Cluster-wise learning network for multi-person pose estimation

Zhao, Ying
Luo, Zhiwei
Quan, Changqin
Liu, Dianchao
Wang, Gang

(Citation)
Pattern Recognition,98:107074

(Issue Date)
2020-02

(Resource Type)
journal article

(Version)
Accepted Manuscript

Cluster-wise Learning Network for Multi-person Pose Estimation

Ying Zhao[a,b,*], Zhiwei Luo[b], Changqin Quan[b], Dianchao Liu[a], Gang Wang[a]

[a] *Ricoh Software Research Center (Beijing) Co., Ltd. Beijing, China.*
[b] *Graduate School of System Informatics, Kobe University. Kobe, Japan.*

Abstract

In this paper, we propose a cluster-wise feature aggregation network that exploits multi-level contextual association for multi-person pose estimation. The recent popular approach for pose estimation is extracting the local maximum response from each detection heatmap that trained for a specific keypoint type. To exploit more contextual information, our network simultaneously learns complementary semantic information to encourage the detected keypoints subject to a certain contextual constraint. Specifically, our network uses dense and sparse branches to generate paired multi-peak detection heatmaps for clusters of keypoints. To enhance the feature passing through the network, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches. We demonstrate competitive performance of our network on the benchmark dataset for multi-person pose estimation.

Keywords: pose estimation, keypoint detection, deep learning

*Corresponding author
Email address: ying.zhao@srcb.ricoh.com (Ying Zhao)

1. Introduction

Human pose estimation refers to the task of locating and recognizing the body keypoints or joints (such as nose, eyes, ears, shoulders, elbows, wrists, hips, knees, ankles, etc.) from multiple persons in a single RGB image or a frame of videos. It is an essential component in many computer vision applications, such as human computer interaction system, somatosensory gaming, human action recognition, video surveillance and sports video analytics. Beside body joints, keypoints can be extended to refer to the small visual units with semantic information indicating the compositions, shapes and poses of the target objects, such as finger joints or key positions of any other objects. Therefore, accurate keypoint detection in unconstrained environments brings benefit to other more detailed visual understanding tasks, including semantic segmentation[1, 2, 3], saliency object segmentation[4, 5, 6], hand segmentation[7] and pose estimation [8, 9], viewpoint estimation [10, 11, 12, 13], salient object detection[14, 15, 16], attention prediction[17] and 3D reconstruction [18, 19, 20].

Similar as many computer vision tasks, the progress on human pose estimation problem is significantly improved by deep convolutional neural networks. The two major branches of human pose estimation networks are structured in top-down and bottom-up. On one hand, the top-down method [21, 22, 23, 24, 25, 26, 27] firstly detect persons and then repeatedly detect keypoints for each of them. On the other hand, the bottom-up methods [28, 29, 30, 31, 32] detect body keypoints without advance knowledge of person locations and then group them into person instances. Generally, most of the networks generate a likelihood detection heatmap for each keypoint and locates the keypoint as the point with the maximum likelihood in the detection heatmap. Moreover, the bottom-up networks need to allocate the detected keypoints to multiple scales and unknown number of persons.

To capture and consolidate information across all scales of the image, the pipeline of first encoding and then decoding is broadly used in both top-down and bottom-up approahces, such as the Stacked Hourglass Network[24] and As-

sociative Embedding (PoseAE)[31]. The backbone of these networks are based on multiple Hourglass modules[24] which firstly pool down the feature maps to a very low scale, and then upsamples and combines features across multiple resolutions. This architecture provides the benefit of capturing both global and local features for the pixel-wise prediction tasks. One commonality of these approaches is that they separately regress keypoint locations from each other according to their corresponding detection heatmaps. However, since body parts connect to each other, such as hip-knee-ankle, positions of some parts provide important contextual information and constraints on locating other related parts and the cues are hard to find if the keypoint representations are separated. This brings us an intuitive thinking of utilizing structural knowledge to enforce the model to exploit clues of related body parts to overcome ambiguous problem caused by occlusions and complex multi-person situations. Therefore, we propose to learn keypoints by clusters to exploit more contextual information. Our backbone structure for multi-scale feature extraction is same as the Stacked Hourglass Network's. However, the prediction heads are different. Our method utilizes multi-peak ground truth heatmaps to supervise the network exploiting contextual information between related body parts while the Stacked Hourglass Network uses a single-peak heatmap to present each keypoint separately.

Without prior assumption of person locations, the bottom-up methods have to estimate tags that guide the detected keypoints to form individual poses for person instances. For grouping the detected keypoints into instances, PoseAE[31] integrates associative embedding with the Stacked Hourglass Network[24], which produces a tagging heatmap for each detection heatmap. The idea of associative embedding is to predict an embedding in addition to the detection score for each keypoint. The embeddings serve as tags that group keypoints from same instances while separate keypoints from different instances. Following PoseAE[31], our network predicts tagging heatmaps in addition to detection heatmaps. Different from PoseAE[31], our tag embedding is applied cluster-wise. Since we divide keypoints into clusters, our tagging heatmaps contain multi-peak responses according to the keypoints in same cluster. Then the tags

are normalized within each keypoint cluster. Specifically, we represent each tag by the cluser-wise mean value with an offset factor to distinguish the keypoints by their relative orders. We supervise the cluster-wise embedded tags with a pull loss that encourages similar tags for clusters from same group and a push loss that enforces different tags for clusters across different groups.

In this paper, we propose a bottom-up method that learns keypoints by clusters with different configurations to exploit more contextual information. The keypoint cluster contains a group of keypoints to be learnt simultaneously according to the connection between body parts. Based on the number of related body parts, we define the dense and sparse cluster representations, such as divide the keypoints of head part into one cluster and the keypoints of limb part into another cluster. Comparing to the sparse cluster representation, the dense keypoint clusters reveal more global contextual information between more related parts. Therefore, our network is a multi-branch architecture that learns dense and sparse cluster of keypoints under the supervise of multi-peak heatmaps. This architecture imposes multi-level contextual relationships among multi-type abstractions of the keypoints.

