Simple Baselines for Human Pose Estimation and Tracking

Bin Xiao[1(✉)], Haiping Wu[2], and Yichen Wei[1]

[1] Microsoft Research Asia, Beijing, China
{Bin.Xiao,yichenw}@microsoft.com
[2] University of Electronic Science and Technology of China, Chengdu, China
v-haipwu@microsoft.com

Abstract. There has been significant progress on pose estimation and increasing interests on pose tracking in recent years. At the same time, the overall algorithm and system complexity increases as well, making the algorithm analysis and comparison more difficult. This work provides simple and effective baseline methods. They are helpful for inspiring and evaluating new ideas for the field. State-of-the-art results are achieved on challenging benchmarks. The code will be available at https://github.com/leoxiaobin/pose.pytorch.

Keywords: Human pose estimation · Human pose tracking

1 Introduction

Similar as many vision tasks, the progress on human pose estimation problem is significantly advanced by deep learning. Since the pioneer work in [30,31], the performance on the MPII benchmark [3] has become saturated in three years, starting from about 80% PCKH@0.5 [30] to more than 90% [7,8,22,33]. The progress on the more recent and challenging COCO human pose benchmark [20] is even faster. The mAP metric is increased from 60.5 (COCO 2016 Challenge winner [5,9]) to 72.1 (COCO 2017 Challenge winner [6,9]) in one year. With the quick maturity of pose estimation, a more challenging task of "simultaneous pose detection and tracking in the wild" has been introduced recently [2].

At the same time, the network architecture and experiment practice have steadily become more complex. This makes the algorithm analysis and comparison more difficult. For example, the leading methods [7,8,22,33] on MPII benchmark [3] have considerable difference in many details but minor difference in accuracy. It is hard to tell which details are crucial. Also, the representative works [5,6,12,21,24] on COCO benchmark are also complex but differ significantly. Comparison between such works is mostly on system level and less

B. Xiao and H. Wu—Equal contribution.
H. Wu—This work is done when Haiping Wu is an intern at Microsoft Research Asia.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11210, pp. 472–487, 2018.
https://doi.org/10.1007/978-3-030-01231-1_29

informative. About pose tracking, although there has not been much work [2], the system complexity can be expected to further increase due to the increased problem dimension and solution space.

This work aims to ease this problem by asking a question from the opposite direction, *how good could a simple method be?* To answer the question, this work provides baseline methods for both pose estimation and tracking. They are quite simple but surprisingly effective. Thus, they hopefully would help inspiring new ideas and simplifying their evaluation. The code, as well as pre-trained models, will be released to facilitate the research community.

Our pose estimation is based on a few deconvolutional layers added on a backbone network, ResNet [13] in this work. It is probably the simplest way to estimate heat maps from deep and low resolution feature maps. Our *single* model's best result achieves the state-of-the-art at mAP of 73.7 on COCO test-dev split, which has an improvement of 1.6% and 0.7% over the winner of COCO 2017 keypoint Challenge's single model and their ensembled model [6,9].

Our pose tracking follows a similar pipeline of the winner [11] of ICCV'17 PoseTrack Challenge [2]. The single person pose estimation uses our own method as above. The pose tracking uses the same greedy matching method as in [11]. *Our only modification is to use optical flow based pose propagation and similarity measurement.* Our best result achieves a mAP score of 74.6 and a MOTA score of 57.8, an absolute 15% and 6% improvement over 59.6 and 51.8 of the winner of ICCV'17 PoseTrack Challenge [11,26]. It is the new state-of-the-art.

This work is not based on any theoretic evidence. It is based on simple techniques and validated by comprehensive ablation experiments, at our best. Note that we do not claim any algorithmic superiority over previous methods, in spite of better results. We do not perform complete and fair comparison with previous methods, because this is difficult and not our intent. As stated, the contribution of this work are solid baselines for the field.

2 Pose Estimation Using a Deconvolution Head Network

ResNet [13] is the most common backbone network for image feature extraction. It is also used in [6,24] for pose estimation. Our method simply adds a few deconvolutional layers over the last convolution stage in the ResNet, called C_5. The whole network structure is illustrated in Fig. 1(c). We adopt this structure because it is arguably the simplest to generate heatmaps from deep and low resolution features and also adopted in the state-of-the-art Mask R-CNN [12].

By default, three deconvolutional layers with batch normalization [15] and ReLU activation [19] are used. Each layer has 256 filters with 4×4 kernel. The stride is 2. A 1×1 convolutional layer is added at last to generate predicted heatmaps $\{H_1 \dots H_k\}$ for all k key points.

Same as in [22,30], Mean Squared Error (MSE) is used as the loss between the predicted heatmaps and targeted heatmaps. The targeted heatmap $\hat{H}_k$ for joint k is generated by applying a 2D gaussian centered on the k^{th} joint's ground truth location.

