Pose for Action – Action for Pose

Umar Iqbal, Martin Garbade, and Juergen Gall
Computer Vision Group, University of Bonn, Germany
{uiqbal, garbade, gall}@iai.uni-bonn.de

***Abstract*— In this work we propose to utilize information about human actions to improve pose estimation in monocular videos. To this end, we present a pictorial structure model that exploits high-level information about activities to incorporate higher-order part dependencies by modeling action specific appearance models and pose priors. However, instead of using an additional expensive action recognition framework, the action priors are efficiently estimated by our pose estimation framework. This is achieved by starting with a uniform action prior and updating the action prior during pose estimation. We also show that learning the right amount of appearance sharing among action classes improves the pose estimation. We demonstrate the effectiveness of the proposed method on two challenging datasets for pose estimation and action recognition with over 80,000 test images.**[1]

I. Introduction

Human pose estimation from RGB images or videos is a challenging problem in computer vision, especially for realistic and unconstrained data taken from the Internet. Popular approaches for pose estimation [2], [5], [6], [18], [29], [38] adopt the pictorial structure (PS) model, which resembles the human skeleton and allows for efficient inference in case of tree structures [7], [8]. Even if they are trained discriminatively, PS models struggle to cope with the large variation of human pose and appearance. This problem can be addressed by conditioning the PS model on additional observations from the image. For instance, [18] detects poselets, which are examples of consistent appearance and body part configurations, and condition the PS model on these.

Instead of conditioning the PS model on predicted configurations of body parts from an image, we propose to condition the PS model on high-level information like activity. Intuitively, the information about the activity of a person can provide a strong cue about the pose and vice versa the activity can be estimated from pose. There have been only few works [15], [39], [42] that couple action recognition and pose estimation to improve pose estimation. In [39], action class confidences are used to initialize an optimization scheme for estimating the parameters of a subject-specific 3D human model in indoor multi-view scenarios. In [42], a database of 3D poses is used to learn a cross-modality regression forest that predicts the 3D poses from a sequence of 2D poses, which are estimated by [38]. In addition, the action is detected and the 3D poses corresponding to the predicted action are used to refine the pose. However, both approaches cannot be applied to unconstrained monocular videos. While [39] requires a subject-specific model and several views, [42] requires 3D pose data for training. More recently, [15] proposed an approach to jointly estimate action classes and refine human poses. The approach decomposes the human poses estimated at each video frame into sub-parts and tracks these sub-parts across time according to the parameters learned for each action. The action class and joint locations corresponding to the best part-tracks are selected as estimates for the action class and poses. The estimation of activities, however, comes at high computational cost since the videos are pre-processed by several approaches, one for pose estimation [16] and two for extracting action related features [32], [34].

In this work, we present a framework for pose estimation that infers and integrates activities with a very small computational overhead compared to an approach that estimates the pose only. This is achieved by an action conditioned pictorial structure (ACPS) model for 2D human pose estimation that incorporates priors over activities. The framework of the approach is illustrated in Figure 1. We first infer the poses for each frame with a uniform distribution over actions. While the binaries of the ACPS are modeled by Gaussian mixture models, which depend on the prior distribution over the action classes, the unaries of the ACPS model are estimated by action conditioned regression forests. To this end, we modify the approach [5], which consists of two layers of random forests, on two counts. Firstly, we replace the first layer by a convolutional network and use convolutional channel features to train the second layer, which consists of regression forests. Secondly, we condition the regression forests on a distribution over actions and learn the sharing among action classes. In our experiments, we show that these modifications increase the pose estimation accuracy by more than 40% compared to [5]. After the poses are inferred with a uniform distribution over actions, we update the action prior and the ACPS model based on the inferred poses to obtain the final pose estimates. Since the update procedure is very efficient, we avoid the computational expensive overhead of [15].

We evaluate our approach on the challenging J-HMDB [13] and Penn-Action [44] datasets, which consist of videos collected from the Internet and contain large amount of scale and appearance variations. In our experiments, we provide a detailed analysis of the impact of conditioning unaries and binaries on a distribution over actions and the benefit of appearance sharing among action classes. We demonstrate the effectiveness of the proposed approach for pose estimation and action recognition on both datasets.

[1]The models and source code are available at http://pages.iai.uni-bonn.de/iqbal_umar/action4pose/.

