Dynamic Kernel Distillation for Efficient Pose Estimation in Videos

Xuecheng Nie[1,*] Yuncheng Li[2] Linjie Luo[3] Ning Zhang[4] Jiashi Feng[1]
[1]Department of Electrical and Computer Engineering, National University of Singapore, Singapore
[2]Snap Inc. [3]ByteDance AI Lab [4]DawnLight Technologies Inc.
niexuecheng@u.nus.edu raingomm@gmail.com linjie.luo@bytedance.com
ning@dawnlight.com elefjia@nus.edu.sg

Abstract

Existing video-based human pose estimation methods extensively apply large networks onto every frame in the video to localize body joints, which suffer high computational cost and hardly meet the low-latency requirement in realistic applications. To address this issue, we propose a novel Dynamic Kernel Distillation (DKD) model to facilitate small networks for estimating human poses in videos, thus significantly lifting the efficiency. In particular, DKD introduces a light-weight distillator to online distill pose kernels via leveraging temporal cues from the previous frame in a one-shot feed-forward manner. Then, DKD simplifies body joint localization into a matching procedure between the pose kernels and the current frame, which can be efficiently computed via simple convolution. In this way, DKD fast transfers pose knowledge from one frame to provide compact guidance for body joint localization in the following frame, which enables utilization of small networks in video-based pose estimation. To facilitate the training process, DKD exploits a temporally adversarial training strategy that introduces a temporal discriminator to help generate temporally coherent pose kernels and pose estimation results within a long range. Experiments on Penn Action and Sub-JHMDB benchmarks demonstrate outperforming efficiency of DKD, specifically, $10\times$ flops reduction and $2\times$ speedup over previous best model, and its state-of-the-art accuracy.

1. Introduction

Human pose estimation in videos aims to generate frame-wise joint localization of the human body. It is important for many applications including surveillance [8], computer animation [18], and AR/VR [19]. Compared to its still-image based counterpart, this task is more challenging due to its low-latency requirement and various distracting factors, *e.g.*, motion blur, pose variation and viewpoint change.

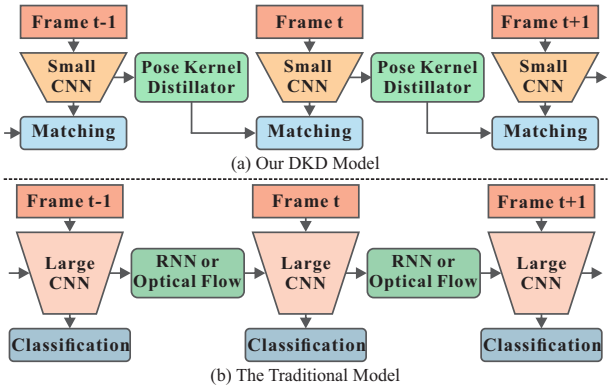


Figure 1. Comparison between (a) our DKD model and (b) the traditional model for video-based human pose estimation. DKD online distills coherent pose knowledge and simplifies body joint localization into a matching procedure, facilitating small networks to efficiently estimate human pose in videos while achieving outperforming accuracy. See text for details.

Prior CNN based methods to solve this task [10, 22, 25, 20] usually use a large network to extract representative features for every frame and localize body joints based on them via pixel-wise classification. Some recent works also incorporate temporal cues from optical flow [9] or RNN units [30] to improve the performance, as shown in Fig. 1 (b). Despite their notable accuracy, these methods suffer expensive computation cost from the large model size, and hardly meet the low-latency requirement for realistic applications. The efficiency of video-based pose estimation still needs to be largely enhanced.

In this paper, we propose to enhance efficiency of human pose estimation in videos by fully leveraging temporal cues to enable small networks to localize body joints accurately. Such an idea is motivated by observing the computational bottleneck for prior models. Considering the temporal consistency across adjacent frames, it is not necessary to pass every frame through a large network for feature extraction. Instead, the model only needs to learn how to effectively transfer knowledge of pose localization in previous frames to the subsequent frames. Such transfer can help alleviate

*This work was partly done while Xuecheng was an intern as Snap Inc.

the requirements of large models and reduce the overall computational cost.

To implement the above idea, we design a novel Dynamic Kernel Distillation (DKD) model. As shown in Fig. 1 (a), DKD online distills pose knowledge from the previous frame into pose kernels through a light-weight distillator. Then, DKD simplifies body joint localization into a matching procedure between the pose kernels and the current frame through simple convolution. In this way, DKD fast re-uses pose knowledge from one frame and provides compact guidance for a small network to learn discriminative features for accurate human pose estimation.

In particular, DKD introduces a light-weight CNN based pose kernel distillator. It takes features and pose estimations of the previous frame as input and infers pose kernels suitable for the current frame. These pose kernels carry knowledge of body joint configuration patterns from the previous frame to the current frame, and guide a small network to learn compact features matchable to the pose kernels for efficient pose estimation. Accordingly, body joint localization is cast as a matching procedure via applying pose kernels on feature maps output from small networks with simple convolution to search for regions with similar patterns. Since it gets rid of the need for using large networks, DKD performs significantly faster than prior models. In addition, this 2D convolution based matching scheme is significantly cheaper than additional optical flow [9], the decoding phase of an RNN unit [30] or the expensive 3D convolutions [16]. Moreover, the distillator framewisely updates the pose kernels according to current joint representations and configurations. This dynamic feature makes DKD more flexible and robust in analyzing various scenarios in videos.

To further leverage temporal cues to facilitate the distillator to infer suitable pose kernels, DKD introduces a temporally adversarial training method that adopts a discriminator to help estimate consistent poses in consecutive frames. The temporally adversarial discriminator learns to distinguish the groundtruth change of joint confidence maps over neighboring frames from the predicted change, and thus supervises DKD to generate temporally coherent poses. In contrast to previous adversarial training methods [6, 5] that learn structure priors in the spatial dimension for recognition over still images, our method constrains the pose variations in the temporal dimension of videos, enforcing plausible changes of estimated poses in videos. In addition, this discriminator can be removed during the inference phase, thus introducing no additional computation.