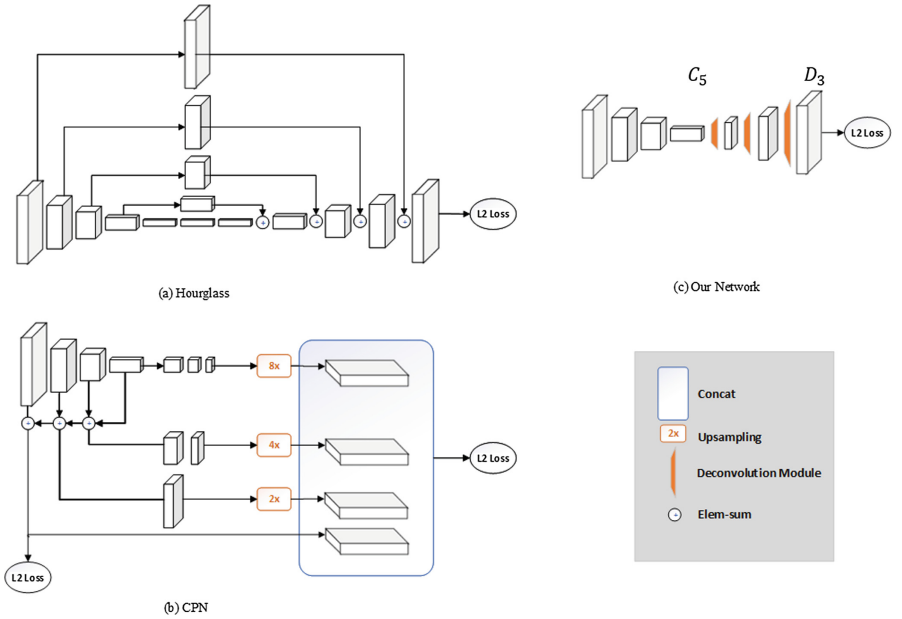

Fig. 1. Illustration of two state-of-the-art network architectures for pose estimation (a) one stage in Hourglass [22], (b) CPN [6], and our simple baseline (c).

Discussions. To understand the simplicity and rationality of our baseline, we discuss two state-of-the-art network architectures as references, namely, Hourglass [22] and CPN [6]. They are illustrated in Fig. 1.

Hourglass [22] is the dominant approach on MPII benchmark as it is the basis for all leading methods [7,8,33]. It features in a multi-stage architecture with repeated bottom-up, top-down processing and skip layer feature concatenation.

Cascaded pyramid network (CPN) [6] is the leading method on COCO 2017 keypoint challenge [9]. It also involves skip layer feature concatenation and an online hard keypoint mining step.

Comparing the three architectures in Fig. 1, it is clear that our method differs from [6,22] in *how high resolution feature maps are generated*. Both works [6,22] use upsampling to increase the feature map resolution and put convolutional parameters in other blocks. In contrary, our method combines the upsampling and convolutional parameters into deconvolutional layers in a much simpler way, without using skip layer connections.

A commonality of the three methods is that three upsampling steps and also three levels of non-linearity (from the deepest feature) are used to obtain high-resolution feature maps and heatmaps. Based on above observations and the good performance of our baseline, it seems that *obtaining high resolution feature maps is crucial, but no matter how*. Note that this discussion is only preliminary and heuristic. It is hard to conclude which architecture in Fig. 1 is better. This is not the intent of this work.

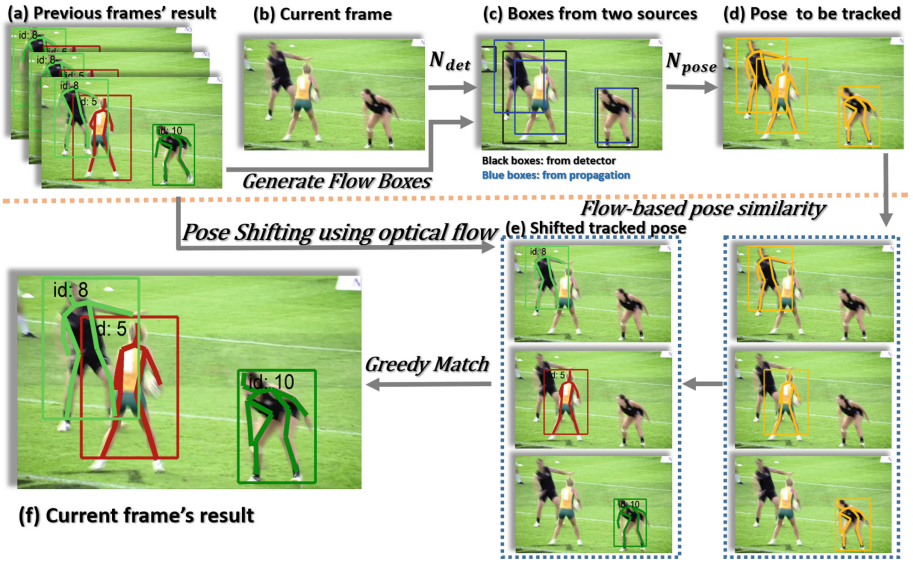

Fig. 2. The proposed flow-based pose tracking framework.

3 Pose Tracking Based on Optical Flow

Multi-person pose tracking in videos first estimates human poses in frames, and then tracks these human pose by assigning a unique identification number (id) to them across frames. We present human instance P with id as $P = (J, id)$, where $J = \{j_i\}_{1:N_J}$ is the coordinates set of N_J body joints and id indicates the tracking id. When processing the k^{th} frame I^k, we have the already processed human instances set $\mathcal{P}^{k-1} = \{P_i^{k-1}\}_{1:N_{k-1}}$ in frame I^{k-1} and the instances set $\mathcal{P}^k = \{P_i^k\}_{1:N_k}$ in frame I^k whose id is to be assigned, where N_{k-1} and N_k are the instance number in frame I^{k-1} and I^k. If one instance P_j^k in current frame I^k is linked to the instance P_i^{k-1} in I^{k-1} frame, then id_i^{k-1} is propagated to id_j^k, otherwise a new id is assigned to P_j^k, indicating a new track.