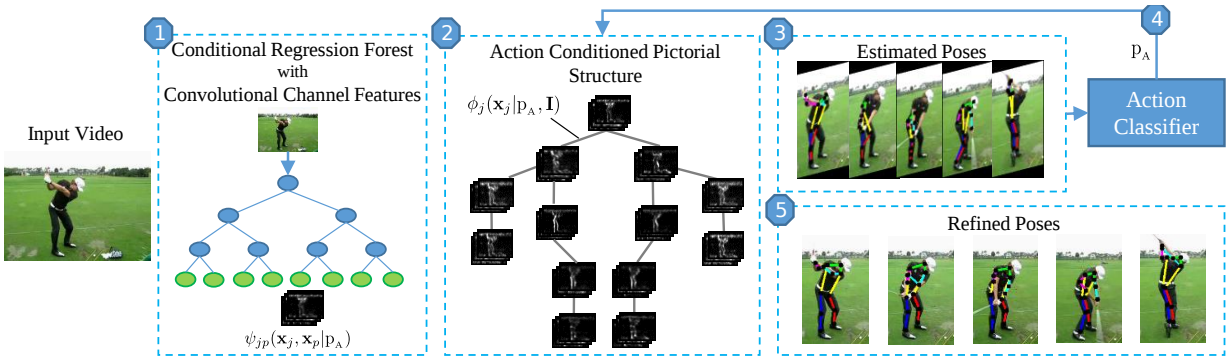

Fig. 1: Overview of the proposed framework. We propose an action conditioned pictorial structure model for human pose estimation (2). Both the unaries ϕ and the binaries ψ of the model are conditioned on the distribution of action classes $\mathrm{p_A}$. While the pairwise terms are modeled by Gaussians conditioned on $\mathrm{p_A}$, the unaries are learned by a regression forest conditioned on $\mathrm{p_A}$ (1). Given an input video, we do not have any prior knowledge about the action and use a uniform prior $\mathrm{p_A}$. We then predict the pose for each frame independently (3). Based on the estimated poses, the probabilities of the action classes $\mathrm{p_A}$ are estimated for the entire video (4). Pose estimation is repeated with the updated action prior $\mathrm{p_A}$ to obtain better pose estimates (5).

Compared to [15], the pose estimation accuracy is improved by over 30%.

II. Related Work

State-of-the-art approaches for pose estimation are mostly based on neural networks [10], [17], [20], [21], [28], [30], [35] or on the pictorial structure framework [5], [18], [38], [43].

Several approaches have been proposed to improve the accuracy of PS models for human pose estimation. For instance, joint dependencies can be modeled not only by the PS model, but also by a mid-level image representation such as poselets [18], exemplars [23] or data dependent probabilities learned by a neural network [1]. Pose estimation in videos can be improved by taking temporal information or motion cues into account [2], [10], [12], [14], [16], [45]. In [16] several pose hypotheses are generated for each video frame and a smooth configuration of poses over time is selected from all hypotheses. Instead of complete articulated pose, [22] and [2] track individual body parts and regularize the trajectories of the body parts through the location of neighboring parts. Similar in spirit, the approach in [43] jointly tracks symmetric body parts in order to better incorporate spatio-temporal constraints, and also to avoid double-counting. Optical flow information has also been used to enhance detected poses at each video frame by analyzing body motion in adjacent frames [9], [45].

Recent approaches for human pose estimation use different CNN architectures to directly obtain the heatmaps of body parts [10], [17], [20], [21], [28], [35]. [20], [28], [35] and [21] use fully convolutional neural network architectures, where [35] proposes a multi-staged architecture that sequentially refines the output in each stage. For pose estimation in videos, [17] combines the heatmaps of body parts from multiple frames with optical flow information to leverage the temporal information in videos. More recently, [10] proposes a convolutional recurrent neural network architecture that takes as input a set of video frames and sequentially estimates the body part locations in each frame, while also using the information of estimated body parts in previous frames.

As done in this work, a few works also combine both PS model and CNNs [1], [29]. In contrast to the approaches that use temporal information in videos for pose refinement, we utilize the detected poses in each video frame to extract high-level information about the activity and use it to refine the poses.

Action recognition based on 3D human poses has been investigated in many works [41]. With the progress in the area of 2D human pose estimation in recent years, 2D poses have also gained an increased attention for action recognition [3], [6], [13], [19], [26], [37]. However, utilizing action recognition to aid human pose estimation is not well studied, in particular not in the context of 2D human pose estimation. There are only a few works [15], [31], [39], [40], [42] that showed the benefit of it. The approaches in [39], [42] rely on strong assumption. The approach [39] assumes that a person-specific 3D model is given and considers pose estimation in the context of multiple synchronized camera views. The approach [42] focuses on 3D pose estimation from monocular videos and assumes that 3D pose data is available for all actions. The approach [31] adopts a mixture of PS models, and learns a model for each action class. For a given image, each model is weighted by the confidence scores of an additional action recognition system and the pose with the maximum weight is taken. A similar approach is adopted in [40] to model object-pose relations. These approaches, however, do not scale with the number of action classes since each model needs to be evaluated.

The closest to our work is the recent approach of [15] that jointly estimates the action classes and refines human poses. The approach first estimates human poses at each video frame and decomposes them into sub-parts. These sub-parts are then tracked across video frames based on action specific spatio-temporal constraints. Finally, the action labels and joint locations are inferred from the part tracks that

maximize a defined objective function. While the approach shows promising results, it does not re-estimate the parts but only re-combines them over frames *i. e.*, only the temporal constraints are influenced by an activity. Moreover, it relies on two additional activity recognition approaches based on optical flow and appearance features to obtain good action recognition accuracy that results in a very large computational overhead as compared to an approach that estimates activities using only the pose information. In this work, we show that additional action recognition approaches are not required, but predict the activities directly from a sequence of poses. In contrast to [15], we condition the pose model itself on activities and re-estimate the entire pose per frame.