The whole framework of the proposed DKD model is end-to-end learnable. Comprehensive experiments on two widely used benchmarks Penn Action [33] and Sub-JHMDB [15] demonstrate the efficiency and effectiveness of our DKD model for resolving human pose estimation in videos. Our main contributions are in three folds: 1) We propose a novel model to facilitate small networks in video-based pose estimation with lifted efficiency, by using a light-weight distillator to online distill the pose knowledge and simplifying body joint localization into a matching procedure with simple convolution. 2) We introduce the first temporally adversarial training strategy for encouraging the coherence of estimated poses in the temporal dimension of videos. 3) Our model achieves outperforming efficiency, *i.e.* 10x flops reduction and 2x speedup over previous best model, also with state-of-the-art accuracy.

2. Related work

For human pose estimation in videos, existing CNN based methods [12, 11, 25, 20, 10] usually focus on leveraging temporal cues to extract complementary information for refining the preliminary results output from a large network for every frame. In [14], Iqbal *et al.* incorporate deep learned representations into an action conditioned pictorial structured model to refine pose estimation results of each frame. In [12] and [10], 3D convolutions are exploited on video clips for implicitly capturing the temporal contexts between frames. In [25], Song *et al.* propose a Thin-Slicing network that uses dense optical flow to warp and align heatmaps of neighboring frames and then performs spatial-temporal inference via message passing through the graph constructed by joint candidates and their relationships among aligned heatmaps. [11] and [20] sequentially estimate human poses in videos following the Encoder-RNN-Decoder framework. Given a frame, this kind of framework first uses an encoder network to learn high-level image representations, then RNN units to explicitly propagate temporal information between neighboring frames and produce hidden states, and finally a decoder network to take hidden states as input and output pose estimation results of current frame. For ensuring good performance, however, these methods always require large network to compactly learn intermediate representations or preliminary poses. Their efficiency is rather limited.

Different from existing methods, our DKD model distills coherent pose knowledge from temporal cues and simplifies body joint localization as a matching problem, thus allowing small networks to accurately and efficiently estimate human poses in videos, which is explained in more detail below.

3. Proposed approach

3.1. Formulation

We first mathematically formulate the proposed Dynamic Kernel Distillation (DKD) model for human pose estimation in videos. For a video $\mathcal{V}=\{I_t\}_{t=1}^{T}$ including T frames, we use $I_t \in \mathbb{R}^{M\times N\times 3}$ to denote its tth frame, where M and N are the height and width of I_t, respectively. DKD aims to estimate a set of confidence maps $\mathcal{H}=\{h_t\}_{t=1}^{T}$ for all frames in $\mathcal{V}$. The $h_t \in \mathbb{R}^{m\times n\times K}$ is of spatial size $m\times n$, where K

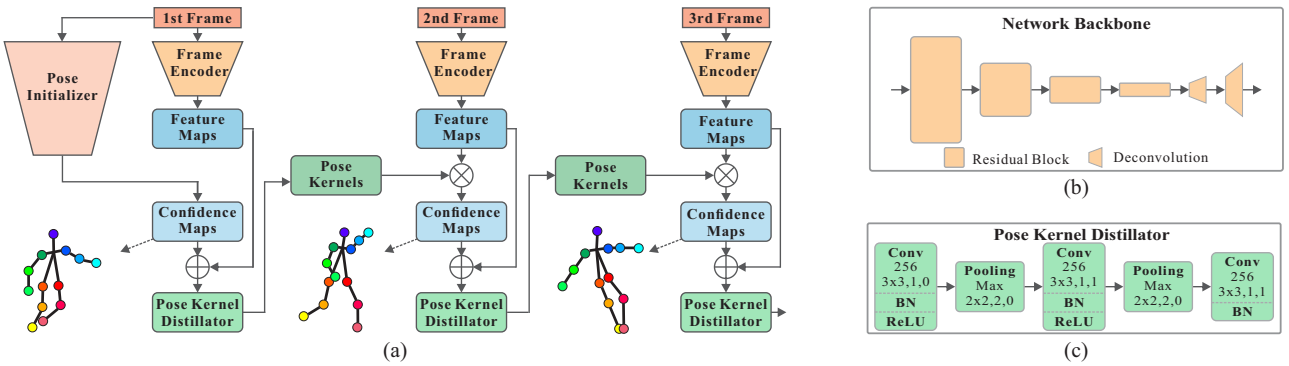


Figure 2. The architecture of the proposed Dynamic Kernel Distillation model. (a) The overall framework of the DKD model for inferencing human poses in videos. $\otimes$ denotes the convolution operation and $\oplus$ the concatenation. (b) The network backbone utilized in the pose initializer and frame encoder. (c) The network architecture of the pose kernel distillator.

is the number of body joints, and each of its elements encodes the confidence of a joint at the corresponding position. Accordingly, DKD performs online human pose estimation frame-by-frame in a sequential manner, by leveraging temporal cues between neighboring frames. In particular, its core is composed of a pose kernel distillator with a temporally adversarial training strategy.

Pose kernel distillation Given a frame I_t, DKD introduces a pose kernel distillator $\Phi(\cdot)$ to transfer pose knowledge provided by I_t to guide pose estimation in the next frame I_{t+1}. In particular, it leverages temporal cues represented with the combination of feature maps f_t and confidence maps h_t, to *online* distill pose kernels k_t via a simple feed-forward computation

$$k_t = \Phi(f_t, h_t), \tag{1}$$

where $k_t \in \mathbb{R}^{S\times S\times C\times K}$ and S is the kernel size. The distilled pose kernels k_t encode knowledge of body joint patterns and provide compact guidance for pose estimation in the posterior frame, which is learnable with light-weight networks. Accordingly, DKD exploits a small frame encoder $\mathrm{F}(\cdot)$ to learn high-level image representations f_{t+1} of frame I_{t+1} to match these distilled pose kernels, alleviating the demand of large networks that troubles prior works [25, 20]. Then, DKD applies the distilled pose kernels k_t on feature maps f_{t+1}, in a sliding window manner to search for the region with similar patterns as each body joint, namely,

$$h^j_{t+1} = k^j_t \otimes f_{t+1}, \tag{2}$$

where $\otimes$ denotes the convolution operation, and $h^j_{t+1} \in \mathbb{R}^{m\times n}$, $k^j_t \in \mathbb{R}^{S\times S\times C}$ are the confidence map and pose kernels of the jth joint, respectively. With the above formulation, DKD casts human pose estimation to a matching problem and locates the position with maximum response on h^j_{t+1} in the $(t+1)$th frame as the jth body joint.