The winner [11] of ICCV'17 PoseTrack Challenge [2] solves this multi-person pose tracking problem by first estimating human pose in frames using Mask R-CNN [12], and then performing online tracking using a greedy bipartite matching algorithm frame by frame.

The greedy matching algorithm is to first assign the id of P_i^{k-1} in frame I^{k-1} to P_j^k in frame I^k if the similarity between P_i^{k-1} and P_j^k is the highest, then remove these two instances from consideration, and repeat the id assigning process with the highest similarity. When an instance P_j^k in frame I^k has no existing P_i^{k-1} left to link, a new id number is assigned, which indicates a new instance comes up.

We mainly follow this pipeline in [11] with two differences. One is that we have two different kinds of human boxes, one is from a human detector and

the other are boxes generated from previous frames using optical flow. The second difference is the similarity metric used by the greedy matching algorithm. We propose to use a flow-based pose similarity metric. Combined with these two modifications, we have our enhanced flow-based pose tracking algorithm, illustrated in Fig. 2. We elaborate our flow-based pose tracking algorithm in the following.

3.1 Joint Propagation Using Optical Flow

Simply applying a detector designed for single image level (e.g. Faster-RCNN [27], R-FCN [16]) to videos could lead to missing detections and false detections due to motion blur and occlusion introduced by video frames. As shown in Fig. 2(c), the detector misses the left black person due to fast motion. Temporal information is often leveraged to generate more robust detections [35,36].

We propose to generate boxes for the processing frame from nearby frames using temporal information expressed in optical flow.

Given one human instance with joints coordinates set J_i^{k-1} in frame I^{k-1} and the optical flow field $F_{k-1\to k}$ between frame I^{k-1} and I^k, we could estimate the corresponding joints coordinates set $\hat{J}_i^k$ in frame I^k by propagating the joints coordinates set J_i^{k-1} according to $F_{k-1\to k}$. More specifically, for each joint location (x, y) in J_i^{k-1}, the propagated joint location would be $(x + \delta x, y + \delta y)$, where $\delta x, \delta y$ are the flow field values at joint location (x, y). Then we compute a bounding of the propagated joints coordinates set $\hat{J}_i^k$, and expand that box by some extend (15% in experiments) as the candidates box for pose estimation.

When the processing frame is difficult for human detectors that could lead to missing detections due to motion blur or occlusion, we could have boxes propagated from previous frames where people have been detected correctly. As shown in Fig. 2(c), for the left black person in images, since we have the tracked result in previous frames in Fig. 2(a), the propagated boxes successfully contain this person.

3.2 Flow-Based Pose Similarity

Using bounding box IoU (Intersection-over-Union) as the similarity metric (S_{Bbox}) to link instances could be problematic when an instance moves fast thus the boxes do not overlap, and in crowed scenes where boxes may not have the corresponding relationship with instances. A more fine-grained metric could be a pose similarity (S_{Pose}) which calculates the body joints distance between two instances using Object Keypoint Similarity (OKS). The pose similarity could also be problematic when the pose of the same person is different across frames due to pose changing. We propose to use a flow-based pose similarity metric.

Given one instance J_i^k in frame I^k and one instance J_j^l in frame I^l, the flow-based pose similarity metric is represented as

$$S_{Flow}(J_i^k, J_j^l) = OKS(\hat{J}_i^l, J_j^l), \tag{1}$$

where OKS represents calculating the Object Keypoint Similarity (OKS) between two human pose, and $\hat{J}_i^l$ represents the propagated joints for J_i^k from frame I^k to I^l using optical flow field $F_{k\to l}$.

Due to occlusions with other people or objects, people often disappear and re-appear again. Considering consecutive two frames is not enough, thus we have the flow-based pose similarity considering multi frames, denoted as $S_{Multi-flow}$, meaning the propagated $\hat{J}_k$ comes from multi previous frames. In this way, we could relink instances even disappearing in middle frames.

3.3 Flow-Based Pose Tracking Algorithm

With the joint propagation using optical flow and the flow-based pose similarity, we propose the flow-based pose tracking algorithm combining these two, as presented in Algorithm 1. Table 1 summarizes the notations used in Algorithm 1.

Table 1. Notations in Algorithm 1.

I^k	k^{th} frame
Q	Tracked instances queue
L_Q	Max capacity of Q
$\mathcal{P}^k$	Instances set in k^{th} frame
$\mathcal{J}^k$	Instances set of body joints in k^{th} frame
P_i^k	i^{th} instance in k^{th} frame
J_i^k	Body joints set of i^{th} instance in k^{th} frame
$F_{k\to l}$	Flow field from k^{th} frame to l^{th} frame
M_{sim}	Similarity matrix
B_{det}^k	Boxes from person detector in k^{th} frame
B_{flow}^k	Boxes generated by joint propagating in k^{th} frame
B_{unified}^k	Boxes unified by box NMS in k^{th} frame
$\mathcal{N}_{\text{det}}$	Person detection network
$\mathcal{N}_{\text{pose}}$	Human pose estimation network
$\mathcal{N}_{\text{flow}}$	Flow estimation network
$\mathcal{F}_{\text{sim}}$	Function for calculating similarity matrix
$\mathcal{F}_{\text{NMS}}$	Function for NMS operation
$\mathcal{F}_{\text{FlowBoxGen}}$	Function for generating boxes by joint propagating
$\mathcal{F}_{\text{AssignID}}$	Function for assigning instance id