III. Overview

Our method exploits the fact that the information about the activity of a subject provides a cue about pose and appearance of the subject, and vice versa. In this work we utilize the high-level information about a person's activity to leverage the performance of pose estimation, where the activity information is obtained from previously inferred poses. To this end, we propose an action conditioned pictorial structure (PS) that incorporates action specific appearance and kinematic models. If we have only a uniform prior over the action classes, the model is a standard PS model, which we will briefly discuss in Section IV. Figure 1 depicts an overview of the proposed framework.

IV. Pictorial Structure

We adopt the joint representation [5] of the PS model [8], where the vector $\mathbf{x}_j \in \mathcal{X}$ represents the 2D location of the j^{th} joint in image $\mathbf{I}$, and $\mathcal{X} = \{\mathbf{x}_j\}_{j\in\mathcal{J}}$ is the set of all body joints. The structure of a human body is represented by a kinematic tree with nodes of the tree being the joints j and edges $\mathcal{E}$ being the kinematic constraints between a joint j and its unique parent joint p as illustrated in Figure 1. The pose configuration in a single image is then inferred by maximizing the following posterior distribution:

$$\mathrm{p}(\mathcal{X}|\mathbf{I}) \propto \prod_{j\in\mathcal{J}} \phi_j(\mathbf{x}_j|\mathbf{I}) \prod_{j,p\in\mathcal{E}} \psi_{jp}(\mathbf{x}_j, \mathbf{x}_p) \qquad (1)$$

where the unary potentials $\phi_j(\mathbf{x}_j|\mathbf{I})$ represent the likelihood of the j^{th} joint at location $\mathbf{x}_j$. The binary potentials $\psi_{jp}(\mathbf{x}_j, \mathbf{x}_p)$ define the deformation cost for the joint-parent pair (j, p), and are often modeled by Gaussian distributions for an exact and efficient solution using a distance transform [8]. We describe the unary and binary terms in Section IV-A and Section IV-B, respectively. In Section V-A, we then discuss how these can be adapted to build an action conditioned PS model.

A. Unary Potentials

Random regression forests have been proven to be robust for the task of human pose estimation [5], [24], [27]. A regression forest $\mathcal{F}$ consists of a set of randomized regression trees, where each tree T is composed of split and leaf nodes. Each split node represents a weak classifier which passes an input image patch P to a subsequent left or right node until a leaf node L_T is reached. As in [5], we use a separate regression forest for each body joint. Each tree is trained with a set of randomly sampled images from the training data. The patches around the annotated joint locations are considered as foreground and all others as background. Each patch consists of a joint label $c \in \{0, j\}$, a set of image features F_P, and its 2D offset $\mathbf{d}_P$ from the joint center. During training, a splitting function is learned for each split node by randomly selecting and maximizing a goodness measure for regression or classification. At the leaves the class probabilities $\mathrm{p}(c|L_T)$ and the distribution of offset vectors $\mathrm{p}(\mathbf{d}|L_T)$ are stored.

During testing, patches are densely extracted from the input image $\mathbf{I}$ and are passed through the trained trees. Each patch centered at location $\mathbf{y}$ ends in a leaf node $L_T(P(\mathbf{y}))$ for each tree $T \in \mathcal{F}$. The unary potentials ϕ_j for the joint j at location $\mathbf{x}_j$ are then given by

$$\phi_j(\mathbf{x}_j|\mathbf{I}) = \sum_{\mathbf{y}\in\Omega} \frac{1}{|\mathcal{F}|} \sum_{T\in\mathcal{F}} \Big\{ \mathrm{p}(c = j|L_T(P(\mathbf{y}))) \cdot \mathrm{p}(\mathbf{x} - \mathbf{y}|L_T(P(\mathbf{y}))) \Big\}. \qquad (2)$$

In [5] a two layer approach is proposed. The first layer consists of classification forests that classify image patches according to the body parts using a combination of color features, HOG features, and the output of a skin color detector. The second layer consists of regression forests that predict the joint locations using the features of the first layer and the output of the first layer as features. For both layers, the split nodes compare feature values at different pixel locations within a patch of size 24×24 pixels.

We propose to replace the first layer by a convolutional network and extract convolutional channel features (CCF) [36] from the intermediate layers of the network to train the regression forests of the second layer. In [36] several pre-trained network architectures have been evaluated for pedestrian detection using boosting as classifier. The study shows that the "conv3-3" layer of the VGG-16 net [25] trained on the ImageNet (ILSVRC-2012) dataset performs very well even without fine tuning, but it is indicated that the optimal layer depends on the task. Instead of pre-selecting a layer, we use regression forests to select the features based on the layers "conv2-2", "conv3-3", "conv4-3", and "conv5-3". An example of the CCF extracted from an image is shown in Figure 2. Since these layers are of lower dimensions than the original image, we upsample them using linear interpolation to make their dimensions equivalent to the input image. This results in a 1408 (128+256+512+512) dimensional feature representation for each pixel. As split nodes in the regression forests, we use axis-aligned split functions. For an efficient feature extraction at multiple image scales, we use patchwork as proposed in [11] to perform the forward pass of the convolutional network only once.