In this way, the pose kernel distillator equips DKD with the capability of transferring pose knowledge among neighboring frames and enables small network to estimate human pose in videos. Its distilled pose kernels can be applied to fast localize body joints with simple convolution, further improving the efficiency. In addition, it can directly leverage temporal cues of one frame to assist body joint localization in the following frame, without requiring auxiliary optical flow models [25] or decoders appended to RNN units [20]. It can also fast distill pose kernels in a one-shot manner, avoiding complex iterating utilized by previous online kernel learning models [4, 27]. Moreover, it framewisely updates pose kernels and improves the robustness of our model to joint appearance and configuration variations.

It is worth noting that, for the first frame, due to the lack of preceding temporal cues, we utilize another pose model $\mathrm{P}(\cdot)$, usually larger than $\mathrm{F}(\cdot)$, to initialize its confidence map, *i.e.*, $h_1 = \mathrm{P}(I_1)$. In particular, $\Phi(\cdot)$ together with $\mathrm{F}(\cdot)$ and $\mathrm{P}(\cdot)$ instantiate the *pose generator*. Given pose annotations $\{\hat{h}_t\}_{t=1}^T$, to learn the pose generator, we define the loss as

$$\mathcal{L}_{\mathrm{G}} = \sum_{t=1}^{T} \ell_2(h_t, \hat{h}_t), \tag{3}$$

where ℓ_2 denotes the Mean Square Error loss.

Temporally adversarial training To further leverage temporal cues, DKD adopts the adversarial training strategy to learn proper supervision in the temporal dimension for improving the pose kernel distillator. Adversarial training was only exploited for images in the spatial dimension in prior works [6, 5]. In contrast, our proposed temporally adversarial training strategy aims to provide constraints for pose changes in the temporal dimension, helping estimate coherent human poses in consecutive frames of videos. Inspired by [6], DKD introduces a discriminator $\mathrm{D}(\cdot)$ to distinguish the changes of groundtruth confidence maps between neighboring frames from predicted ones. The discriminator $\mathrm{D}(\cdot)$ takes as input two neighboring confidence maps (either from groundtruth or prediction) concatenated with the corresponding images, and reconstructs the change of the confidence maps. For real (groundtruth) samples $\hat{h}_t$ and

$\hat{h}_{t+1}$, the discriminator $\mathrm{D}(\cdot)$ targets at approaching their change $\hat{d}_t{=}\hat{h}_{t+1}{-}\hat{h}_t$, while for fake (predicted) samples h_t and h_{t+1}, keeping the reconstructed change away from $d_t{=}h_{t+1}{-}h_t$. Therefore, the discriminator can better differentiate groundtruth change from erroneous predictions. In this way, the discriminator $\mathrm{D}(\cdot)$ criticizes pixel-wise variations of confidence maps and judges whether joint positions are in rational movements, to encourage the pose kernel distillator to distill suitable pose kernels and ensure consistency of estimated poses between neighboring frames. To train the discriminator $\mathrm{D}(\cdot)$, we define its loss function as

$$\mathcal{L}_{\mathrm{D}}{=}\lambda \sum_{t=1}^{T-1} \ell_2(d_t^{\mathrm{f}}, d_t) - \sum_{t=1}^{T-1} \ell_2(d_t^{\mathrm{r}}, \hat{d}_t), \tag{4}$$

where $d_t^{\mathrm{r}}{=}\mathrm{D}(I_t, \hat{h}_t, I_{t+1}, \hat{h}_{t+1})$ denotes the output from the discriminator for real samples and $d_t^{\mathrm{f}}{=}\mathrm{D}(I_t, h_t, I_{t+1}, h_{t+1})$ denotes the one for fake samples. λ is a variable for dynamically balancing the relative learning speed between the pose generator and temporally adversarial discriminator.

The temporally adversarial training conventionally follows a two-player minmax game. Therefore, the final objective function of the DKD model is written as

$$\min_{\mathrm{P},\mathrm{F},\Phi} \max_{\mathrm{D}} \mathcal{L}_{\mathrm{G}} + \eta \mathcal{L}_{\mathrm{D}}, \tag{5}$$

where η is a constant for weighting generator loss and discriminator loss, set as 0.1. The training process to optimize the above object function will be illustrated in Section 3.3.

3.2. Network architecture

Pose initializer For the first frame I_1, DKD utilizes a pose initializer $\mathrm{P}(\cdot)$ to directly estimate its confidence maps h_1. Here, $\mathrm{P}(\cdot)$ exploits the network following [29], which achieves outstanding performance with a simple architecture. The network follows a U-shape architecture. It first encodes down-sized feature maps from the input image, and then gradually recovers high-resolution feature maps by appending several deconvolution layers, as shown in Fig. 2 (b). In particular, we use ResNet [13] as the backbone and append two deconvolution layers, resulting in a total stride of the network of 8. The other settings follow [29].

Frame encoder DKD utilizes an encoder $\mathrm{F}(\cdot)$ to extract high-level features f_t of frame I_t to match the pose kernels from the pose kernel distillator. Here, we design $\mathrm{F}(\cdot)$ with the same network architecture as the pose initializer $\mathrm{P}(\cdot)$, with only the last classification layer removed from $\mathrm{P}(\cdot)$. Note, the backbone of $\mathrm{F}(\cdot)$ is much smaller than $\mathrm{P}(\cdot)$.

Pose kernel distillator The pose kernel distillator $\Phi(\cdot)$ in DKD takes as input the temporal information, represented by the concatenation of feature maps f_t and confidence maps h_t, and distills the pose kernels k_t in a one-shot feed-forward manner. We implement $\Phi(\cdot)$ with a CNN, including three convolution layers followed by BatchNorm and ReLU layers and two pooling layers. Its architecture is shown in Fig. 2 (c). This light-weight CNN guarantees the efficiency of $\Phi(\cdot)$. However, it is inefficient and infeasible for $\Phi(\cdot)$ to directly learn all kernels $k_t{\in}\mathbb{R}^{S\times S\times C\times K}$ due to their large scale which brings high computational complexity and also the risk of overfitting. To avoid these issues, inspired by [3], DKD exploits $\Phi(\cdot)$ to learn the kernel bases k'_t instead of full size k_t via performing the following factorization:

$$k_t = U \otimes k'_t \otimes_{\mathrm{C}} V, \tag{6}$$

where $\otimes$ is the convolution operation, $\otimes_{\mathrm{C}}$ the channel-wise convolution, and $U{\in}\mathbb{R}^{1\times 1\times C\times K}$, $V{\in}\mathbb{R}^{1\times 1\times C\times C}$ are coefficients over the kernel bases $k'_t{\in}\mathbb{R}^{S\times S\times C}$. In this way, the size of actual outputs k'_t from the pose kernel distillator is smaller than original k_t by a magnitude, thus enhancing the efficiency of the DKD model.