To group the detected keypoints into instances, our network predicts tagging heatmaps in addition to the detection heatmaps. Therefore, each cluster branch consists of two sub-branches for keypoint detection and tag embedding. The pixels in the tagging heatmap serve to group keypoint clusters belong to the same instance. Besides that, the regions of person instances are simultaneously exposed in tagging heatmaps. Hence, our architecture can be taken as a multi-task learning, which can jointly handle the keypoint detection and coarse person segmentation problems. Despite the coarse boundaries of person regions generated by the tag embedding branch, it unveils holistic context information that benefits to keypoint detection. Based on this observation, we use an in-branch block to aggregate the tag features to the detection features in each branch to generate a region attention for keypoint detection. In addition, the detection features from dense and sparse branches complement each other. We design a cross-branch block to aggregate the sparse detection features to the

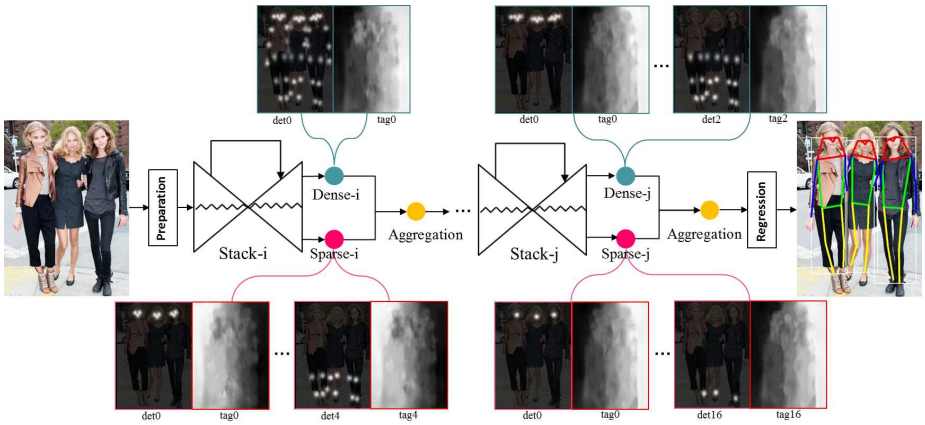


Figure 1: Overview of cluster-wise learning based keypoint detection. Our network learns keypoints by cluster. The two branches of the network simultaneously learn different clusters containing dense and sparse keypoint divisions. To take advantage of multi-level contextual information, each stack is set with different cluster configuration in both dense and sparse flows.

dense detection features to improve the flow of information between different branches. Thus, our network optimizes the features at a specific keypoint by the received information from diverse context.

Fig.1 illustrates overview of the proposed cluster-wise learning network. The backbone of our network consists of multi-stack Hourglass modules. Each stack module consists of branches of dense and sparse keypoint clusters. Specifically, the dense keypoint cluster contains more related keypoints than the sparse keypoint cluster,such as entire body parts versus only one body part in each heatmap. By doing this, the network learns global and local contextual information for each keypoint. For each branch, the network predicts multi-peak detection heatmaps that indicate keypoint locations. To retrieve instance identification, our network exploits tag embedding to predict instance tagging heatmaps that naturally serves as a rough instance segmentation. Therefore, before passing features to next stack, we aggregate the tagging heatmaps and detection heatmaps in and cross branches to enforce the network focus on global and local consistency of entire body and different parts. In summary, this paper

makes three main contributions:

- Cluster-wise keypoint detection. Instead of detect each keypoint separately, our network predicts multi-peak detection heatmaps for clusters of dense and sparse keypoints, which exploits global and local contextual information to improve the detection robustness.

- Feature aggregation. To enhance feature passing from shallow stack to deep stack, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches.

- Cluster-wise tag embedding. To better grouping the detected keypoints into instances, our network embedds relationships among the intra-cluster and inter-cluster keypoints with offset learning, which not only benefits the instance grouping but also individual keypoint identification.

2. Related work

Keypoint detection, human joint[21][27][28][29][30][31][32][35] and rigid object keypoint[13][36][37][38][39] detection in particular, is an important issue extensively discussed and researched in computer vision in recent years. This problem is firstly converted to a supervised regression of the x, y coordinates of the keypoints. Note that human pose estimation requires type information of each keypoint but object keypoint detection may not. DeepPose proposed by Toshev et al.[40] is the pioneer work in using the deep neural network to estimate 2D human pose. Since DeepPose[40], the pose estimation solutions thrive in the presence of deep neural network and branch out into top-down and bottom-up approaches.

By integrating person detection with single person pose estimation, the top-down approaches generates keypoints for all persons in the given image. Since

the pose estimator only have to focus on single instance, these two-stage methods usually produce fine localization. G-RMI by Papandreou et al.[41] predicts dense detection heatmaps and offsets for each keypoint type by using ResNet[42] and combined these outputs by an aggregation procedure to obtain highly localized keypoint predictions. In contrast to Gaussian detection heatmaps, the authors estimated a disk-shaped keypoint masks and 2-D offset vector fields to accurately localize keypoints. G-RMI[41] also uses a novel form of keypoint-based NonMaximum-Suppression (NMS), instead of the cruder box-level NMS, and a novel form of keypoint-based confidence score estimation, instead of box-level scoring. Xia et al.[2] ultilize a PoseFCN and a PartFCN to provide initial estimation of pose joint potential and semantic part potential. They declare that the estimated pose provides object-level shape prior to regularize part segments while the part-level segments constrain the variation of pose locations. RMPE by Fang et al.[22] integrates the Symmetric Spatial Transformer Network (SSTN), Parametric Pose NonMaximum-Suppression (NMS), and Pose-Guided Proposals Generator (PGPG) with stacked Hourglass modules[24] to handle inaccurate bounding boxes. MaskRCNN by He et al.[23] solves multi-task of object detection, instance segmentation and keypoint prediction together based on the RoI aligned feature maps extracted from ResNet[42] integrated with FPN[43]. CMUpose by Cao et al.[28] uses a nonparametric representation named as Part Affinity Fields to learn to associate body parts with individuals in the image. The architecture is designed to jointly learn part locations and their association via two branches of the same sequential prediction process.

Bottom-up multi-person pose estimation approaches firstly detect body parts instead of full persons and subsequently associate these parts to human instances. Methods of this fashion are similar in part detectors that generate detection heatmaps and differ in how to associate parts with each other and group into person instances. Compare to top-down approaches, they are faster in test time and smaller in model size. However, their accuracy always lower than the top-down methods since they are lack of prior knowledge of the number of people, their locations and detailed high resolution features. Deepcut

by Pishchulin et al.[35] proposes a partitioning and labeling formulation of a set of body-part hypotheses generated with CNN-based part detectors. Due to proposed clustering algorithm, Deepcut[35] is very slow and its processing time is in the order of hours even though it doesn't use person detections. Following Deepcut[35], Deepercut by Insafutdinov et al.[29] benefits from deeper ResNet[42] architectures as part detectors and improves the parsing efficiency of Deepcut[35] with an incremental optimization strategy. Iqbal et al. [30] also formulate the problem as part grouping and labeling via a linear programming. Different from Deepcut[35] and Deepercut[29], Iqbal et al. [30] propose to solve the densely connected graphical model locally to improve time efficiency significantly. Following CPM[26], CMUPose by Cao et al.[28] uses a part affinity field learn to associate body parts and group them to person instances with greedy bottom-up parsing steps. PoseAE[31] proposes associative embedding to simultaneously generate and group detections. Chu et al.[21] use geometrical transform kernels and a bi-direction tree structured model to capture and pass relationships between joints. PersonLab[32] proposes a part-induced geometric embedding descriptor to associate semantic person pixels with their corresponding person instance. MultiPoseNet[44] receives keypoint and person detections, and produces accurate poses by assigning keypoints to person instances.