B. Binary Potentials

Binary potentials $\psi_{jp}(\mathbf{x}_j, \mathbf{x}_p)$ are modeled as a Gaussian mixture model for each joint j with respect to its parent joint

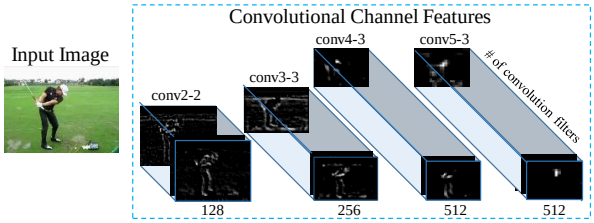

Fig. 2: Example of convolutional channel features extracted using VGG-16 net [25].

p in the kinematic tree. As in [5], we obtain the relative offsets between child and parent joints from the training data and cluster them into $k = 1, \ldots, K$ clusters using k-means clustering. Each cluster k takes the form of a weighted Gaussian distribution as

$$\psi_{jp}(\mathbf{x}_j, \mathbf{x}_p) = w_{jp}^k \cdot \exp\left(-\frac{1}{2}\left(\mathbf{d}_{jp} - \mu_{jp}^k\right)^T \left(\Sigma_{jp}^k\right)^{-1} \left(\mathbf{d}_{jp} - \mu_{jp}^k\right)\right) \quad (3)$$

with mean μ_{jp}^k and covariance Σ_{jp}^k, where $\mathbf{d}_{jp} = (\mathbf{x}_j - \mathbf{x}_p)$. The weights w_{jp}^k are set according to the cluster frequency $\mathrm{p}(k|j,p)^\alpha$ with a normalization constant $\alpha = 0.1$ [5].

For inference, we select the best cluster k for each joint by computing the max-marginals for the root node and backtrack the best pose configuration from the maximum of the max-marginals.

V. Action Conditioned Pose Estimation

As illustrated in Figure 1, our goal is to estimate the pose $\mathcal{X}$ conditioned by the distribution $\mathrm{p_A}$ for a set of action classes $a \in \mathcal{A}$. To this end, we introduce in Section V-A a pictorial structure model that is conditioned on $\mathrm{p_A}$. Since we do not assume any prior knowledge of the action, we estimate the pose first with the uniform distribution $\mathrm{p_A}(a) = 1/|\mathcal{A}|, \forall a \in \mathcal{A}$. The estimated poses for N frames are then used to estimate the probabilities of the action classes $\mathrm{p_A}(a|\mathcal{X}_{n=1\ldots N}), \forall a \in \mathcal{A}$ as described in Section V-B. Finally, the poses $\mathcal{X}_n$ are updated based on the distribution $\mathrm{p_A}$.

A. Action Conditioned Pictorial Structure

In order to integrate the distribution $\mathrm{p_A}$ of the action classes obtained from the action classifier into (1), we make the unaries and binaries dependent on $\mathrm{p_A}$:

$$\mathrm{p}(\mathcal{X}|\mathrm{p_A}, \mathbf{I}) \propto \prod_{j \in \mathcal{J}} \phi_j(\mathbf{x}_j|\mathrm{p_A}, \mathbf{I}) \cdot \prod_{j,p \in \mathcal{E}} \psi_{jp}(\mathbf{x}_j, \mathbf{x}_p|\mathrm{p_A}). \quad (4)$$

While the unary terms are discussed in Section V-A.1, the binaries $\psi_{jp}(\mathbf{x}_j, \mathbf{x}_p|\mathrm{p_A})$ are represented by Gaussians as in (3). However, instead of computing mean and covariance from all training poses with equal weights, we weight each training pose based on its action class label and $\mathrm{p_A}(a)$. In our experiments, we will also investigate the case where $\mathrm{p_A}(a)$ is simplified to

$$\mathrm{p_A}(a) = \begin{cases} 1 & \text{if} \quad a = \mathrm{argmax}_{a'}\, \mathrm{p_A}(a'|\mathcal{X}_{n=1\ldots N}) \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Fig. 3: Example patches centered at the wrist of the left hand side. We can see a large amount of appearance variation for a single body part. However, for several activities, in particular sports such as *golf* and *pull-up*, this variation is relatively small within the action classes. Nonetheless, a few classes also share appearance with each other e.g., *golf* and *baseball* or activities such as *run* and *kick ball*. This clearly shows the importance of class specific appearance models with a right amount of appearance sharing across action classes for efficient human pose estimation.