To generate the confidence maps h_{t+1} of I_{t+1}, the calculation between k_t and f_{t+1} is implemented with convolution layers. In particular, we first use a $1{\times}1$ convolution parameterized by V on f_{t+1}. Then we apply k'_t in a dynamic convolution layer [21], which is the same with traditional convolution layer, just replacing the pre-learned static convolution kernels with the dynamically learned ones. Finally, we adopt another $1{\times}1$ convolution with U to produce h_{t+1}. To scale the estimation results with the pose kernels, we add a BatchNorm layer in the last to facilitate the training.

Temporally adversarial discriminator DKD utilizes the temporally adversarial discriminator $\mathrm{D}(\cdot)$ to enhance the learning process of the pose kernel distillator with confidence map variations as auxiliary temporal supervision. We design $\mathrm{D}(\cdot)$ with the same network backbone as the frame encoder $\mathrm{F}(\cdot)$ to balance the learning capability between pose generator and discriminator.

3.3. Training and inference

In this subsection, we will explain the training and inference process of the DKD model for human pose estimation in videos. Specifically, DKD exploits a temporally adversarial training strategy. The discriminator is optimized via maximizing the loss function defined in Eqn. (5) for distinguishing the changes of groundtruth confidence maps from estimated ones between neighboring frames. On the other hand, the generator produces a set of confidence maps for consecutive frames in a video and meanwhile fools the discriminator via making the changes of estimated poses approach those of groundtruth ones. To synchronize the learning speed between generator and discriminator, we follow [2, 6] to update λ in Eqn. (4) for each iteration i:

$$\lambda_{i+1} = \lambda_i + \gamma\big(\sum_{t=1}^{T-1} \ell_2(d_t^{\mathrm{r}}, \hat{d}_t) - \sum_{t=1}^{T-1} \ell_2(d_t^{\mathrm{f}}, d_t)\big) \tag{7}$$

Algorithm 1: Training process for our DKD model.

input : video $\{I_t\}_{t=1}^{T}$, groundtruth $\{\hat{h}_t\}_{t=1}^{T}$, iteration number E
initialization: $\mathcal{L}_D \leftarrow 0$, $\mathcal{L}_G \leftarrow 0$
for *iteration* i, i=1 *to* E **do**
  Forward pose initializer $h_1 \leftarrow \mathrm{P}(I_1)$
  Update loss $\mathcal{L}_\mathrm{G} \leftarrow \ell_2(h_1, \hat{h}_1)$
  for *frame* t, t=1 *to* T **do**
    if t *equals 1* **then**
      Encode image representations $f_1 \leftarrow \mathrm{F}(I_1)$
    end
    else
      Forward discriminator $d^{\mathrm{r}}_{t-1} \leftarrow \mathrm{D}(I_{t-1}, \hat{h}_{t-1}, I_t, \hat{h}_t)$
      Update loss $\mathcal{L}_D \leftarrow \mathcal{L}_D - \ell_2(d^{\mathrm{r}}_{t-1}, \hat{d}_{t-1})$
      Update pose kernels $k_{t-1} \leftarrow \Phi(f_{t-1}, h_{t-1})$
      Encode image representations $f_t \leftarrow \mathrm{F}(I_t)$
      Estimate confidence map h_t with Eqn. (2)
      Update loss $\mathcal{L}_\mathrm{G} \leftarrow \mathcal{L}_\mathrm{G} + \ell_2(h_t, \hat{h}_t)$
      Forward discriminator $d^{\mathrm{f}}_{t-1} \leftarrow \mathrm{D}(I_{t-1}, h_{t-1}, I_t, h_t)$
      Update loss $\mathcal{L}_D \leftarrow \mathcal{L}_D + \lambda_i \ell_2(d^{\mathrm{f}}_{t-1}, d_{t-1})$
      Update loss $\mathcal{L}_\mathrm{G} \leftarrow \mathcal{L}_\mathrm{G} + \eta \ell_2(d^{\mathrm{f}}_{t-1}, d_{t-1})$
    end
  end
  Update discriminator $\mathrm{D}(\cdot)$ with $-\mathcal{L}_\mathrm{D}$ via backpropagation
  Update $\mathrm{P}(\cdot)$, $\Phi(\cdot)$, and $\mathrm{F}(\cdot)$ with $\mathcal{L}_\mathrm{G}$ via backpropagation
  Update λ_i with Eqn. (7)
end

where γ is a hyper-parameter controlling the update rate and set as 0.1. λ is initialized as 0 and bounded in $[0, 1]$. As defined in Eqn. (7), when the generator successfully fools the discriminator, λ will be increased to make the optimizer emphasize improving the discriminator, and vice versa. The overall training process is illustrated in Algorithm 1.

During inference, the discriminator $\mathrm{D}(\cdot)$ is removed. Given a video, DKD first utilizes the pose initializer $\mathrm{P}(\cdot)$ to estimate the confidence maps h_1 of the first frame. Then, h_1 is combined with the feature maps f_1 from the encoder $\mathrm{F}(\cdot)$ as input to the pose kernel distillator $\Phi(\cdot)$ for distilling the initial pose kernels k_1. For the second and subsequent frames, DKD applies the framewisely updated pose kernels k_t on the feature maps f_{t+1}=$\mathrm{F}(I_{t+1})$ of the posterior frame to estimate the confidence maps h_{t+1}. Finally, DKD outputs body joint positions for each frame by localizing the maximum responses on the corresponding confidence maps. The overall inference procedure of DKD is given in Fig. 2 (a).

4. Experiments

4.1. Experimental setup

Datasets We evaluate our model on two widely used benchmarks: Penn Action [33] and Sub-JHMDB [15]. Penn Action dataset is a large-scale unconstrained video dataset. It contains 2,326 video clips, 1,258 for training and 1,068 for testing. Each person in a frame is annotated with 13 body joints, including the coordinates and visibility. Following conventions, evaluations on the Penn Action dataset only consider the visible joints. Sub-JHMDB is another dataset for video based human pose estimation. It provides labels for 15 body joints. Different from Penn Action dataset, it only annotates visible joints for complete bodies. It contains 316 video clips with 11,200 frames in total. The ratio for the number of training and testing videos is roughly 3:1. In addition, it includes three different split schemes. Following previous works [20, 25], we separately conduct evaluations on these three splits and report the average precision.