Among the most dominant approaches to pose estimation from videos is a two-stage approach, which first deploys a frame-level keypoint estimator, and then connects these keypoints in space and time using optimization techniques. For frame-level keypoint estimation, Detect-and-track by Girdhar et al.[45] experiment with Mask R-CNN[23], as well as their proposed 3D extension of this model, which leverages temporal information over small clips to generate more robust frame predictions. SimpleBaseline by Xiao et al.[46] builds a frame-level keypoint detector by simply adding a few deconvlutional layers over the last convolution stage in the residual network(ResNet) by He et al.[42]. The one-stage approach Poseflow by Xiu et al.[47] proposes a functionally structured spatial-temporal deep network, PoseFlow Net(PFN), to jointly solve the skeleton localization and matching problems of PoseFlow. Its spatial derivative reasoning

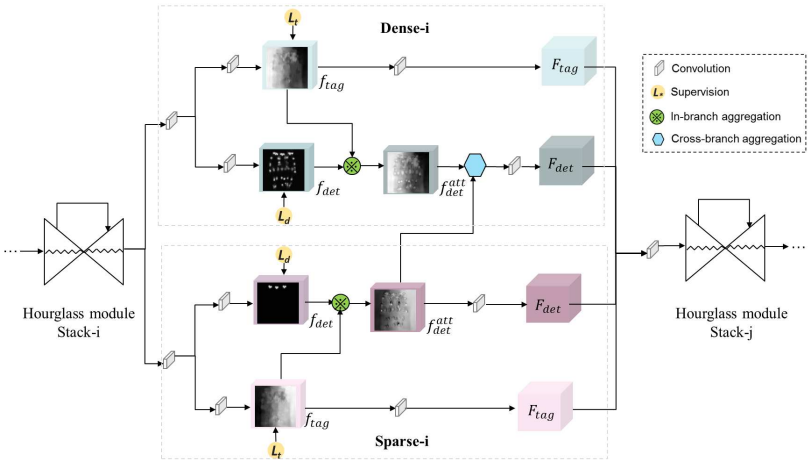


Figure 2: Illustration of our cluster-wise learning network. The backbone of our network consists of multi-stack Hourglass modules. Each Hourglass module is followed by two branches for predicting keypoints based on dense and sparse clusters. For each branch, the network predicts detection heatmaps to indicate keypoint positions and tagging heatmaps to group the detected keypoints into instances. Additionally, the detection heatmaps and tagging heatmaps are fused to enhanced feature maps by in-branch and cross-branch aggregation. All feature maps are fused together by 1x1 convolutions and passed to next Hourglass module. The details of feature aggregation are shown in Figure 4.

branch is an encoder-decoder(convolution,pooling-deconvolution,upsampling) network architecture with input as the fth video frame.

Inspired by recent works, we propose a network simultaneously detecting and grouping keypoints with the help of multi-level contextual exploration. Overall, two branches of our network learns keypoints by clusters. The cluster structures are set in different keypoint densities. The denser the cluster, the higher the global information capacity. To group instance while retrieve individual keypoint identification, our network learns offset embedding. To enhance feature passing from stack to stack, the network aggregates features in and cross branches. The in-branch aggregation enhances keypoint detection features by fusing them with holistic instance region attention maps in each branch. The cross-branch aggregation further strengthens the keypoint detection feature by fusing global and local context information between dense and sparse branches.

3. Methodology

3.1. Network overview

In this section, we introduce a multi-person pose estimation network consists of feature extraction backbone, keypoint detection and tag embedding. As illustrate in Figure 2, the network stacks multiple Hourglass modules[24] to extracts multi-scale features by repeating top-down and bottom-up information passing. The Hourglass module[24] has the symmetric topology that consists of a series of convolution, pooling and upsampling layers. This architecture provides the benefit of capturing both global and local features for the pixel-wise prediction tasks. We stack multiple Hourglass modules under supervision of multi-dense keypoint clusters to make the network gradually strengthen the capacity of accurate prediction. Figure 3 illustrates predictions of each stack module. To consider all keypoints simultaneously and learn configurations of keypoints, each stack outputs two branches of keypoint clusters. Different cluster conveys different semantic information and are complementary to each other. This enforces the module to have different localization abilities. To enhance features passing from stack to stack, the network aggregate information from different branches. As shown in Figure 5b, the tag value obviously changes across instances that makes the tapmaps to be a rough instance segmentation. Therefore, the combination of detection heatmaps and tagging heatmaps serves as a bootstrap for passing semantic enhanced features from stack to stack. The module embeds multiple relation of the keypoints and provides instance information for grouping the keypoints and individual identification. The design of these modules is motivated by the need to capture contextual correlation of keypoints and use that to produce accurate detection. Under supervisions of variant ground truth, these consecutive modules gradually transform the features into detection and tag heatmaps that respectively indicate keypoint locations and groups. More specifically, the network contains multiple stacks which consists of sparse and dense branches. The predictions of all branches in all stacks are used for training. The sparse branch of the last stack is used for inference.

To detect keypoints of an object, a popular way is detecting each of them as an individual class. This kind of independent detection can produce accurate localizations for the keypoints having distinctive appearances. However, they often fail to output desirable results for the keypoints with ambiguous appearances. Since spatial distributions and semantic meaning of adjacent keypoints are highly correlated, context information should be learnt together with the appearance features. Thus, we take a cluster of keypoints as an entity of the network rather than each individual keypoint. Our network consists of branches of dense and sparse keypoint clusters. The denser the cluster, the higher the global information capacity. More specifically, the dense keypoint cluster contains keypoints of entire body while the sparse keypoint clusters are consists of part of them. By doing this, the network learns global and local contextual information for each keypoint. For each branch, the network predicts multi-peak detection heatmaps that indicate keypoint locations.

By supervising multiple keypoint detectors built on hierarchical features with multi-level semantic supervisors, our network fully explores the diversities of keypoint contextual structures. Then, the detection heatmaps produced by each branch will be unified to produce the final keypoint locations. So far, we only know where are the keypoints but have no idea about how do they relate to each other and which instances do they belong to. In addition to producing the detection heatmaps, the network correspondingly outputs tagging heatmaps for keypoint-wise relation embedding and instance retrieving. In the tagging heatmaps, pixels having similar values indicate the locations from the same instance. Finally, we use a loss function that enforces the keypoints detection close to the ground truth and encourages pairs of tags to have similar values if the corresponding detections belong to the same instance or different values otherwise.

3.2. Cluster-wise keypoint detection

In most recent keypoint detection network, ordered multi-channel detection heatmaps are widely used for representing keypoints. The number of channels

is fixed by the object category, e.g. 17-channel detection heatmaps for human data from MSCOCO[48] dataset. Each channel detection heatmap associates one category specific keypoint and is learnt independently with the others. The relation and restriction among keypoints are not being fully explored and exploited. In addition, although these representations are compatible with invisible keypoints, it is difficult to learn categories with varying number of keypoints together. One brute force approach is to stack all detection heatmaps from all categories and certainly results in high computational costs.

