural architecture search towards general-purpose multi-task learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11543–11552, 2020.

[27] Xiao Chu, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Structured feature learning for pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[28] Sijin Li, Zhi-Qiang Liu, and Antoni B Chan. Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 482–489, 2014.

[29] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017.

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

[31] Eldar Insafutdinov, Mykhaylo Andriluka, Leonid Pishchulin, Siyu Tang, Evgeny Levinkov, Bjoern Andres, and Bernt Schiele. Arttrack: Articulated multi-person tracking in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6457–6465, 2017.

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

[33] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2277–2287, 2017.

[34] Vasileios Belagiannis and Andrew Zisserman. Recurrent human pose estimation. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 468–475. IEEE, 2017.

[35] Wei Tang, Pei Yu, and Ying Wu. Deeply learned compositional models for human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 190–206, 2018.

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

[37] Elie Bienenstock, Stuart Geman, and Daniel Potter. Compositionality, mdl priors, and object recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 838–844, 1997.

[38] Pedro F Felzenszwalb, David A McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 7, 2008.

[39] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1014–1021. IEEE, 2009.

[40] Pedro F Felzenszwalb and Daniel P Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision (IJCV)*, 61(1):55–79, 2005.

[41] Seyoung Park, Bruce Xiaohan Nie, and Song-Chun Zhu. Attribute and-or grammar for joint parsing of human pose, parts and attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(7):1555–1569, 2017.

[42] Min Sun and Silvio Savarese. Articulated part-based model for joint object detection and pose estimation. In *2011 International Conference on Computer Vision (ICCV)*, pages 723–730. IEEE, 2011.

[43] Lin Zhao, Xinbo Gao, Dacheng Tao, and Xuelong Li. Tracking human pose using max-margin markov models. *IEEE Transactions on Image Processing (TIP)*, 24(12):5274–5287, Dec 2015.

[44] Xuecheng Nie, Jiashi Feng, Junliang Xing, Shengtao Xiao, and Shuicheng Yan. Hierarchical contextual refinement networks for human pose estimation. *IEEE Transactions on Image Processing (TIP)*, 28(2):924–936, Feb 2019.

[45] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 35(12):2878–2890, 2012.

[46] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1799–1807, 2014.

[47] Yu Chen, Chunhua Shen, Xiu-Shen Wei, Lingqiao Liu, and Jian Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1212–1221, 2017.

[48] Lipeng Ke, Ming-Ching Chang, Honggang Qi, and Siwei Lyu. Multi-scale structure-aware network for human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 713–728, 2018.

[49] George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, and Kevin Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[50] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with EM routing. In *International Conference on Learning Representations*, 2018.

[51] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3856–3866, 2017.

[52] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition (CVPR)*, pages 3686–3693, 2014.

[53] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV)*, pages 740–755. Springer, 2014.

[54] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research (JMLR)*, 13(Feb):281–305, 2012.

[55] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. *arXiv preprint arXiv:1902.07638*, 2019.

[56] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision (IJCV)*, 115(3):211–252, 2015.

[57] Mark B. Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018.

[58] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

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

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

[61] Adrian Bulat and Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 717–732. Springer, 2016.

[62] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 483–499. Springer, 2016.

[63] Feng Zhang, Xiatian Zhu, and Mao Ye. Fast human pose estimation. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3512–3521, 2019.

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

[65] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015.

APPENDIX A
EXPLANATION FOR THE RELATIONSHIP BETWEEN EXPECTED $\|\vec{\mathbf{v}}\|$ AND SUPERVISION LEVEL p

In the section III-D, we use Squash Function to normalize $\vec{\mathbf{v}}$ to $\vec{\mathbf{v}}_s$ whose length ranges in $[0, 1)$,

$$\vec{\mathbf{v}}_s = \frac{\|\vec{\mathbf{v}}\|^2}{1 + \|\vec{\mathbf{v}}\|^2} \frac{\vec{\mathbf{v}}}{\|\vec{\mathbf{v}}\|}, \tag{8}$$

$$\|\vec{\mathbf{v}}_s\| = \frac{\|\vec{\mathbf{v}}\|^2}{1 + \|\vec{\mathbf{v}}\|^2}, \tag{9}$$

$$\|\vec{\mathbf{v}}\| = \sqrt{\frac{1}{1 - \|\vec{\mathbf{v}}_s\|} - 1}, \tag{10}$$

where $\|\vec{\mathbf{v}}_s\|$ is supervised by numerical value p in each pixel position from groundtruth score maps. Ideal value of $\|\vec{\mathbf{v}}_s\|$, denoted as $\|\vec{\mathbf{v}}_*\|$, equals to $c \in [0, 1)$. Therefore, $\|\vec{\mathbf{v}}\|$ is supervised by $\sqrt{\frac{1}{1-c} - 1}$.