Data augmentation For both the Penn Action dataset and Sub-JHMDB dataset, we perform data augmentation following conventional strategies, including random scaling with a factor from $[0.8, 1.4]$, random rotation in $[-40°, 40°]$ and random flipping. The same augmentation setting is applied to all the frames in a training video clip. In addition, each frame is cropped based on the person center on the original image and padded to 256×256 as input for training.

Implementation For fair comparison with previous works [20, 25], we first pre-train the pose initializer and the frame encoder for single-person pose estimation on the MPII [1] dataset. Then, we fine-tune the pre-trained models together with the randomly initialized pose kernel distillator and the temporally adversarial discriminator on Penn Action dataset and Sub-JHMDB dataset for 40 epochs, respectively. In particular, each training sample contains 5 frames, which are consecutively sampled from a video. We set the channel number C of the pose kernels k_t as 256 and the kernel size S as 7. We implement our DKD model with Pytorch [24] and use RMSprop as the optimizer [26]. We set the initial learning rate as 0.0005 and drop it with a multiplier 0.1 at the 15th and 25th epochs. For evaluation, we perform seven-scale testing with flipping.

Evaluation metrics We evaluate the performance with PCK [32]—the localization of a body joint is considered to be correct if it falls within $\alpha\cdot L$ pixels of the groundtruth. α controls the relative threshold and conventionally set as 0.2. L is the reference distance, set as $L=\max(H, W)$ following prior works [20, 25] with H and W being height and width of the person bounding box. We term this metric as PCK normalized by person size. This metric is somewhat loose to precisely evaluate the model performance as person size is usually relatively large. Thereby, we follow the conventions of still-image based pose estimation [17, 7, 31, 28], and also adopt another metric that takes torso size as reference distance. We term it as PCK normalized by torso size.

4.2. Ablation analysis

We first conduct ablation studies on Penn Action dataset to analyze the efficacy of each core component of our DKD model: the pose kernel distillator and the temporally adversarial training. We fix the backbone of the pose initializer as ResNet101. We vary the backbone of

Table 1. Ablation studies on Penn Action dataset with PCK normalized by torso size as evaluation metric.

Methods	Flops(G)	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	PCK
Baseline(ResNet101)	11.02	96.1	90.7	91.4	89.5	86.2	92.2	88.9	90.7
DKD(ResNet50)	8.65	96.6	93.7	92.9	91.2	88.8	94.3	93.7	92.9
DKD(ResNet50)-w/o-TAT	8.65	96.6	92.6	92.9	90.8	87.5	93.4	92.4	92.1
DKD(ResNet50)-w/o-PKD	7.66	96.0	91.8	92.4	90.4	88.3	93.5	89.8	91.6
Baseline(ResNet50)	7.66	96.0	90.5	89.4	87.6	83.8	89.7	86.0	88.8
DKD(ResNet34)	7.68	96.4	91.9	93.0	90.8	88.6	93.5	91.9	92.1
DKD(ResNet34)-w/o-TAT	7.68	96.4	91.2	92.7	89.9	87.3	93.3	90.9	91.4
DKD(ResNet34)-w/o-PKD	6.69	95.9	91.1	91.9	89.3	87.7	92.5	90.3	91.0
Baseline(ResNet34)	6.69	95.8	88.7	88.5	86.7	83.6	89.6	85.3	87.3
DKD(ResNet18)	5.27	95.7	90.0	92.2	89.4	86.8	92.3	89.5	90.6
DKD(ResNet18)-w/o-TAT	5.27	95.5	89.3	91.9	89.1	85.0	91.6	89.0	89.9
DKD(ResNet18)-w/o-PKD	4.28	95.0	89.1	92.4	88.7	85.5	91.4	87.7	89.7
Baseline(ResNet18)	4.28	94.7	86.0	87.7	84.6	81.1	87.4	84.3	86.1

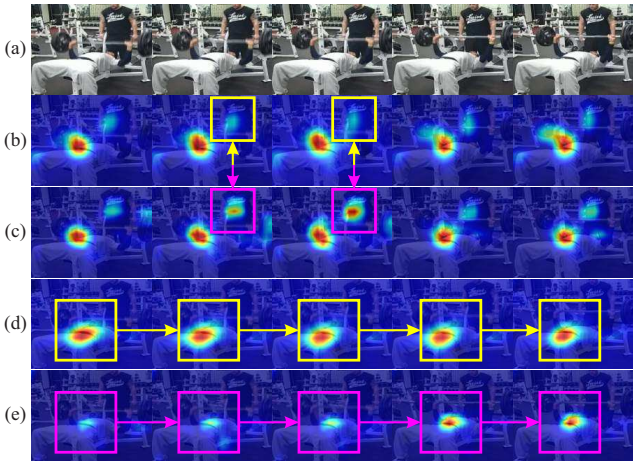


Figure 3. Comparison of confidence maps estimated from the proposed model DKD(ResNet34) and the baseline one Baseline(ResNet34). (a) are input frames. (b) and (d) are estimated confidence maps from our model for right elbow and right hip, respectively, and (c) and (e) from baseline. Best viewed in color.

frame encoder ranging in ResNet18/34/50, since it dominates the computational cost of pose estimation of our model. We use DKD(ResNet$x$) to denote our full model, where $x \in \{18, 34, 50\}$ represents the backbone depth of the frame encoder. We use DKD(ResNet$x$)-w/o-TAT to denote the model without the temporally adversarial training and DKD(ResNet$x$)-w/o-PKD the model without the pose kernel distillator. We use Baseline(ResNet$x$) to denote the single-image pose estimation model without using temporal cues. Results are shown in Tab. 1.

From Tab. 1, we can see that DKD(ResNet34) and DKD(ResNet50) use smaller networks for frame feature learning while achieve much better performance than Baseline(ResNet101) which is much deeper. We can also see DKD(ResNet18) achieves comparable performance to Baseline(ResNet101) (90.6% PCK vs 90.7% PCK), with up to $2\times$ flop reduction (5.27G vs 11.02G Flops). These results verify the efficacy of DKD to enable small networks to estimate human pose in videos, bring efficiency enhancement while achieving outperforming accuracy.