First, we solve the pose estimation problem. For the processing frame in videos, the boxes from a human detector and boxes generated by propagating joints from previous frames using optical flow are unified using a bounding box

Algorithm 1. The flow-based inference algorithm for video human pose tracking

input: video frames $\{I^k\}$, $Q = []$, Q's max capacity L_Q.
$B^0_{\text{det}} = \mathcal{N}_{\text{det}}(I^0)$
$\mathcal{J}^0 = \mathcal{N}_{\text{pose}}(I^0, B^0_{\text{det}})$
$\mathcal{P}^0 = (\mathcal{J}^0, id)$ ▷ initialize the id for the first frame
$Q = [\mathcal{P}_0]$ ▷ append the instance set $\mathcal{P}_0$ to Q
for $k = 1$ **to** ∞ **do**
  $B^k_{\text{det}} = \mathcal{N}_{\text{det}}(I^k)$
  $B^k_{\text{flow}} = \mathcal{F}_{\text{FlowBoxGen}}(\mathcal{J}^{k-1}, F_{k-1\to k})$
  $B^k_{\text{unified}} = \mathcal{F}_{\text{NMS}}(B^k_{\text{det}}, B^k_{\text{flow}})$ ▷ unify detection boxes and flow boxes
  $\mathcal{J}^k = \mathcal{N}_{\text{pose}}(I^k, B^k_{\text{unified}})$
  $M_{\text{sim}} = \mathcal{F}_{\text{sim}}(Q, \mathcal{J}^k)$
  $\mathcal{P}^k = \mathcal{F}_{\text{AssignID}}(M_{\text{sim}}, \mathcal{J}^k)$
  append $\mathcal{P}^k$ to Q ▷ update the Q
end for

Non-Maximum Suppression (NMS) operation. The boxes generated by propagating joints serve as the complement of missing detections of the detector (e.g. in Fig. 2(c)). Then we estimate human pose using the cropped and resized images by these boxes through our proposed pose estimation network in Sect. 2.

Second, we solve the tracking problem. We store the tracked instances in a double-ended queue (Deque) with fixed length L_Q, denoted as

$$Q = [\mathcal{P}_{k-1}, \mathcal{P}_{k-2}, \ldots, \mathcal{P}_{k-L_Q}] \tag{2}$$

where $\mathcal{P}_{k-i}$ means tracked instances set in previous frame I^{k-i} and the Q's length L_Q indicates how many previous frames considered when performing matching.

The Q could be used to capture previous multi frames' linking relationship, initialized in the first frame in a video. For the k^{th} frame I^k, we calculate the flow-based pose similarity matrix M_{sim} between the untracked instances set of body joints $\mathcal{J}^k$ (id is none) and previous instances sets in Q. Then we assign id to each body joints instance J in $\mathcal{J}^k$ to get assigned instance set $\mathcal{P}^k$ by using greedy matching and M_{sim}. Finally we update our tracked instances Q by adding up k^{th} frame instances set $\mathcal{P}^k$.

4 Experiments

4.1 Pose Estimation on COCO

The COCO Keypoint Challenge [20] requires localization of multi-person keypoints in challenging uncontrolled conditions. The COCO train, validation, and test sets contain more than 200k images and 250k person instances labeled with keypoints. 150k instances of them are publicly available for training and validation. Our models are only trained on all COCO *train2017* dataset (includes 57K images and 150K person instances) no extra data involved, ablation are studied

on the *val2017* set and finally we report the final results on *test-dev2017* set to make a fair comparison with the public state-of-the-art results [5,6,12,24].

The COCO evaluation defines the object keypoint similarity (OKS) and uses the mean average precision (AP) over 10 OKS thresholds as main competition metric [9]. The OKS plays the same role as the IoU in object detection. It is calculated from the distance between predicted points and ground truth points normalized by scale of the person.

Training. The ground truth human box is made to a fixed aspect ratio, e.g., $height{:}width = 4{:}3$ by extending the box in height or width. It is then cropped from the image and resized to a fixed resolution. The default resolution is 256:192. It is the same as the state-of-the-art method [6] for a fair comparison. Data augmentation includes scale ($\pm 30\%$), rotation ($\pm 40°$) and flip.

Our ResNet [13] backbone network is initialized by pre-training on ImageNet classification task [28]. In the training for pose estimation, the base learning rate is 1e−3. It drops to 1e−4 at 90 epochs and 1e−5 at 120 epochs. There are 140 epochs in total. Mini-batch size is 128. Adam [18] optimizer is used. Four GPUs on a GPU server is used.

ResNet of depth 50, 101 and 152 layers are experimented. ResNet-50 is used by default, unless otherwise noted.

Testing. A two-stage top-down paradigm is applied, similar as in [6,24]. For detection, by default we use a faster-RCNN [27] detector with detection AP 56.4 for the person category on COCO *val2017*. Following the common practice in [6,22], the joint location is predicted on the averaged heatmaps of the original and flipped image. A quarter offset in the direction from highest response to the second highest response is used to obtain the final location.

Table 2. Ablation study of our method on COCO val2017 dataset. Those settings used in comparison are in **bold**. For example, (a, e, f) compares backbones.