In this paper, we propose a novel keypoint representation which encodes multi-level relation of keypoints while provides flexibility to represent varying numbers of keypoints across different categories. Our representation consists of two level entities. One is the dense keypoint cluster that contains global information by using more keypoints in a cluster, such as all keypoints of the entire body. The other is the sparse keypoint cluster that are consists of partial keypoints to be learnt simultaneously. To extract multiple levels of relations, we stack multiple single stage modules. The dense and sparse cluster configurations can be varied from stack to stack and the detection heatmaps from dense and sparse flow are fused and passed to next stack. As shown in Figure 3, we predict C_d^i and C_s^i detection heatmaps for the dense and sparse keypoint cluster, where C^i is the number of keypoint clusters in ith stack, such as $C_d^0 = 1, C_s^0 = 5$ in the 0-th stack. Local peak values in each detection heatmap indicate the most probably locations for the entities from multiple instances of the same specific cluster. The StarMap[13] proposes a category-agnostic keypoint representation, which combines all keypoints of one object together and forms a multi-peak detection heatmap (called StarMap). This is in contrast to our method which uses multi-peak detection heatmaps to detect multiple instances of the same specific cluster of keypoints. We divide the keypoints into clusters based on the structure of body joints connection, such as divide the keypoints of head part into one cluster and the keypoints of limb part into another cluster.

Given a set of ground truth keypoint locations, we divide them into C_d^i and C_s^i groups and generate a set of ground truth detection heatmaps for train-

ing the dense and sparse keypoint clusters. More specifically, the input labels for supervision are the keypoint coordinates which are converted to ground truth heatmaps by using Gaussian kernel. Thus, each pixel in the ground truth heatmaps represents the probability of location being the keypoint. The prediction penalty is increased along the radius with the amount given by a 2D Gaussian kernel. To detect the keypoint locations, we enforce the prediction loss that encourages the prediction detection heatmaps having consistent local peaks with the ground truth. The topK local peaks will be selected as the candidate keypoints for each channel. The K indicates the number of visible keypoints based on the maximum number of people can be detected and the cluster setting. In our experiments, we set K as $30c_i$ where the maximum number of people is 30 and the c_i is the number of keypoints in the ith cluster. Specifically, this loss is the mean Euclidean distance over all pairs of prediction and ground truth detection heatmap. For each keypoint cluster, the loss of location prediction is defined as

$$L_{det}^{B} = \frac{1}{M} \sum_{i} (p_i - g_i)^2 \tag{1}$$

Where, B indicates branch of dense or sparse cluster, M is the number of pixels in the detection heatmap, p_i and g_i are pixel values at position i in the detection heatmap and ground truth heatmap respectively.

3.3. Cluster-wise tag embedding

Following Newell et al.[31], we group detected keypoints into instances by tag embedding. For each detection heatmap, the network predicts a tagging heatmap which is responsible for guiding the corresponding keypoint to group with other keypoints. The difference is that our tag embedding is applied in a cluster-wise way. Similar as detection heatmaps, our tagging heatmaps contain multi-peak responses since we divide keypoints into clusters. The tag values are normalized within each tagging heatmap. Specifically, we represent each tag by the cluser-wise mean value with an offset factor to distinguish the keypoints by their relative orders. By doing this, we can refine the detected position by

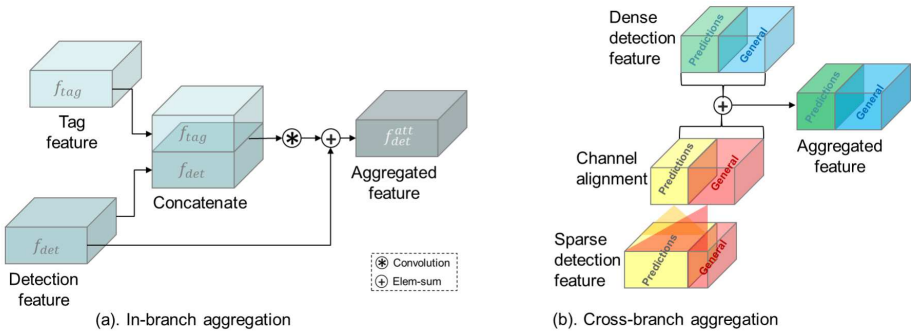


Figure 3: Illustration of keypoint(left) and tagging(right) heatmaps of each stack module. The network consists of four stack modules with dense(a,c,e,g) and sparse(b,d,f,h) cluster settings (1,1,1,3) and (5,5,5,17). For simplicity, we only display a part of each cluster.

checking whether the difference between two tags satisfies the offset setting. The generation of tagging heatmaps does not need specific ground truth labels for supervision since the clusters can be grouped into instances based on the differences between the tag values. Therefore, we supervise the cluster-wise embedded tags with a pull loss that encourages similar tags for clusters from same instance and a push loss that enforces different tags for clusters across different instances. Figure 3 visualizes the tagging heatmaps.

In general, the tagging heatmaps for Bth branch are trained with embedding loss.

$$L_{emb}^{B} = L_{pull} + L_{push} \tag{2}$$

The pull loss L_{pull} is defined as the squared distance between the normalized embedding and the reference embedding of each keypoint.

$$L_{pull} = \frac{1}{NJ} \sum_{n} \sum_{k} (\hat{h_k} - \overline{h_n})^2 \tag{3}$$

Where, N is the number of instances, J is the total number of visible keypoints and assuming all instances have J visible keypoints, $\hat{h_k}$ is the normalized embedding for the kth keypoint, $\overline{h_n}$ is the reference embeddings for the nth instance.

The normalized embedding is calculated as the cluster-wise mean value $\tilde{h_k}$

with the offset that encodes the keypoint index d_k in the cluster, such as $d_k = 0$ for the first keypoint.

$$\hat{h_k} = \tilde{h_k} + d_k * \delta \tag{4}$$

$$\tilde{h_k} = \frac{1}{c_i} \sum_k h_k \tag{5}$$

Where, $d_k \in [0, max(c_i)]$ and c_i is the number of visible keypoints in ith cluster, δ is the base factor for the offset, h_k is the predicted tagging heatmap for the kth keypoint.

The push loss L_{push} is computed between reference embeddings of different instances with a penalty that drops exponentially to zero as the increase of embedding difference.

$$L_{push} = \frac{1}{N(N-1)} \sum_n \sum_{m \neq n} exp(-\frac{1}{2\Delta^2}(\overline{h_n} - \overline{h_m})^2) \tag{6}$$

$$\overline{h_n} = \frac{1}{J} \sum_k \tilde{h_k} \tag{7}$$

Where, $\overline{h_n}$ and $\overline{h_m}$ are the reference embeddings for different instances, Δ leads a margin for distance between each pair of reference embeddings. Considering inter-instance difference should be much larger than intra-instance difference, we assign the margin Δ based on the offset $d_k\delta$ to far apart reference embeddings from each other. Then, the embedding values are separated both in clusters and instances and benefitted to individual keypoint retrieving. We set (Δ,δ) as (4, 0.03) in our experiments.