Table 2. Comparison of temporally *vs.* spatially adversarial training, and pose kernel distillator *vs.* Convolutional LSTM. The accuracy is measured with PCK normalized by torso size.

Methods	Flops(G)	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	PCK
DKD(ResNet34)	7.68	96.4	91.9	93.0	90.8	88.6	93.5	91.9	92.1
DKD(ResNet34)-w-SAT	7.68	96.4	91.4	92.8	90.1	87.7	93.4	91.2	91.6
DKD(ResNet34)-w-LSTM	10.16	95.7	89.5	92.9	90.2	86.9	93.5	90.1	91.1

By comparing the DKD(ResNet$x$)-w/o-TATs and the Baseline(ResNet$x$)s, we find that the computation overhead of the pose kernel distillator is small, only bringing slight flops increase, *e.g.*, with ResNet50 as backbone, from 7.66G to 8.65G. We can also find the pose kernel distillator improves frame-level performance for human pose estimation over baselines by 4.3% in average. Besides, DKD(ResNet$x$)-w/o-TATs always outperform DKD(ResNet$x$)-w/o-PKDs, this implies the distilled pose kernels carry knowledge of body joint patterns and provide compact guidance for pose estimation between neighboring frames, which are absent in still-image based inference. The above results verify the efficacy of the pose kernel distillator for efficiently transferring pose knowledge to assist poses estimation in videos.

By comparing the time cost of the DKD(ResNet$x$)-w/o-PKDs and the Baseline(ResNet$x$)s, we find temporally adversarial training does not hurt inference speed, since the discriminator is used only in training. In addition, the temporally adversarial training consistently improves the baseline performance for all body joints, in particular for the joints difficult to localize, *e.g.*, DKD(ResNet34)-w/o-PKD improves the accuracy of ankles from 85.3% PCK to 90.3% PCK. This demonstrates the proposed temporally adversarial training is effective for regularizing temporal changes over pose predictions during model training.

Combining temporally adversarial training with the pose kernel distillator, the full DKD model further boosts the performance over all the ablated models, showing they are complementary to each other. Especially, DKD(ResNet$x$)s achieves average 5.5% performance gain over the corresponding vanilla baselines Baseline(ResNet$x$)s.

To better reveal the advantages of our DKD model over single-frame based models, we visualize the confidence maps estimated from DKD(ResNet34) and Baseline (ResNet34) for the elbow and ankle in Fig. 3. By comparing Fig. 3 (b) and (c), we can observe that our DKD model produces pose kernels of the correct person of interest with more accurate response. In contrast, the baseline model produces false alarms on the elbow of another person in the frame. We can also see that the proposed model can produce consistent confidence maps for the hip in Fig. 3 (d) while the baseline model produces unstable estimations even with fixed hip Fig. 3 (e). These results further validate the capability of the proposed model for generating accurate and temporally consistent human pose estimations in videos.

Next, we analyze how well our pose kernel distillator

performs for propagating temporal information via comparing it with the state-of-the-art Convolutional LSTMs [20]. We also compare our temporally adversarial training with the spatially one in [6]. All the compared models adopt the ResNet101 as the backbone of the pose initializer and ResNet34 as the frame encoder. Except for the compared components, all the other settings are the same. Results are shown in Tab. 2. We use DKD(ResNet34)-w-LSTM to denote the model utilizing Convolutional LSTM for temporal cues propagation instead of our pose kernel distillator in the DKD model. We can observe that DKD(ResNet34)-w-LSTM degrades the accuracy of DKD(ResNet34) for all body joints, especially for wrist and ankle. In addition, it increases the flops from 7.68G to 10.16G. These results evaluate the superiority of the pose kernel distillator in both efficiency and efficacy for transferring pose knowledge between neighboring frames over traditional RNN units.

We use DKD(ResNet34)-w-SAT to denote the model in which our temporally adversarial training is replaced with the spatially one in [6]. Specifically, [6] introduces a discriminator to distinguish the single-frame groundtruth confidence maps from estimated ones for obtaining structural spatial constraints on poses. We can see DKD(ResNet34) consistently outperforms DKD(ResNet34)-w-SAT. In addition, by comparing DKD(ResNet34)-w-SAT with DKD(ResNet34)-w/o-TAT in Tab. 1, spatially adversarial training only brings limited improvement. These results further verify the efficacy of using adversarial training in temporal dimension.

4.3. Comparisons with state-of-the-arts

Tab. 3 show the comparisons of our DKD model with state-of-the-arts on Penn Action dataset. In particular, the method proposed in [20] follows the Encoder-RNNs-Decoder framework with Convolutional LSTMs, while [25] exploits optical flow models to align confidence maps of neighboring frames. We report the performance of our model with both person and torso size as reference distance under the PCK evaluation metric. For comparison with current best model [20], we report both its performance with PCK normalized by torso size, flops and running time[1]. For our DKD model, we fix the backbone of the pose initializer as ResNet101. We vary the backbone of frame encoder ranging in ResNet18/34/50. Since both of state-of-the-arts [20] and [25] use the same network as Convolutional Pose Machines (CPM) [28], we also experiment our DKD model with a frame encoder as a simplified version of CPM by replacing its kernels with size larger than 3 to 3×3 kernels, denoted as DKD(SmallCPM), to further verifying the efficacy of DKD to facilitate small networks in video-based pose estimation.

[1]We reproduce the results of [20] with PCK normalized by torso size via running the codes released by the authors on the repo: https://github.com/lawy623/LSTM_Pose_Machines. The running time is counted on GPU GTX 1080ti for both [20] and our model.

Table 3. Comparison with state-of-the-arts on Penn Action dataset.