Intuition and Explanation. The extra advantage of vector in pixel is that ambiguity between image feature and groundtruth position can be reduced in some cases of occlusion. In supervised learning, the difficulty of fitting label is usually not under consideration, hard or easy samples of the same category receive the same level of supervision. This issue occurs in keypoints localization because image appearance varies dramatically in some partially occluded and non-occluded areas. Once the area within the keypoint groundtruth position is occluded, the image feature around the keypoint will be disturbed, (e.g. the first image in the Fig. 9, the man's ankle is occluded by a dog, but his ankle's position is labeled), as a result it becomes hard to force the network to predict high confidence to match the strong supervision. In such case, our method can handle it as $\|\vec{\mathbf{v}}_s\|$ replaces $\|\vec{\mathbf{v}}\|$ under supervision (element value of each dimension of vector has no explicit property and is unsupervised) and the expected length for $\vec{\mathbf{v}}$ in groundtruth keypoint pixel is not directly supervised by numerical value c from groundtruth score but supervised by $\sqrt{\frac{1}{1-c} - 1} \in [0, +\infty)$ where $c \in [0, 1)$. In a slight abuse of notation, we write $\|\vec{\mathbf{v}}_*\|$ as the expected length of $\vec{\mathbf{v}}$, which provides a relatively loose range space for $\vec{\mathbf{v}}$, even if under strong supervision.

APPENDIX B
MOBLIENET-V2 BACKBONE ARCHITECTURE DETAILS

A. CNF Backbone Architecture Details

B. MoblieNet-V2 Backbone Architecture Details

TABLE VIII
THE DETAILED CONFIGURATIONS FOR FABRIC-1,2,3. THE LAYERS RESERVED MEANS THE NUMBER OF LAYERS RESERVED BY DISCARDING THE IN THE LATTER LAYERS OF CNF

-	Fabric-1	Fabric-2	Fabric-3
L	7	8	8
C	10	10	12
H	2	1	1
Layers reserved	3	5	5
Number of Cells	9	17	17

TABLE IX
GIVEN A $H \times W \times 3$ RGB IMAGE, THE LAYER OF MOBILENET-V2 BACKBONE. EACH LINE DESCRIBES A SEQUENCE OF 1 OR MORE IDENTICAL (MODULO STRIDE) LAYERS, REPEATED N TIMES. ALL LAYERS IN THE SAME SEQUENCE HAVE THE SAME NUMBER C OF OUTPUT CHANNELS. THE FIRST LAYER OF EACH SEQUENCE HAS A STRIDE S AND ALL OTHERS USE STRIDE 1. THE EXPANSION FACTOR T IS ALWAYS APPLIED TO THE INPUT SIZE. P1~P4 ARE TAKEN AS THE FEATURE PYRAMIDS THAT ARE SEND TO EACH CNF. SEE MORE CONFIGURATIONS IN THE PAPER [57]

Input Size	Operator	t	c	n	s
$H \times W \times 3$	conv2d	-	32	1	2
$\frac{H}{2} \times \frac{W}{2} \times 32$	bottleneck	1	16	1	1
$\frac{H}{2} \times \frac{W}{2} \times 16$	bottleneck	6	24	2	2
$\frac{H}{4} \times \frac{W}{4} \times 24 \rightarrow$ P1	bottleneck	6	32	3	2
$\frac{H}{8} \times \frac{W}{8} \times 32$	bottleneck	6	64	4	2
$\frac{H}{8} \times \frac{W}{8} \times 64 \rightarrow$ P2	bottleneck	6	96	3	1
$\frac{H}{16} \times \frac{W}{16} \times 96 \rightarrow$ P3	bottleneck	6	160	3	2
$\frac{H}{32} \times \frac{W}{32} \times 160 \rightarrow$ P4	-	-	-	-	-