1) Conditional Joint Regressors: Figure 3 shows examples of patches of the wrist extracted from images of different action classes. We can see a large amount of appearance variation across different classes regardless of the fact that all patches belong to the same body joint. However, it can also be seen that within individual activities this variation is relatively small. We exploit this observation and propose action specific unary potentials for each joint j. To this end we adopt conditional regression forests [4], [27] that have been proven to be robust for facial landmark detection in [4] and 3D human pose estimation in [27]. While [4] trains a separate regression forest for each head pose and selects a specific regression forest conditioned on the output of a face pose detector, [27] proposes partially conditioned regression forests, where a forest is jointly trained for a set of discrete states of a human attribute like human orientation or height and the conditioning only happens at the leaf nodes. Since the continuous attributes are discretized, interpolation between the discrete states is achieved by sharing the votes.

In this work we resort to partially conditional forests due to its significantly reduced training time and memory requirements. During training we augment each patch P with its action class label a. Instead of $\mathrm{p}(c|L_T)$ and $\mathrm{p}(\mathbf{d}|L_T)$, the leaf nodes model the conditional probabilities $\mathrm{p}(c|a, L_T)$ and $\mathrm{p}(\mathbf{d}|a, L_T)$. Given the distribution over action classes $\mathrm{p_A}$, we obtain the conditional unary potentials:

$$\begin{aligned} \phi_j(\mathbf{x}_j|\mathrm{p_A}, \mathbf{I}) &= \sum_{\mathbf{y} \in \Omega} \frac{1}{|\mathcal{F}|} \sum_{T \in \mathcal{F}} \sum_{a \in \mathcal{A}} \Big\{ \mathrm{p_A}(a) \\ &\quad \cdot \mathrm{p}(c = j|a, L_T(P(\mathbf{y}))).\mathrm{p}(\mathbf{x} - \mathbf{y}|a, L_T(P(\mathbf{y})) \Big\} \\ &= \sum_{a \in \mathcal{A}} \phi_j(\mathbf{x}_j|a, \mathbf{I}) \mathrm{p_A}(a). \end{aligned} \quad (6)$$

Since the terms $\phi_j(\mathbf{x}_j|a,\mathbf{I})$ need to be computed only once for an image $\mathbf{I}$, $\phi_j(\mathbf{x}_j|\mathrm{p_A},\mathbf{I})$ can be efficiently computed after an update of $\mathrm{p_A}$.

2) Appearance Sharing Across Actions: Different actions sometimes share body pose configurations and appearance of parts as shown in Figure 3. We therefore propose to learn the sharing among action classes within a conditional regression forest. To this end, we replace the term $\phi_j(\mathbf{x}_j|a,\mathbf{I})$ in (6) by a weighted combination of the action classes:

$$\phi_j^{sharing}(\mathbf{x}_j|a,\mathbf{I}) = \sum_{a'\in\mathcal{A}} \gamma_a(a')\phi_j(\mathbf{x}_j|a',\mathbf{I}) \tag{7}$$

where the weights $\gamma_a(a')$ represent the amount of sharing between action class a and a'. To learn the weights γ_a for each class $a \in \mathcal{A}$, we apply the trained conditional regression forests to a set of validation images scaled to a constant body size and maximize the response of (7) at the true joint locations and minimize it at non-joint locations. Formally, this can be stated as

$$\gamma_a = \underset{\gamma}{\mathrm{argmax}} \sum_{n_a}\sum_{j} \left\{ \sum_{a'\in\mathcal{A}} \gamma(a')\phi_j^*\left(\mathbf{x}_{j,n_a}^{gt}|a',\mathbf{I}_{n_a}\right) - \max_{\mathbf{x}\in\mathbf{X}_{j,n_a}^{neg}} \left(\sum_{a'\in\mathcal{A}} \gamma(a')\phi_j^*\left(\mathbf{x}|a',\mathbf{I}_{n_a}\right)\right)\right\} - \lambda\|\gamma\|^2 \tag{8}$$

subject to $\sum_{a'\in\mathcal{A}}\gamma(a') = 1$ and $\gamma(a') \geq 0$. $\mathbf{I}_{n_a}$ denotes the n^{th} scaled validation image of action class a, $\mathbf{x}_{j,n_a}^{gt}$ is the annotated joint position for joint j in image $\mathbf{I}_{n_a}$, and $\mathbf{X}_{j,n_a}^{neg}$ is a set of image locations which are more than 5 pixels away from $\mathbf{x}_{j,n_a}^{gt}$. The set of negative samples is obtained by computing $\phi_j^*(\mathbf{x}|a',\mathbf{I}_{n_a})$ and taking the 10 strongest modes, which do not correspond to $\mathbf{x}_{j,n_a}^{gt}$, for each image. For optimization, we use the smoothed unaries

$$\phi_j^*(\mathbf{x}|a,\mathbf{I}) = \sum_{\mathbf{y}\in\Omega} \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\sigma^2}\right)\phi_j(\mathbf{y}|a,\mathbf{I}) \tag{9}$$

with $\sigma = 3$ and replace $\max$ by the softmax function to make (8) differentiable. The last term in (8) is a regularizer that prefers sharing, *i. e.*, $\|\gamma\|^2$ attains its minimum value for uniform weights. In our experiments, we use $\lambda = 0.4$ as weight for the regularizer. We optimize the objective function by constrained local optimization using uniform weights for initialization $\gamma(a') = 1/|\mathcal{A}|$. In our experiments, we observed that similar weights are obtained when the optimization is initialized by $\gamma(a) = 1$ and $\gamma(a') = 0$ for $a' \neq a$, indicating that the results are not sensitive to the initialization. In (8), we learn the weights γ_a for each action class but we could also optimize for each joint independently. In our experiments, however, we observed that this resulted in over-fitting.