Methods	Flops(G)	Time(ms)	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	PCK
			Normalized by Person Size							
Park *et al.* [23]	-	-	62.8	52.0	32.3	23.3	53.3	50.2	43.0	45.3
Nie *et al.* [22]	-	-	64.2	55.4	33.8	24.4	56.4	54.1	48.0	48.0
Iqal *et al.* [14]	-	-	89.1	86.4	73.9	73.0	85.3	79.9	80.3	81.1
Gkioxari *et al.* [11]	-	-	95.6	93.8	90.4	90.7	91.8	90.8	91.5	91.8
Song *et al.* [25]	-	-	98.0	97.3	95.1	94.7	97.1	97.1	96.9	96.5
Luo *et al.* [20]	70.98	25	**98.9**	98.6	96.6	96.6	98.2	**98.2**	**97.5**	97.7
DKD(SmallCPM)	9.96	12	98.4	97.3	96.1	95.5	97.0	97.3	96.6	96.8
DKD(ResNet50)	**8.65**	**11**	98.8	**98.7**	**96.8**	**97.0**	**98.2**	98.1	97.2	**97.8**
			Normalized by Torso Size							
Luo *et al.* [20]	70.98	25	96.0	93.6	92.4	91.1	88.3	94.2	93.5	92.6
DKD(SmallCPM)	9.96	12	96.0	93.5	92.0	90.6	87.8	94.0	93.1	92.4
DKD(ResNet50)	**8.65**	**11**	**96.6**	**93.7**	**92.9**	**91.2**	**88.8**	**94.3**	**93.7**	**92.9**

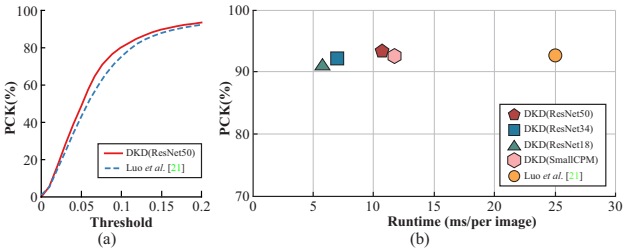
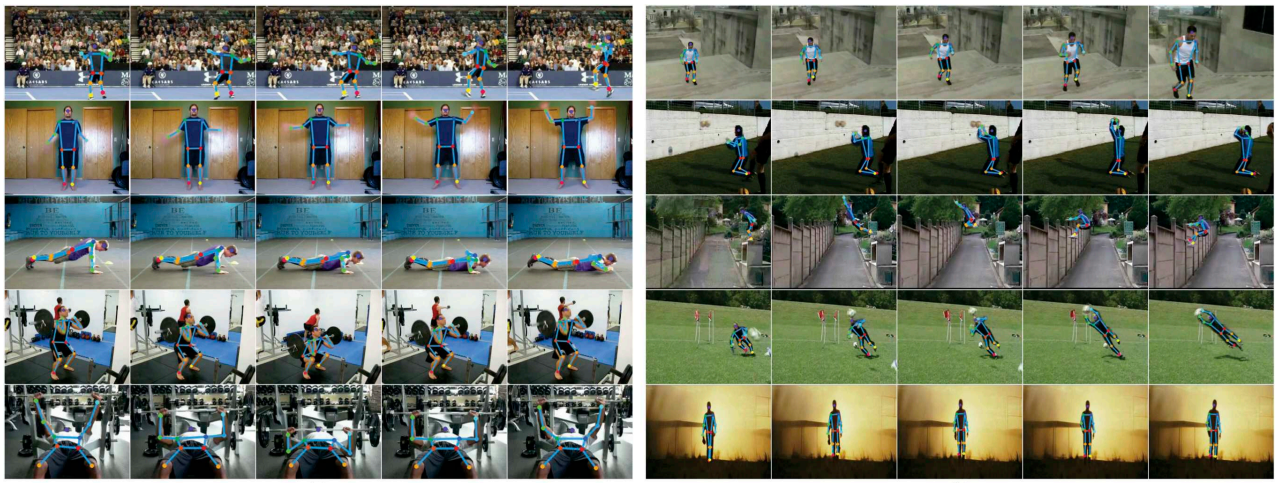


Figure 4. Extensive analysis for comparing our method with state-of-the-art [20] on (a) PCK over different thresholds with α ranging from 0 to 0.2; (b) speed *vs.* accuracy.

From Tab. 3, we can observe that our best model DKD(ResNet50) reduces the computation flops by a magnitude over [20] (8.65G vs 70.98G) and achieves 2x faster speed (11ms vs 25ms per image), verifying the outperforming efficiency of our model. In addition, we can see under PCK normalized by person size, DKD(ResNet50) achieves comparable accuracy with state-of-the-art [20]. When using PCK normalized by torso size, DKD(ResNet50) achieves superior accuracy over [20] (92.9% PCK vs 92.6% PCK) and with better performance for all of the body joints. We also compare our model with [20] via evaluating the performance with PCK normalized by torso size when varying threshold α from 0 to 0.2 with 0.01 as the step size, and results are shown in Fig. 4 (a). We can see that DKD consistently outperforms [20] under more critic metrics by decreasing α. These results demonstrate the superior speed and accuracy of our model for human pose estimation in videos.

By comparing DKD(SmallCPM) with [20], we can find our DKD model maintains high accuracy (92.4% PCK vs 92.6% PCK) in case of significant simplification to the network (9.96G vs 70.98G Flops). This result verifies the effectiveness of our DKD model for alleviating the demands of large networks for video-based human pose estimation.

To evaluate the effects of different frame encoder backbones on the efficiency and efficacy of DKD, we plot speed *vs.* accuracy analysis for different models in Fig. 4 (b). We can observe that reducing depth of frame encoder backbone from ResNet50 to ResNet18 slightly degrades the accuracy, but speeds up 2x from 11ms to 6.5ms per image. In addition, we can see that DKD(ResNet18) achieves comparable



(a) (b)

Figure 5. Qualitative results on (a) Penn Action dataset and (b) Sub-JHMDB dataset. Best viewed in color and 2x zoom.

Table 4. Comparison with state-of-the-arts on Sub-JHMDB dataset.

Methods	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	PCK
Normalized by Person Size								
Park *et al.* [23]	79.0	60.3	28.7	16.0	74.8	59.2	49.3	52.5
Nie *et al.* [22]	80.3	63.5	32.5	21.6	76.3	62.7	53.1	55.7
Iqal *et al.* [14]	90.3	76.9	59.3	55.0	85.9	76.4	73.0	73.8
Song *et al.* [25]	97.1	95.7	87.5	81.6	98.0	92.7	89.8	92.1
Luo *et al.* [20]	98.2	96.5	89.6	86.0	98.7	95.6	90.9	93.6
DKD(ResNet50)	**98.3**	**96.6**	**90.4**	**87.1**	**99.1**	**96.0**	**92.9**	**94.0**
Normalized by Torso Size								
Luo *et al.* [20]	92.7	75.6	66.8	64.8	78.0	73.1	73.3	73.6
DKD(ResNet50)	**94.4**	**78.9**	**69.8**	**67.6**	**81.8**	**79.0**	**78.8**	**77.4**

performance with [20] but 4x faster. These results further validate the efficacy of our DKD model to facilitate small networks in video-based pose estimation.