Given the predicted keypoints and tagging heatmaps, we group the keypoints across channels by retrieving the tag values at keypoint locations from corresponding channels. To decode the output of the network, the keypoints having similar tag values are matched up to form instances. The full training loss is

$$L = \sum_B (\lambda_d L_{det}^B + \lambda_e L_{emb}^B) \tag{8}$$

Where λ_d and λ_e are the weights for the detection and embedding loss respectively. We set (λ_d, λ_e) as (1, 1e-3) in our experiments.

3.4. Feature aggregation

For each detected keypoint, we have to estimate a tag to indicate which person instance it belongs to. Thus, we separate each cluster branch into two sub-branches for keypoint detection and tag embedding. Then, the intra and inter relation of keypoint clusters are embedded to tagging heatmaps. The pixels in the tagging heatmap serve to group keypoint clusters belong to the same instance. Besides that, the regions of person instances are simultaneously exposed, as shown in Figure 5. Hence, our architecture can be taken as a multi-task learning, which can jointly handle the keypoint detection and coarse person segmentation problems. Despite the coarse boundaries of person regions generated by the tag embedding branch, it unveils holistic context information that benefits to keypoint detection. Moreover, the stacking strategy of Hourglass modules can be taken as refinement process. Multi-scale features in shallow stack are aggregated into high-level features to be further exploited by deep stack. To enhance the features passing from shallow stack to deep stack, we aggregate information from different branches. More specifically, we enrich the features to be passed to next stack by aggregating tag to detection in-branch and sparse to dense detections cross-branch. We use an in-branch block to aggregate the tag features to the detection features in each branch to generate a region attention for keypoint detection. In addition, the detection features from dense and sparse branches complement each other. We design a cross-branch block to aggregate the sparse detection features to the dense detection features to improve the flow of information between different branches. Thus, our network optimizes the features at a specific keypoint by the received information from diverse context.

The in-branch aggregation targets to enhance keypoint detection features by fusing holistic human region attention maps that are byproducts of tag embedding. The tag embedding generates tagging heatmaps by pulling together the keypoints from same instance while pushing away those from different instances. The supervision is realized through comparison of relative tag values between keypoints but not all pixels in the map. Then, the keypoints can be grouped

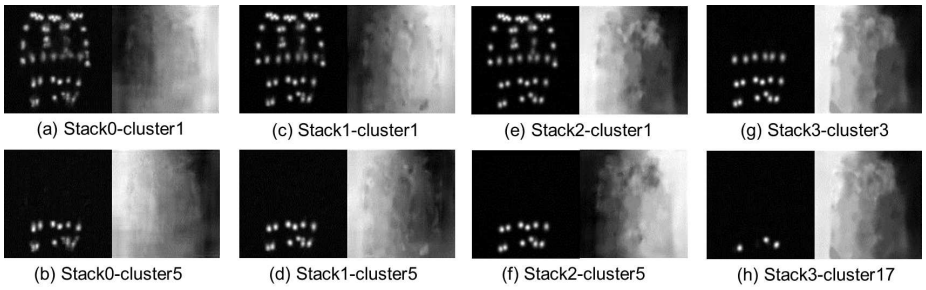


Figure 4: Demonstration of feature aggregation. In-branch aggregation combines tag and detection features in each branch and cross-branch aggregation fuses detection features from dense and sparse branches.

into instances based on the differences between the tag values. Moreover, the non-keypoint pixels obtain tag values through convolution propagation, and aggregate into body regions around keypoints and eventually form human regions. Therefore, tagging heatmaps also provide crucial clues of semantic regions for keypoint detection. Based on this observation, we aggregate tag features to detection features, as demonstrated in Figure 4a. The tag and detection features are firstly concat-enated and then fused by a 1x1 convolution layer to generate merged features. Then, the residual connection adds the identity mapping of detection feature to the merged features. Figure 5c visualizes some examples of in-branch aggregation.

The cross-branch aggregation focuses on fusing global and local context information between dense and sparse branches for keypoint detection. To balance features in different branches and make the network easier to extend, the number of channels are same in dense and sparse branches. The feature maps of each branch are composed of the supervised predictions and the unsupervised general features. In addition, the proportions of these two components in dense and sparse branches are different. The dims of predictions correspond to the number of supervisions while the extra features are the rest channels of total amount. Therefore, we apply different aggregation strategy to these two kinds of feature maps, as illustrated in Figure 4b. The numbers of channels are dif-

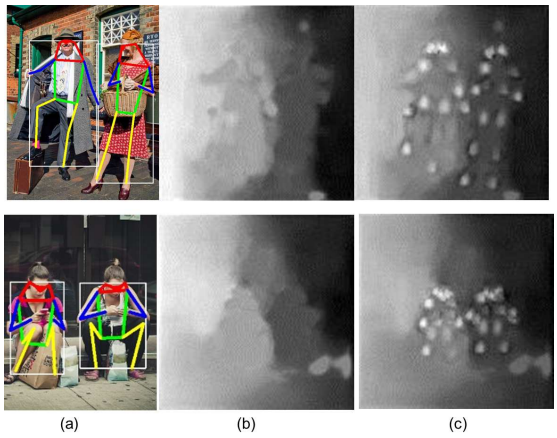


Figure 5: Visualization of tag embeddings of dense keypoint cluster. Columns from left to right are pose predictions overlapping on inputs(cropped for simplicity illustration), tagging heatmaps and combination of detection heatmaps and tagging heatmaps. Note that, the tag value obviously changes across instances.

ferent in two branches due to the dense and sparse cluster settings. Therefore, we firstly unify the numbers of channels of two branches before feature aggregation. We use group convolution to merge the feature maps of sparse branch and the group setting is according to the dense branch, such as the feature maps of keypoints in head part are merged together. Thus, we align the channels of sparse branch to dense branch based on their semantic meaning. Considering parameters and computation burden, we use 1x1 kernels to do the channel-wise fusion. Through 1x1 kernels, the prediction features are mapped from sparse to dense and the general features are enriched by aggregating the supervised information. We aggregate the aligned features from sparse branch to dense branch by element-wise sum.