B. Action Classification

For pose-based action recognition, we use the bag-of-word approach proposed in [13]. From the estimated joint positions $\mathcal{X}_{n=1\ldots N}$, a set of features called *NTraj+* is computed that encodes spatial and directional joint information. Additionally, differences between successive frames are computed to encode the dynamics of the joint movements. Since we use a body model with 13 joints, we compute the locations of missing joints (neck and belly) in order to obtain the same 15 joints as in [13]. We approximate the neck position as the mean of the face and the center of shoulder joints. The belly position is approximated by the mean of the shoulder and hip joints.

For each of the $3,223$ descriptor types, a codebook is generated by running k-means 8 times on all training samples and choosing the codebook with minimum compactness. These codebooks are used to extract a histogram for each descriptor type and video. For classification, we use an SVM classifier in a multi-channel setup. To this end, for each descriptor type t, we compute a distance matrix D_t that contains the χ^2-distance between the histograms (h_i^t, h_j^t) of all video pairs (v_i, v_j). We then obtain the kernel matrix that we use for the multi-class classification as follows

$$K(v_i,v_j) = \exp\left(-\frac{1}{L}\sum_{t=1}^{L}\frac{D_t(h_i^t,h_j^t)}{\mu^t}\right) \tag{10}$$

where L is the number of descriptor types and μ^t is the mean of the distance matrix D_t. For classification we use a one-vs-all approach with $C = 100$ for the SVM.

VI. Experiments

In order to evaluate the proposed method, we follow the same protocol as proposed in [15]. In particular, we evaluate the proposed method on two challenging datasets, namely sub-J-HMDB [13] and the Penn-Action dataset [44]. Both datasets provide annotated 2D poses and activity labels for each video. They consist of videos collected from the Internet and contain large amount of scale and appearance variations, low resolution frames, occlusions and foreshortened body poses. This makes them very challenging for human pose estimation. While sub-J-HMDB [13] comprises videos from 12 action categories with fully visible persons, the Penn-Action dataset comprises videos from 15 action categories with a large amount of body part truncations. As in [15], we discard the activity class "playing guitar" since it does not contain any fully visible person. For testing on sub-J-HMDB, we follow the 3-fold cross validation protocol proposed by [13]. On average for each split, this includes 229 videos for training and 87 videos for testing with $8,124$ and $3,076$ frames, respectively. The Penn-Action dataset consists of $1,212$ videos for training and $1,020$ for testing with $85,325$ and $74,308$ frames, respectively. To evaluate the performance of pose estimation, we use the APK (Average Precision of Keypoints) metric [15], [38].

A. Implementation Details

For the Penn-Action dataset, we split the training images half and half into a training set and a validation set. Since the dataset sub-J-HMDB is smaller, we create a validation set by mirroring the training images. The training images are scaled such that the mean upper body size is 40 pixels. Each forest

Unary \ Binary	Indep.	Cond. (5)	Cond. ($\mathrm{p_A}$)
Indep. + CCF	51.5	53.8	51.0
Cond. (5) + CCF	48.9	49.9	48.4
Cond. (5) + AS + CCF	53.8	**55.3**	52.9
Cond. ($\mathrm{p_A}$) + CCF	52.3	53.1	52.0
Cond. ($\mathrm{p_A}$) + AS + CCF	53.4	**55.1**	52.5

(a)

Unary \ Binary	Indep.	Cond. (5)	Cond. ($\mathrm{p_A}$)
Indep.	36.7	38.5	36.7
Cond. (5)	29.3	32.5	29.7
Cond. (5) + AS	38.0	**39.6**	37.2
Cond. ($\mathrm{p_A}$)	37.0	39.0	36.8
Cond. ($\mathrm{p_A}$) + AS	38.0	**39.5**	37.3

(b)

TABLE II: Analysis of the proposed framework under different settings. Cond. (5) denotes if the action class probabilities $\mathrm{p_A}$ are replaced by (5). **(a)** using CCF features. **(b)** using features from [5]. (APK threshold: 0.1)

Features	
HOG, Color, Skin [5]	CCF
36.7	51.5

TABLE I: Comparison of the features used in [5] with the proposed convolutional channel features (CCF). APK with threshold 0.1 on split-1 of sub-J-HMDB.

consists of 20 trees, where 10 trees are trained on the training and 10 on the validation set, with a maximum depth of 20 and a minimum of 20 patches per leaf. We train each tree with $50,000$ positive and $50,000$ negative patches extracted from $5,000$ randomly selected images and generate $40,000$ split functions at each node. For the binary potentials, we use $k = 24$ mixtures per part.