Tab. 4 show the comparisons of our DKD model with state-of-the-arts on Sub-JHMDB dataset. We can see that our DKD model achieves new state-of-the-art 94.0% PCK and performs best for all the body joints. When using the stricter metric PCK normalized by torso size, the superiority of our model over [20] is more significant, achieving over 5% improvement (77.4% PCK vs 73.6% PCK) on average. In addition, we can find that our model well applies to small-scale datasets, such as Sub-JHMDB with only 316 videos. These small datasets are challenging since they provide only limited training samples, while in our DKD model, the one-shot pose kernel distillator is able to fast adapt pose kernels, without requiring a large number of training samples for iteratively tuning classifiers as in existing methods.

Qualitative results Fig. 5 shows the qualitative results to visualize efficacy of the DKD model for human pose estimation in videos on Penn Action and Sub-JHMDB, respectively. We can observe DKD can accurately estimate human poses in various challenging scenarios, *e.g.*, cluttered backgrounds (the 1st row of Fig. 5 (a)), scale variations (the 1st row of Fig. 5 (b)), motion blur (the 2nd rows of Fig. 5 (a) and (b)). In addition, it can leverage temporal cues to handle occasional disappearance of a body joint caused by occlusion, as shown in the 3rd row of Fig. 5 (a), and encourage pose consistency in presence of fast and large-degree pose variations, as shown in the 3rd and 4th rows of Fig. 5 (b). Moreover, it is robust to various view-point and lighting conditions, as shown in the 5th rows of Fig. 5 (a) and (b), respectively. These results further verify the effectiveness of DKD.

5. Conclusion

This paper presents a Dynamic Kernel Distillation (DKD) model for improving efficiency of human pose estimation in videos. In particular, it adopts a pose kernel distillator to online distill the pose kernels from temporal cues of one frame in a one-shot feed-forward manner. The distilled pose kernels encode knowledge of body joint patterns and provide compact guidance for pose estimation in the posterior frame. With these pose kernels, DKD simplifies body joint localization into a matching procedure with simple convolution. In this way, DKD fast transfers pose knowledge between neighboring frames and enables small networks to accurately estimate human poses in videos, thus significantly lifting the efficiency. DKD also introduces the temporally adversarial training strategy via constraining the changes of estimated confidence maps between neighboring frames. The whole framework can be end-to-end trained and inferred. Experiments on two benchmarks demonstrate that our model achieves state-of-the-art efficiency with only 1/10 flops and 2x faster speed of the previous best model, and also outperforming accuracy for human pose estimation in videos.

Acknowledgement

Jiashi Feng was partially supported by NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014.

[2] David Berthelot, Thomas Schumm, and Luke Metz. Began: boundary equilibrium generative adversarial networks. *arXiv:1703.10717*, 2017.

[3] Luca Bertinetto, João F Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi. Learning feed-forward one-shot learners. In *NIPS*, 2016.

[4] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCV Workshop*, 2016.

[5] Yu Chen, Chunhua Shen, Xiu-Shen Wei, Lingqiao Liu, and Jian Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation. In *ICCV*, 2017.

[6] Chia-Jung Chou, Jui-Ting Chien, and Hwann-Tzong Chen. Self adversarial training for human pose estimation. In *CVPR Workshop*, 2017.

[7] Xiao Chu, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Structured feature learning for pose estimation. In *CVPR*, 2016.

[8] Marco Cristani, Ramya Raghavendra, Alessio Del Bue, and Vittorio Murino. Human behavior analysis in video surveillance: A social signal processing perspective. *Neurocomputing*, 100:86–97, 2013.

[9] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015.

[10] Rohit Girdhar, Georgia Gkioxari, Lorenzo Torresani, Manohar Paluri, and Du Tran. Detect-and-track: Efficient pose estimation in videos. In *CVPR*, 2018.

[11] Georgia Gkioxari, Alexander Toshev, and Navdeep Jaitly. Chained predictions using convolutional neural networks. In *ECCV*, 2016.

[12] Agne Grinciunaite, Amogh Gudi, Emrah Tasli, and Marten den Uyl. Human pose estimation in space and time using 3d cnn. In *ECCV Workshops*, 2016.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[14] Umar Iqbal, Martin Garbade, and Juergen Gall. Pose for action-action for pose. In *FG*, 2017.

[15] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black. Towards understanding action recognition. In *ICCV*, 2013.

[16] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Trans. on Pattern Anal. Mach. Intell.*, 35(1):221–231, 2013.

[17] Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. In *CVPR*, 2011.

[18] Jehee Lee, Jinxiang Chai, Paul SA Reitsma, Jessica K Hodgins, and Nancy S Pollard. Interactive control of avatars animated with human motion data. In *ACM Trans. on Graphics*, volume 21, pages 491–500, 2002.

[19] Huei-Yung Lin and Ting-Wen Chen. Augmented reality with human body interaction based on monocular 3d pose estimation. In *ACIVS*, 2010.

[20] Yue Luo, Jimmy Ren, Zhouxia Wang, Wenxiu Sun, Jinshan Pan, Jianbo Liu, Jiahao Pang, and Liang Lin. Lstm pose machines. In *CVPR*, 2018.

[21] Xuecheng Nie, Jiashi Feng, and Shuicheng Yan. Mutual learning to adapt for joint human parsing and pose estimation. In *ECCV*, 2018.

[22] Xiaohan Nie, Caiming Xiong, and Song-Chun Zhu. Joint action recognition and pose estimation from video. In *CVPR*, 2015.

[23] Dennis Park and Deva Ramanan. N-best maximal decoders for part models. In *ICCV*, 2011.

[24] Adam Paszke, Sam Gross, and Soumith Chintala. Pytorch, 2017.

[25] Jie Song, Limin Wang, Luc Van Gool, and Otmar Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. In *CVPR*, 2017.

[26] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 2012.

[27] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, 2017.

[28] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*, 2016.

[29] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, 2018.

[30] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015.

[31] Wei Yang, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In *CVPR*, 2016.

[32] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Trans. on Pattern Anal. Mach. Intell.*, 35(12):2878–2890, 2013.

[33] Weiyu Zhang, Menglong Zhu, and Konstantinos G Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *ICCV*, 2013.