Method	Backbone	Input size	#Deconv. layers	Deconv. kernel size	*AP*
a	**ResNet-50**	**256 × 192**	**3**	**4**	70.4
b	ResNet-50	256 × 192	**2**	4	67.9
c	ResNet-50	256 × 192	3	**2**	70.1
d	ResNet-50	256 × 192	3	**3**	70.3
e	**ResNet-101**	256 × 192	3	4	71.4
f	**ResNet-152**	256 × 192	3	4	72.0
g	ResNet-50	**128 × 96**	3	4	60.6
h	ResNet-50	**384 × 288**	3	4	72.2

Ablation Study. Table 2 investigates various options in our baseline in Sect. 2.

1. *Heat map resolution.* Method (a) uses three deconvolutional layers to generate 64×48 heatmaps. Method (b) generates 32×24 heatmaps with two deconvolutional layers. (a) outperform (b) by 2.5 AP with only slightly increased model capacity. By default, three deconvolutional layers are used.
2. *Kernel size.* Methods (a, c, d) show that a smaller kernel size gives a marginally decrease in AP, which is 0.3 point decrease from kernel size 4 to 2. By default, deconvolution kernel size of 4 is used.
3. *Backbone.* As in most vision tasks, a deeper backbone model has better performance. Methods (a, e, f) show steady improvement by using deeper backbone models. AP increase is 1.0 from ResNet-50 to Resnet-101 and 1.6 from ResNet-50 to ResNet-152.
4. *Image size.* Methods (a, g, h) show that image size is critical for performance. From method (a) to (g), the image size is reduced by half and AP drops points. On the other hand, relative 75% computation is saved. Method (h) uses a large image size and increases 1.8 AP from method (a), at the cost of higher computational cost.

Table 3. Comparison with Hourglass [22] and CPN [6] on COCO val2017 dataset. Their results are cited from [6]. OHKM means Online Hard Keypoints Mining.

Method	Backbone	Input size	OHKM	*AP*
8-stage Hourglass	-	256×192	✗	66.9
8-stage Hourglass	-	256×256	✗	67.1
CPN	ResNet-50	256×192	✗	68.6
CPN	ResNet-50	384×288	✗	70.6
CPN	ResNet-50	256×192	✓	69.4
CPN	ResNet-50	384×288	✓	71.6
Ours	ResNet-50	256×192	✗	70.4
Ours	ResNet-50	384×288	✗	72.2

Comparison with Other Methods on COCO val2017. Table 3 compares our results with a 8-stage Hourglass [22] and CPN [6]. All the three methods use a similar top-down two-stage paradigm. For reference, the person detection AP of hourglass [22] and CPN [6] is 55.3 [6], which is comparable to ours 56.4.

Compared with Hourglass [6,22], our baseline has an improvement of 3.5 in AP. Both methods use an input size of 256×192 and no Online Hard Keypoints Mining (OHKM) involved.

CPN [6] and our baseline use the same backbone of ResNet-50. When OHKM is not used, our baseline outperforms CPN [6] by 1.8 AP for input size 256×192, and 1.6 AP for input size 384×288. When OHKM is used in CPN [6], our baseline is better by 0.6 AP for both input sizes.

Note that the results of Hourglass [22] and CPN [6] are cited from [6] and not implemented by us. Therefore, the performance difference could come from implementation difference. Nevertheless, we believe it is safe to conclude that our baseline has comparable results but is simpler.

Table 4. Comparisons on COCO test-dev dataset. Top: methods in the literature, trained only on COCO training dataset. Middle: results submitted to COCO test-dev leaderboard [9], which have either extra training data (*) or models ensamled ($^+$). Bottom: our single model results, trained only on COCO training dataset.

Method	Backbone	Input size	AP	AP_{50}	AP_{75}	AP_m	AP_l	AR
CMU-Pose [5]	-	-	61.8	84.9	67.5	57.1	68.2	66.5
Mask-RCNN [12]	ResNet-50-FPN	-	63.1	87.3	68.7	57.8	71.4	-
G-RMI [24]	ResNet-101	353 × 257	64.9	85.5	71.3	62.3	70.0	69.7
CPN [6]	ResNet-Inception	384 × 288	72.1	91.4	80.0	68.7	77.2	78.5
FAIR* [9]	ResNeXt-101-FPN	-	69.2	90.4	77.0	64.9	76.3	75.2
G-RMI* [9]	ResNet-152	353 × 257	71.0	87.9	77.7	69.0	75.2	75.8
oks* [9]	-	-	72.0	90.3	79.7	67.6	78.4	77.1
bangbangren*$^+$ [9]	ResNet-101	-	72.8	89.4	79.6	68.6	**80.0**	78.7
CPN$^+$ [6,9]	ResNet-Inception	384 × 288	73.0	**91.7**	80.9	69.5	78.1	**79.0**
Ours	ResNet-152	384 × 288	**73.7**	**91.9**	**81.1**	**70.3**	**80.0**	**79.0**

Comparisons on COCO test-dev Dataset. Table 4 summarizes the results of other state-of-the-art methods in the literature on COCO Keypoint Leaderboard [9] and COCO *test-dev* dataset. For our baseline here, a human detector with *person detection* AP of 60.9 on COCO *std-dev* split dataset is used. For reference, CPN [6] use a human detector with *person detection* AP of 62.9 on COCO *minival* split dataset.