For learning the appearance sharing among action classes (Section V-A.2) and training the action classifier (Section V-B), we use the 10 trees trained on the training set and apply them to the validation set. The action classifier and the sharing are then learned on the validation set.

For pose estimation, we create an image pyramid and perform inference at each scale independently. We then select the final pose from the scale with the highest posterior (4). In our experiments, we use 4 scales with scale factor 0.8. The evaluation of 260 trees (20 trees for each of the 13 joints) including feature extraction takes roughly 15 seconds on average.[2] Inference with the PS model for all 4 scales takes around 1 second. The action recognition with feature computation takes only 0.18 seconds per image and it does not increase the time for pose estimation substantially.

B. Pose Estimation

We first evaluate the impact of the convolutional channel features (CCF) for pose estimation on split-1 of sub-J-HMDB. The results in Table I show that the CCF outperform the combination of color features, HOG features, and the output of a skin color detector, which is used in [5].

In Table IIa we evaluate the proposed ACPS model under different settings on split-1 of sub-J-HMDB when using CCF features for joint regressors. We start with the first step of our framework where neither the unaries nor the binaries depend on the action classes. This is equivalent to the standard PS model described in Section IV, which achieves an average joint estimation accuracy of 51.5%. Given the estimated poses, the pose-based action recognition approach described in Section V-B achieves an action recognition accuracy of 66.3% for split-1.

Having estimated the action priors $\mathrm{p_A}$, we first evaluate action conditioned binary potentials while keeping the unary potentials as in the standard PS model. As described in Section V-A, we can use in our model the probabilities $\mathrm{p_A}$ or replace them by the distribution (5), which considers only the classified action class. The first setting is denoted by "Cond. ($\mathrm{p_A}$)" and the second by "Cond. (5)". It can be seen that the conditional binaries based on (5) already outperform the baseline by improving the accuracy from 51.5% to 53.8%. However, taking the priors from all classes slightly decreases the performance. This shows that conditioning the binary potentials on the most probable class is a better choice than using priors from all classes.

Secondly, we analyze how action conditioned unary potentials affect pose estimation. For the unaries, we have the same options "Cond. ($\mathrm{p_A}$)" and "Cond. (5)" as for the binaries. In addition, we can use appearance sharing as described in Section V-A.2, which is denoted by "AS". For all three binaries, the conditional unaries based on (5) decrease the performance. Since the conditional unaries based on (5) are specifically designed for each action class, they do not generalize well in case of a misclassified action class. However, adding appearance sharing to the conditional unaries boost the performance for both conditioned on (5) and $\mathrm{p_A}$. Adding appearance sharing outperforms all other unaries without appearance sharing, *i. e.*, conditional unaries, independent unaries and the unaries conditioned on $\mathrm{p_A}$. For all unaries, the binaries conditioned on (5) outperform the other binaries. This shows that appearance sharing and binaries conditioned on the most probable class performs best, which gives an improvement of the baseline from 51.5% to 55.3%.

In Table IIb, we also evaluate the proposed ACPS when using the weaker features from [5]. Although the accuracies as compared to CCF features are lower, the benefit of the proposed method remains the same. For the rest of this paper, we will use CCF for all our experiments.

In Table III we compare the proposed action conditioned PS model with other state-of-the-art approaches on all three splits of sub-J-HMDB. In particular, we provide a comparison with [1], [2], [5], [15], [16], [38]. The accuracies for the approaches [2], [15], [16], [38] are taken from [15]

[2]Measured on a 3.4GHz Intel processor using only one core with NVidia GeForce GTX 780 GPU. The image size for all videos in sub-J-HMDB is 320×240 pixels.

Method	Head	Sho	Elb	Wri	Hip	Knee	Ank	Avg thr.=0.2	Avg thr.=0.1
Cond(5) + AS U. & Cond(5) B. + CCF	90.3	76.9	59.3	55.0	85.9	76.4	73.0	73.8	51.6
Cond. (p_A) + AS U. & Cond(5) B. + CCF	90.1	76.7	59.2	54.7	85.6	76.2	72.9	73.6	51.2
Indep. U. & Indep. B. + CCF	88.1	76.3	57.0	49.2	85.0	75.4	71.7	71.8	48.7
State-of-the-art approaches									
Indep. U. & Indep. B. [5]	65.6	56.4	39.1	31.1	65.2	62.8	60.9	54.4	34.4
Yang & Ramanan [38]	73.8	57.5	30.7	22.1	69.9	58.2	48.9	51.6	—
Park & Ramanan [16]	79.0	60.3	28.7	16.0	74.8	59.2	49.3	52.5	—
Cherian *et al.* [2]	47.4	18.2	0.08	0.07	—	—	—	16.4	—
Nie *et al.* [15]	80.3	63.5	32.5	21.6	76.3	62.7	53.1	55.7	—
Chen & Yuille [1]	78.7	68.4	48.3	39.7	76.3	66.3	60.3	62.6	42.2

TABLE III: Comparison with the state-of-the-art on sub-J-HMDB using APK threshold 0.2. In the last column, the average accuracy for the threshold 0.1 is given.