4. Results and discussion

4.1. Evaluation on benchmark dataset

We train and evaluate our network on the MSCOCO-Keypoint2017[48] dataset which consists of 118,287 training, 5000 validation and 40,670 testing images (in-

cluding 20,880 test-dev and 19,790 test-challenge). More specifically, over 150K labeled person instances from 56,599 training and 2346 validation images are publicly available. The annotation for each people is a sequence of 17 keypoints with a certain order. Therefore, we can retrieve the type of each keypoint from a predefined dictionary (e.g., the second keypoint is the left eye). We trained our network under the supervising of the ground truth heatmaps generated from the keypoint annotations with Gaussian. Following state-of-the-art, our evaluation is done using the Object Keypoint Similarity (OKS) metrics provided by MSCOCO online evaluation platform [34]. The OKS measures the similarity between ground truth keypoints and predicted keypoints based on distance rather than the bounding box overlap. Given the OKS, we can compute average precision (AP) and average recall (AR) of the keypoint detection just as the IoU allows us to compute these metrics for object detection. The OKS is defined as

$$OKS = \sum_i [exp(-d^2/2s^2{k_i}^2\delta(v_i))]/\sum_i [\delta(v_i > 0)] \tag{9}$$

Where, d_i is the Euclidean distance between detected keypoint and ground truth, the v_i indicates visibility of the ground truth, sk_i is the standard deviation of an unnormalized Guassian, where s is the object scale and k_i is a per-keypiont constant that controls falloff. For each keypoint, this yields a keypoint similarity that ranges between 0 and 1.

4.2. Implementation

Our network is implemented under the PyTorch framework and trained and tested on two GTX 1080 GPUs. The test experiments are also developed on the same environment. We tried on combinations of different randomly initialization and found out that the training works well under the default setting of PyTorch. During training, we set the input resolution of the network to 512x512, which leads to an output resolution of 128x128. To reduce overfitting, we adopt standard data augmentation techniques including random horizontal flipping, scaling, cropping and adjusting the brightness, saturation and contrast of an

image. We train the network using a batch size of 8 for 200 epochs with an initial learning rate of 2e-4 (dropped to 1e-5 after 150 epochs). We find that larger values of learning rate lead to diverging gradient and kill some neural units. The forward passing is terminated by a 1x1 convolution and output detection heatmaps corresponding to each cluster. We use Adam to optimize the full training loss L defined in Eq(8). The loss minimizes the differences between the predicted and the ground truth locations and groupings. The relation embedding term is weighted by a factor of 1e-3 relative to the keypoint detection term. During each round of iterations, 1000 training steps and 10 evaluation steps are alternatively carried on. The network performance is evaluated and samples of each batch are randomly shuffled. Our pipeline needs totally around 0.229 seconds for generating the final detection heatmaps and keypoints converting.

4.3. Performance

To get an intuitive sense of the network's predictions, we firstly visualize the final keypoint locations of our system extracted from validation dataset. Figure 6 illustrates the typical cases of occlusion, crowding, deformation and low resolution. We see that our system has superior and robust performance on self-occluded joints with variant deformation of human poses. Our system also makes better balance between false detection and miss detection for the intractable cases caused by dense mutual occlusions and low resolutions. This is quite encouraging since our approach is designed to be a general purpose keypoint predictor. The quantitative evaluation for keypoint detection on MSCOCO is presented in Table 1. The primary challenge metric AP and AR are averaged over multiple OKS values with 0.05 interval. The AP^{50} means AP at $OKS = 0.50$. Small objects (segment area $< 32^2$) do not contain keypoint annotations. The AP^M is for the medium objects having areas between of 32^2 and 96^2 and the AP^L is for large objects having area larger than 96^2.

We test the effectiveness of cluster-wise learning by testing the network under different clustering configurations. The cluster configuration (1,1,1,1) means that the 17 keypoints are divided into 1 cluster and will be learnt simultane-

Table 1: Comparison results on MSCOCO testdev dataset with OKS.

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
CMUPose[28]	0.618	0.849	0.675	0.571	0.682	0.665	0.872	0.718	0.606	0.746
RMPE[22]	0.618	0.837	**0.698**	0.586	0.673	0.676	0.875	0.746	0.630	0.740
MaskRCNN[23]	0.631	**0.873**	0.687	0.578	**0.714**	-	-	-	-	-
G-RMI[41]	**0.649**	0.855	0.713	**0.623**	0.700	0.679	**0.887**	**0.755**	**0.644**	0.771
PoseAE[31](baseline)	0.633	0.857	0.689	0.580	0.704	**0.688**	0.884	0.742	0.620	**0.781**
Cluster1	0.617	0.852	0.675	0.564	0.691	0.682	0.894	0.735	0.617	0.769
Cluster2	0.620	0.852	0.677	0.564	0.698	0.682	0.891	0.734	0.615	0.773
In-branch	0.625	0.855	0.682	0.570	**0.703**	0.687	**0.897**	0.739	0.621	**0.777**
Cross-branch	0.625	0.854	0.681	0.571	0.701	0.687	0.895	0.739	0.622	0.776
In-Cross-branch	**0.627**	**0.857**	**0.687**	**0.572**	0.702	**0.688**	0.896	**0.742**	**0.623**	**0.777**

ously in each stack. Similarly, the 17 keypoints are evenly divided into 3 clusters and 5 clusters with (5,6,6) and (3,3,3,4,4) points in each cluster. Thus, for experiment Cluster-1, the dense cluster configuration is (1,1,1,3) and the sparse cluster configuration is (5,5,5,17). And for experiment Cluster-2,the dense cluster configuration is (1,1,1,1) and the sparse cluster configuration is (17,17,17,17). Following prior arts, we use four stacks of corresponding networks for each experiment. Based on Cluster-2, we further study the benefits of feature aggregation by training and testing the networks with/without In-branch and Cross-branch aggregation blocks. Overall, the In-Cross branch achieves our best result by enforcing the network learns more complex contextual information and semantic consistency for keypoint detection. This result indicates that the more global information encoded, the finer the detection result.

In Table 1, we compare our network with other state-of-the-art detectors based on results from their public implementation. The result of PoseAE[31] is our baseline. Comparing with the baseline, our method has better performance at AR^{50} and AR^{M} and comparable performance at most other metrics, like AP^{50}, AP^{75}, AP^{L}, AR and AR^{75}. The overall performance of our network

Table 2: Comparison results on PoseTrack validation dataset with mAP.

Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
DetectTrack[45]	67.5	70.2	62.0	51.7	60.7	58.7	49.8	60.6
PoseFlow[47]	66.7	73.3	68.3	61.1	67.5	67.0	61.3	66.5
SimpleBaseline[46]	81.7	**83.4**	**80.0**	**72.4**	**75.3**	**74.8**	**67.1**	**76.7**
In-Cross-branch	**83.4**	80.9	66.4	56.0	72.5	64.8	52.0	70.3

is higher than the state-of-the-art CMUpose[28] and RMPE[22] and reaches a competitive result over the other recent methods. Note that our recalls (AR^{50}, AR^{50} and AR^{L}) are higher than most of the compared methods. The dense branch encodes global consistency information by using more keypoints in a cluster, such as all keypoints of the entire body. By using dense branch, our network explores more keypoints which are difficult to be found.