Compared with CMU-Pose [5], which is a bottom-up approach for multi-person pose estimation, our method is significantly better. Both G-RMI [24] and CPN [6] have a similar top-down pipeline with ours. G-RMI also uses ResNet as backbone, as ours. Using the same backbone Resnet-101, our method outperforms G-RMI for both small (256 × 192) and large input size (384 × 288). CPN uses a stronger backbone of ResNet-Inception [29]. As evidence, the top-1 error rate on ImageNet validation set of Resnet-Inception and ResNet-152 are 18.7% and 21.4% respectively [29]. Yet, for the same input size 384 × 288, our result 73.7 outperforms both CPN's single model and their ensembled model, which have 72.1 and 73.0 respectively.

4.2 Pose Estimation and Tracking on PoseTrack

PoseTrack [2] dataset is a large-scale benchmark for multi-person pose estimation and tracking in videos. It requires not only pose estimation in single frames, but also temporal tracking across frames. It contains 514 videos including 66,374

frames in total, split into 300, 50 and 208 videos for training, validation and test set respectively. For training videos, 30 frames from the center are annotated. For validation and test videos, besides 30 frames from the center, every fourth frame is also annotated for evaluating long range articulated tracking. The annotations include 15 body keypoints location, a unique person id and a head bounding box for each person instance.

The dataset has three tasks. Task 1 evaluates single-frame pose estimation using mean average precision (mAP) metric as is done in [25]. Task 2 also evaluates pose estimation but allows usage of temporal information across frames. Task 3 evaluates tracking using multi-object tracking metrics [4]. As our tracking baseline uses temporal information, we report results on Task 2 and 3. Note that our pose estimation baseline also performs best on Task 1 but is not reported here for simplicity.

Training. Our pose estimation model is fine-tuned from those pre-trained on COCO in Sect. 4.1. As only key points are annotated, we obtain the ground truth box of a person instance by extending the bounding box of its all key points by 15% in length (7.5% on both sides). The same data augmentation as in Sect. 4.1 is used. During training, the base learning rate is 1e−4. It drops to 1e−5 at 10 epochs and 1e−6 at 15 epochs. There are 20 epochs in total. Other hyper parameters are the same as in Sect. 4.1.

Testing. Our flow based tracking baseline is closely related to the human detector's performance, as the propagated boxes could affect boxes from a detector. To investigate its effect, we experiment with two off-the-shelf detectors, a faster but less accurate R-FCN [16] and a slower but more accurate FPN-DCN [10]. Both use ResNet-101 backbone and are obtained from public implementation [1]. No additional fine tuning of detectors on PoseTrack dataset is performed.

Similar as in [11], we first drop low-confidence detections, which tends to decrease the mAP metric but increase the MOTA tracking metric. Also, since the tracking metric MOT penalizes false positives equally regardless of the scores, we drop low confidence joints first to generate the result as in [11]. We choose the boxes and joints drop threshold in a data-driven manner on validation set, 0.5 and 0.4 respectively.

For optical flow estimation, the fastest model FlowNet2S in FlowNet family [14] is used, as provided on [23]. We use the PoseTrack evaluation toolkit for results on validation set and report final results on test set from the evaluation server. Figure 3 illustrates some results of our approach on PoseTrack test dataset.

Our main ablation study is performed on ResNet-50 with input size 256×192, which is already strong when compared with state-of-the-art. Our best result is on ResNet-152 with input size 384×288.

Effect of Joint Propagation. Table 5 shows that using boxes from joint propagation introduces improvement on both mAP and MOTA metrics using different

Fig. 3. Some sample results on PoseTrack Challenge test set.

Table 5. Ablation study on PoseTrack Challenge validation dataset. Top: Results of ResNet-50 backbone using R-FCN detector. Middle: Results of ResNet-50 backbone using FPN-DCN detector. Bottom:Results of ResNet-152 backbone using FPN-DCN detector.

Method	Backbone	Detector	With joint propagation	Similarity metric	mAP total	MOTA total
a_1	ResNet-50	R-FCN	✗	S_{Bbox}	66.0	57.6
a_2	ResNet-50	R-FCN	✗	S_{Pose}	66.0	57.7
a_3	ResNet-50	R-FCN	✓	S_{Bbox}	70.3	61.4
a_4	ResNet-50	R-FCN	✓	S_{Pose}	70.3	61.8
a_5	ResNet-50	R-FCN	✓	S_{Flow}	70.3	61.8
a_6	ResNet-50	R-FCN	✓	$S_{Multi-Flow}$	**70.3**	**62.2**
b_1	ResNet-50	FPN-DCN	✗	S_{Bbox}	69.3	59.8
b_2	ResNet-50	FPN-DCN	✗	S_{Pose}	69.3	59.7
b_3	ResNet-50	FPN-DCN	✓	S_{Bbox}	72.4	62.1
b_4	ResNet-50	FPN-DCN	✓	S_{Pose}	72.4	61.8
b_5	ResNet-50	FPN-DCN	✓	S_{Flow}	72.4	62.4
b_6	ResNet-50	FPN-DCN	✓	$S_{Multi-Flow}$	**72.4**	**62.9**
c_1	ResNet-152	FPN-DCN	✗	S_{Bbox}	72.9	62.0
c_2	ResNet-152	FPN-DCN	✗	S_{Pose}	72.9	61.9
c_3	ResNet-152	FPN-DCN	✓	S_{Bbox}	76.7	64.8
c_4	ResNet-152	FPN-DCN	✓	S_{Pose}	76.7	64.9
c_5	ResNet-152	FPN-DCN	✓	S_{Flow}	76.7	65.1
c_6	ResNet-152	FPN-DCN	✓	$S_{Multi-Flow}$	**76.7**	**65.4**