4.4. Application on pose tracking

The PoseTrack[49] dataset is a large-scale benchmark for human pose estimation and tracking in video and based on the diverse real-world activity videos from the MPII Human Pose dataset[50]. It annotates 15 body keypoints from 550 video sequences (292 training, 50 validation and 208 testing) with 66, 374 frames. The length of the most videos ranges between 41 and 151 frames, and 30 frames from the middle of the sequence have annotations. The validation and test sequences are densely annotated with a step of four frames. In total, PoseTrack2018 provides around 23,000 labeled frames and 153,615 pose annotations. Besides the difference in keypoint types from one to four, the annotation format is compatible with MSCOCO dataset. Following [49], we use the PCKh metric to estimate the correctness of predicted keypoint location. PCK is short for Probability of Correct Keypoint and defined a candidate keypoint to be correct if it falls within $\alpha max(h, w)$ pixels of the ground truth keypoint, where h and w are the height and width of the person bounding box respectively, and α controls the relative threshold for considering correctness. PCKh defines the

Figure 6: Visualization of our results on MSCOCO validation dataset.

matching threshold as 50% of the head segment length where the head length corresponds to 60% of the diagonal length of the ground truth head bounding box. For measuring frame-wise multi-person pose accuracy, we use mean Average Precision (mAP) as [49].

We retrain our network for multi-person pose estimation on the PoseTrack2018 training set and we use the model pre-trained on MSCOCO dataset to initialize the network. Since it is a fine-tuning, the training iteration converges within 20 epochs and the base learning rate 1e-5 is changed to 1e-6 after 15 epochs. The other training configures are the same as that for MSCOCO. We use the PoseTrack evaluation toolkit for results on validation set and report results on test set from the evaluation server. Since our network is designed for single-frame pose estimation, we estimate our frame-wise prediction results on validation set. Figure 7 and Table 2 illustrate results of our approach on PoseTrack validation dataset. Overall, our network performs better in exploring keypoints of head region and competitive in detecting keypoints of shoulders and hips. Figure 8 shows some failure cases of our method on MSCOCO and PoseTrack datasets.

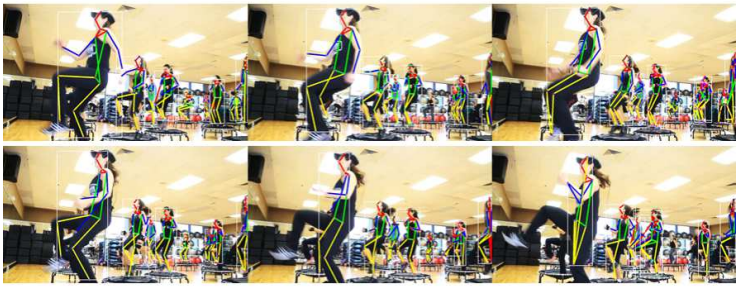
5. Conclusion

In this paper, we propose the idea of exploring global and local contextual relations from dense and sparse cluster-wise multi-peak detection heatmaps for keypoint detection. To enhance repeated bottom-up and top-down feature passing, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches. Meanwhile, the intra-cluster and inter-cluster relationships are embedded with tag learning to guide the instance grouping and individual keypoint identification.

In the future work, further improvement is expected by integrating with more advanced existing techniques of flownet[51] tracking, relation extraction[52] and

Figure 7: Visualization of our results on PoseTrack test dataset.

(a) Results on COCO validation dataset.



(b) Results on PoseTrack test dataset.

Figure 8: Illustration of failure cases of our method.

information passing[53]. Moreover, we will exploit a compact architecture for multi-scale feature extraction to reduce the model size. In addition, more extensive experiments will be carried on to test the possibility that network can detect variant number of keypoints for novel categories.

References

[1] Z. Wu, G. Lin, J. Cai, Keypoint based weakly supervised human parsing, CoRR abs/1809.05285.

[2] F. Xia, P. Wang, X. Chen, A. L. Yuille, Joint multi-person pose estimation and semantic part segmentation, in: CVPR, IEEE Computer Society, 2017, pp. 6080–6089.

[3] Y. Yang, D. Ramanan, Articulated human detection with flexible mixtures of parts, IEEE Trans. Pattern Anal. Mach. Intell. 35 (12) (2013) 2878–2890.

[4] W. Wang, J. Shen, F. Porikli, R. Yang, Semi-supervised video object segmentation with super-trajectories, IEEE Trans. Pattern Anal. Mach. Intell. 41 (4) (2019) 985–998.

[5] W. Wang, J. Shen, F. Porikli, R. Yang, Semi-supervised video object segmentation with super-trajectories, IEEE Trans. Pattern Anal. Mach. Intell. 41 (4) (2019) 985–998.

[6] W. Wang, J. Shen, L. Shao, Video salient object detection via fully convolutional networks, IEEE Trans. Image Processing 27 (1) (2018) 38–49.

[7] Y. Zhao, Z. Luo, C. Quan, Coarse-to-fine online learning for hand segmentation in egocentric video, EURASIP J. Image and Video Processing 2018 (2018) 20.

[8] S. Yuan, G. Garcia-Hernando, B. Stenger, G. Moon, J. Y. Chang, K. M. Lee, P. Molchanov, J. Kautz, S. Honari, L. Ge, J. Yuan, X. Chen, G. Wang, F. Yang, K. Akiyama, Y. Wu, Q. Wan, M. Madadi, S. Escalera, S. Li, D. Lee, I. Oikonomidis, A. A. Argyros, T. Kim, 3d hand pose estimation: From current achievements to future goals, CoRR abs/1712.03917.

[9] X. Zhou, Q. Wan, W. Zhang, X. Xue, Y. Wei, Model-based deep hand pose estimation, in: IJCAI, IJCAI/AAAI Press, 2016, pp. 2421–2427.

[10] H. Su, C. R. Qi, Y. Li, L. J. Guibas, Render for CNN: viewpoint estimation in images using cnns trained with rendered 3d model views, in: ICCV, IEEE Computer Society, 2015, pp. 2686–2694.

[11] S. Tulsiani, J. Malik, Viewpoints and keypoints, in: CVPR, IEEE Computer Society, 2015, pp. 1510–1519.

[12] S. Tulsiani, T. Zhou, A. A. Efros, J. Malik, Multi-view supervision for single-view reconstruction via differentiable ray consistency, in: CVPR, IEEE Computer Society, 2017, pp. 209–217.

[13] X. Zhou, A. Karpur, L. Luo, Q. Huang, Starmap for category-agnostic keypoint and viewpoint estimation, in: ECCV (1), Vol. 11205 of Lecture Notes in Computer Science, Springer, 2018, pp. 328–345.

[14] W. Wang, Q. Lai, H. Fu, J. Shen, H. Ling, Salient object detection in the deep learning era: An in-depth survey, CoRR abs/1904.09146.

[15] W. Wang, J. Shen, R. Yang, F. Porikli, Saliency-aware video object segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 40 (1) (2018) 20–33.

[16] W. Wang, J. Shen, H. Sun, L. Shao, Video co-saliency guided co-segmentation, IEEE Trans. Circuits Syst. Video Techn. 28 (8) (2018) 1727–1736.