Table 6. Multi-person pose estimation performance on PoseTrack Challenge dataset. "*" means models trained on train + validation set. Top: Results on PoseTrack validation set. Bottom: Results on PoseTrack test set

Method	Dataset	Head mAP	Sho. mAP	Elb. mAP	Wri. mAP	Hip mAP	Knee mAP	Ank. mAP	Total mAP
Girdhar et al. [11]	val	67.5	70.2	62.0	51.7	60.7	58.7	49.8	60.6
Xiu et al. [32]	val	66.7	73.3	68.3	61.1	67.5	67.0	61.3	66.5
Ours:ResNet-50	val	79.1	80.5	75.5	66.0	70.8	70.0	61.7	72.4
Ours:ResNet-152	val	81.7	83.4	80.0	72.4	75.3	74.8	67.1	**76.7**
Girdhar et al.* [11]	test	-	-	-	-	-	-	-	59.6
Xiu et al. [32]	test	64.9	67.5	65.0	59.0	62.5	62.8	57.9	63.0
Ours:ResNet-50	test	76.4	77.2	72.2	65.1	68.5	66.9	60.3	70.0
Ours:ResNet-152	test	79.5	79.7	76.4	70.7	71.6	71.3	64.9	**73.9**

backbones and detectors. With R-FCN detector, using boxes from joint propagation (method a_3 vs. a_1) introduces improvement of 4.3% mAP and 3.8% MOTA. With the better FPN-DCN detector, using boxes from joint propagation (method b_3 vs. b_1) introduces improvement of 3.1% mAP and 2.3% MOTA. With ResNet-152 as backbone (method c_3 vs. c_1), improvement is 3.8% mAP and 2.8% MOTA. Note that such improvement does not only come from more boxes. As noted in [11], simply keeping more boxes of a detector, e.g., by using a smaller threshold, would lead to an improvement in mAP, but a drop in MOTA since more false positives would be introduced. The joint propagation improves both mAP and MOTA metrics, indicating that it finds more persons that are missed by the detector, possibly due to motion blur or occlusion in video frames.

Another interesting observation is that the less accurate R-FCN detector benefits more from joint propagation. For example, the gap between using FPN-DCN and R-FCN detector in ResNet-50 is decreased from 3.3% mAP and 2.2% MOTA (from a_1 to b_1) to 2.1% mAP and 0.4% MOTA (from a_3 to b_3). Also, method a_3 outperforms method b_1 by 1.0% mAP and 1.6% MOTA, indicating that a weak detector R-FCN combined with joint propagation could perform better than a strong detector FPN-DCN along. While, the former is more efficient as joint propagation is fast.

Effect of Flow-Based Pose Similarity. Flow-based pose similarity is shown working better when compared with bounding box similarity and pose similarity in Table 5. For example, flow-based similarity using multi frames (method b_6) and single frame (method b_5) outperforms bounding box similarity (method b_3) by 0.8% MOTA and 0.3% MOTA.

Note that flow-based pose similarity is better than bounding box similarity when person moves fast and their boxes do not overlap. Method b_6 with flow-based pose similarity considers multi frames and have an 0.5% MOTA improvement when compared to method b_5, which considers only one previous frame. This improvement comes from the case when people are lost shortly due to occlusion and appear again.

Comparison with State-of-the-Art. We report our results on both Task 2 and Task 3 on PoseTrack dataset. As verified in Table 5, method b_6 and c_6 are the best settings and used here. Backbones are ResNet-50 and ResNet-152, respectively. The detector is FPN-DCN [10].

Table 6 reports the results on pose estimation (Task 2). Our small model (ResNet-50) outperforms the other methods already by a large margin. Our larger model (ResNet-152) further improves the state-of-the-art. On validation set it has an absolute 16.1% improvement in mAP over [11], which is the winner of ICCV'17 PoseTrack Challenge, and also has an 10.2% improvement over a recent work [32], which is the previous best.

Table 7 reports the results on pose tracking (Task 3). Compared with [11] on validation and test dataset, our larger model (ResNet-152) has an 10.2 and 5.8 improvement in MOTA over its 55.2 and 51.8 respectively. Compared with the recent work [32], our best model (ResNet-152) has 7.1% and 6.6% improvement on validation and test dataset respectively. Note that our smaller model (ResNet-50) also outperform the other methods [11,32].

Table 8 summarizes the results on PoseTrack's leaderboard. Our baseline outperforms all public entries by a large margin. Note that all methods differ significantly and this comparison is only on system level.

Table 7. Multi-person Pose Tracking Performance on PoseTrack Challenge dataset. "*" means models trained on train + validation set. Top: Results on PoseTrack validation set. Bottom: Results on PoseTrack test set

Method	Dataset	MOTA Head	MOTA Sho.	MOTA Elb.	MOTA Wri.	MOTA Hip	MOTA Knee	MOTA Ank.	MOTA Total	MOTP Total	Prec Total	Rec Total
Girdhar et al. [11]	val	61.7	65.5	57.3	45.7	54.3	53.1	45.7	55.2	61.5	66.4	88.1
Xiu et al. [32]	val	59.8	67.0	59.8	51.6	60.0	58.4	50.5	58.3	67.8	70.3	87.0
Ours:ResNet-50	val	72.1	74.0	61.2	53.4	62.4	61.6	50.7	62.9	84.5	86.3	76.0
Ours:ResNet-152	val	73.9	75.9	63.7	56.1	65.5	65.1	53.5	**65.4**	85.4	85.5	80.3
Girdhar et al.* [11]	test	-	-	-	-	-	-	-	51.8	-	-	-
Xiu et al. [32]	test	52.0	57.4	52.8	46.6	51.0	51.2	45.3	51.0	16.9	71.2	78.9
Ours:ResNet-50	test	65.9	67.0	51.5	48.0	56.2	54.6	46.9	56.4	45.5	81.0	75.7
Ours:ResNet-152	test	67.1	68.4	52.2	48.9	56.1	56.6	48.8	**57.6**	62.6	79.4	79.9

Table 8. Results of mulit-person pose tracking on PoseTrack challenge leaderboard. "*" means models trained on train + validation set.

Entry	Additional training dataset	mAP	MOTA
ProTracker [11]	COCO	59.6	51.8
PoseFlow [26]	COCO + MPII-Pose	63.0	51.0
MVIG [26]	COCO + MPII-Pose	63.2	50.7
BUTD2 [17]	COCO	59.2	50.6
SOPT-PT [26]	COCO + MPII-Pose	58.2	42.0
ML-LAB [34]	COCO + MPII-Pose	70.3	41.8
Ours:ResNet152*	COCO	**74.6**	**57.8**

5 Conclusions

We present simple and strong baselines for pose estimation and tracking. They achieve state-of-the-art results on challenging benchmarks. They are validated via comprehensive ablation studies. We hope such baselines would benefit the field by easing the idea development and evaluation.

References

1. Deformable-ConvNet. https://github.com/msracver/Deformable-ConvNets
2. Andriluka, M., et al.: PoseTrack: a benchmark for human pose estimation and tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5167–5176 (2018)
3. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: new benchmark and state of the art analysis. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2014
4. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: the CLEAR MOT metrics. J. Image Video Process. **2008**, 1 (2008)
5. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: CVPR (2017)
6. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. In: CVPR (2018)
7. Chen, Y., Shen, C., Wei, X.S., Liu, L., Yang, J.: Adversarial posenet: a structure-aware convolutional network for human pose estimation. In: IEEE International Conference on Computer Vision, pp. 1212–1221 (2017)
8. Chu, X., Yang, W., Ouyang, W., Ma, C., Yuille, A.L., Wang, X.: Multi-context attention for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1831–1840 (2017)
9. COCO: COCO Leader Board. http://cocodataset.org
10. Dai, J., et al.: Deformable convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 764–773 (2017)
11. Girdhar, R., Gkioxari, G., Torresani, L., Paluri, M., Tran, D.: Detect-and-track: efficient pose estimation in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 350–359 (2018)
12. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988. IEEE (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
14. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: FlowNet 2.0: evolution of optical flow estimation with deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2 (2017)
15. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456 (2015)
16. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: NIPS (2016)
17. Jin, S., et al.: Towards multi-person pose tracking: bottom-up and top-down methods. In: ICCV PoseTrack Workshop (2017)

18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
20. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
21. Newell, A., Huang, Z., Deng, J.: Associative embedding: end-to-end learning for joint detection and grouping. In: Advances in Neural Information Processing Systems, pp. 2274–2284 (2017)
22. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 483–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_29
23. NVIDIA: flownet2-pytorch (2018). https://github.com/NVIDIA/flownet2-pytorch. Accessed March 2018
24. Papandreou, G., et al.: Towards accurate multi-person pose estimation in the wild. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3711–3719. IEEE (2017)
25. Pishchulin, L., et al.: DeepCut: joint subset partition and labeling for multi person pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4929–4937 (2016)
26. PoseTrack: PoseTrack Leader Board. https://posetrack.net/leaderboard.php
27. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
28. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
29. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI, vol. 4, p. 12 (2017)
30. Tompson, J.J., Jain, A., LeCun, Y., Bregler, C.: Joint training of a convolutional network and a graphical model for human pose estimation. In: Advances in Neural Information Processing Systems, pp. 1799–1807 (2014)
31. Toshev, A., Szegedy, C.: DeepPose: human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1653–1660 (2014)
32. Xiu, Y., Li, J., Wang, H., Fang, Y., Lu, C.: Pose Flow: efficient online pose tracking. arXiv preprint arXiv:1802.00977 (2018)
33. Yang, W., Li, S., Ouyang, W., Li, H., Wang, X.: Learning feature pyramids for human pose estimation. In: IEEE International Conference on Computer Vision (2017)
34. Zhu, X., Jiang, Y., Luo, Z.: Multi-person pose estimation for posetrack with enhanced part affinity fields. In: ICCV PoseTrack Workshop (2017)
35. Zhu, X., Wang, Y., Dai, J., Yuan, L., Wei, Y.: Flow-guided feature aggregation for video object detection. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 408–417. IEEE (2017)
36. Zhu, X., Xiong, Y., Dai, J., Yuan, L., Wei, Y.: Deep feature flow for video recognition. In: Proceedings of the CVPR, vol. 2, p. 7 (2017)