[17] W. Wang, J. Shen, Deep visual attention prediction, IEEE Trans. Image Processing 27 (5) (2018) 2368–2378.

[18] A. Kanazawa, S. Tulsiani, A. A. Efros, J. Malik, Learning category-specific mesh reconstruction from image collections, in: ECCV (15), Vol. 11219 of Lecture Notes in Computer Science, Springer, 2018, pp. 386–402.

[19] A. Kar, S. Tulsiani, J. Carreira, J. Malik, Category-specific object reconstruction from a single image, in: CVPR, IEEE Computer Society, 2015, pp. 1966–1974.

[20] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, J. Tenenbaum, Marrnet: 3d shape reconstruction via 2.5d sketches, in: NIPS, 2017, pp. 540–550.

[21] X. Chu, W. Ouyang, H. Li, X. Wang, Structured feature learning for pose estimation, in: CVPR, IEEE Computer Society, 2016, pp. 4715–4723.

[22] H. Fang, S. Xie, Y. Tai, C. Lu, RMPE: regional multi-person pose estimation, in: ICCV, IEEE Computer Society, 2017, pp. 2353–2362.

[23] K. He, G. Gkioxari, P. Dollár, R. B. Girshick, Mask R-CNN, in: ICCV, IEEE Computer Society, 2017, pp. 2980–2988.

[24] A. Newell, K. Yang, J. Deng, Stacked hourglass networks for human pose estimation, in: ECCV (8), Vol. 9912 of Lecture Notes in Computer Science, Springer, 2016, pp. 483–499.

[25] W. Yang, S. Li, W. Ouyang, H. Li, X. Wang, Learning feature pyramids for human pose estimation, in: ICCV, IEEE Computer Society, 2017, pp. 1290–1299.

[26] S. Wei, V. Ramakrishna, T. Kanade, Y. Sheikh, Convolutional pose machines, in: CVPR, IEEE Computer Society, 2016, pp. 4724–4732.

[27] S. Huang, M. Gong, D. Tao, A coarse-fine network for keypoint localization, in: ICCV, IEEE Computer Society, 2017, pp. 3047–3056.

[28] Z. Cao, T. Simon, S. Wei, Y. Sheikh, Realtime multi-person 2d pose estimation using part affinity fields, in: CVPR, IEEE Computer Society, 2017, pp. 1302–1310.

[29] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, B. Schiele, Deepercut: A deeper, stronger, and faster multi-person pose estimation model, in: ECCV (6), Vol. 9910 of Lecture Notes in Computer Science, Springer, 2016, pp. 34–50.

[30] U. Iqbal, J. Gall, Multi-person pose estimation with local joint-to-person associations, in: ECCV Workshops (2), Vol. 9914 of Lecture Notes in Computer Science, 2016, pp. 627–642.

[31] A. Newell, Z. Huang, J. Deng, Associative embedding: End-to-end learning for joint detection and grouping, in: NIPS, 2017, pp. 2274–2284.

[32] G. Papandreou, T. Zhu, L. Chen, S. Gidaris, J. Tompson, K. Murphy, Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model, in: ECCV (14), Vol. 11218 of Lecture Notes in Computer Science, Springer, 2018, pp. 282–299.

[33] G. Pavlakos, X. Zhou, K. G. Derpanis, K. Daniilidis, Coarse-to-fine volumetric prediction for single-image 3d human pose, in: CVPR, IEEE Computer Society, 2017, pp. 1263–1272.

[34] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, C. Bregler, Efficient object localization using convolutional networks, in: CVPR, IEEE Computer Society, 2015, pp. 648–656.

[35] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, B. Schiele, Deepcut: Joint subset partition and labeling for multi person pose estimation, in: CVPR, IEEE Computer Society, 2016, pp. 4929–4937.

[36] H. Altwaijry, A. Veit, S. J. Belongie, Learning to detect and match keypoints with deep architectures, in: BMVC, BMVA Press, 2016.

[37] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, K. Daniilidis, 6-dof object pose from semantic keypoints, in: ICRA, IEEE, 2017, pp. 2011–2018.

[38] J. Thewlis, H. Bilen, A. Vedaldi, Unsupervised learning of object landmarks by factorized spatial embeddings, xxxx 11111 (2017) 1111.

[39] Y. Zhang, Y. Guo, Y. Jin, Y. Luo, Z. He, H. Lee, Unsupervised discovery of object landmarks as structural representations, in: CVPR, IEEE Computer Society, 2018, pp. 2694–2703.

[40] A. Toshev, C. Szegedy, Deeppose: Human pose estimation via deep neural networks, in: CVPR, IEEE Computer Society, 2014, pp. 1653–1660.

[41] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, K. Murphy, Towards accurate multi-person pose estimation in the wild, in: CVPR, IEEE Computer Society, 2017, pp. 3711–3719.

[42] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: CVPR, IEEE Computer Society, 2016, pp. 770–778.

[43] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, S. J. Belongie, Feature pyramid networks for object detection, CoRR abs/1612.03144.

[44] M. Kocabas, S. Karagoz, E. Akbas, Multiposenet: Fast multi-person pose estimation using pose residual network, in: ECCV (11), Vol. 11215 of Lecture Notes in Computer Science, Springer, 2018, pp. 437–453.

[45] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, D. Tran, Detect-and-track: Efficient pose estimation in videos, in: CVPR, IEEE Computer Society, 2018, pp. 350–359.

[46] B. Xiao, H. Wu, Y. Wei, Simple baselines for human pose estimation and tracking, in: ECCV (6), Vol. 11210 of Lecture Notes in Computer Science, Springer, 2018, pp. 472–487.

[47] Y. Xiu, J. Li, H. Wang, Y. Fang, C. Lu, Pose flow: Efficient online pose tracking, in: BMVC, BMVA Press, 2018, p. 53.

[48] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft COCO: common objects in context, in: ECCV (5), Vol. 8693 of Lecture Notes in Computer Science, Springer, 2014, pp. 740–755.

[49] M. Andriluka, U. Iqbal, E. Insafutdinov, L. Pishchulin, A. Milan, J. Gall, B. Schiele, Posetrack: A benchmark for human pose estimation and tracking, in: CVPR, IEEE Computer Society, 2018, pp. 5167–5176.

[50] M. Andriluka, L. Pishchulin, P. V. Gehler, B. Schiele, 2d human pose estimation: New benchmark and state of the art analysis, in: CVPR, IEEE Computer Society, 2014, pp. 3686–3693.

[51] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, T. Brox, Flownet 2.0: Evolution of optical flow estimation with deep networks, in: CVPR, IEEE Computer Society, 2017, pp. 1647–1655.

[52] H. Hu, J. Gu, Z. Zhang, J. Dai, Y. Wei, Relation networks for object detection, in: CVPR, IEEE Computer Society, 2018, pp. 3588–3597.

[53] A. Kumar, R. Chellappa, A convolution tree with deconvolution branches: Exploiting geometric relationships for single shot keypoint detection, CoRR abs/1704.01880.