Method	Head	Sho	Elb	Wri	Hip	Knee	Ank	Avg thr.=0.2	Avg thr.=0.1
Cond(5) + AS U. & Cond(5) B. + CCF	89.1	86.4	73.9	73.0	85.3	79.9	80.3	81.1	64.8
Indep. U. & Indep. B. + CCF	84.5	81.3	66.2	62.6	82.4	75.1	76.5	75.5	57.3
State-of-the-art approaches									
Yang & Ramanan [38]	57.9	51.3	30.1	21.4	52.6	49.7	46.2	44.2	—
Park & Ramanan [16]	62.8	52.0	32.3	23.3	53.3	50.2	43.0	45.3	—
Nie *et al.* [15]	64.2	55.4	33.8	24.4	56.4	54.1	48.0	48.0	—
Gkioxari *et al.* [10]	95.6	93.8	90.4	90.7	91.8	90.8	91.5	91.8	—

TABLE IV: Comparison with the state-of-the-art in terms of joint localization error on the Penn-Action dataset.

where the APK threshold 0.2 is used. We also evaluated the convolutional network based approach [1] using the publicly available source code trained on sub-J-HMDB. Our approach outperfroms the other methods by a margin, and notably improves wrist localization by more than 5% as compared to the baseline.

Table IV compares the proposed ACPS with the state-of-the-art on the Penn-Action dataset. The accuracies for the approaches [15], [16], [38] are taken from [15]. We can see that the proposed method improves the baseline from 75.5% to 81.1%, while improving the elbow and wrist localization accuracy by more than 7% and 10%, respectively. The proposed method also significantly outperforms other approaches Only the approach [10] achieves a higher accuracy than our method. [10], however, uses a better multi-staged CNN architecture as baseline compared to our network for computing CCF features. Since the gain of ACPS compared to our baseline even increases when better features are used as shown in Table IIa & Table IIb, we expect at least a similar performance gain when we use the baseline architecture from [10] for ACPS.

C. Action Recognition

In Table VI, we compare the action recognition accuracy obtained by our approach with state-of-the-approaches for action recognition. On sub-J-HMDB, the obtained accuracy using only pose as feature is comparable to the other approaches. Only the recent work [3] which combines pose, CNN, and motion features achieves a better action recognition accuracy. However, if we combine our pose-based action recognition with Fisher vector encoding of improved dense trajectories [33] using late fusion, we outperform other methods that also combine pose and appearance. The results are similar on the Penn-Action dataset.

Method	sub-J-HMDB	Penn-Action
Appearance features only		
Dense [13]	46.0%	—
IDT-FV [33]	60.9%	92.0%
Pose features only		
Pose [13]	54.1%	—
Pose (Ours)	61.5%	79.0%
Pose + Appearance features		
MST [34]	45.3%	74.0%
Pose + Dense [13]	52.9%	—
AOG [15]	61.2%	85.5%
P-CNN [3]	66.8%	—
Pose (Ours) + IDT-FV	74.6%	92.9%

TABLE VI: Comparison of action recognition accuracy with the state-of-the-art approaches on sub-J-HMDB and Penn-Action datasets.

In Table V, we report the effect of different action recognition approaches on pose estimation. We report the pose estimation accuracy for split-1 of sub-J-HMDB when the action classes are not inferred by our framework, but estimated using improved dense trajectories with Fisher vector encoding (IDT-FV) [33] or the fusion of our pose-based method and IDT-FV. Although the action recognition rate is higher when pose and IDT-FV are combined, the pose estimation accuracy is not improved. If the action classes are not predicted but are provided (GT), the accuracy improves slightly for sub-J-HMDB and from 64.8% to 68.1% for the Penn-Action dataset. We also experimented with several iterations in our framework, but the improvements compared to the achieved accuracy of 51.6% were not more than 0.1% on all three splits of sub-J-HMDB.

VII. Conclusion

In this paper, we have demonstrated that action recognition can be efficiently utilized to improve human pose estimation on realistic data. To this end, we presented a pictorial

	Indep. U. + Indep. B. + CCF	Cond. (5)+AS U. & Cond. (5) B. + CCF			
		Pose	IDT-FV [33]	Pose+IDT-FV	GT
sub-J-HMDB (split-1)	51.5	55.3 (56.2%)	52.6 (66.3%)	55.3 (76.4%)	55.9 (100%)
Penn-Action	57.3	64.8 (79.0%)	—	—	68.1 (100%)

TABLE V: Analysis of pose estimation accuracy with respect to action recognition accuracy. The values in the parentheses are the corresponding action recognition accuracies. (APK threshold: 0.1)

structure model that incorporates high-level activity information by conditioning the unaries and binaries on a prior distribution over action labels. Although the action priors can be estimated by an accurate, but expensive action recognition system, we have shown that the action priors can also be efficiently estimated during pose estimation without substantially increasing the computational time of the pose estimation. In our experiments, we thoroughly analyzed various combinations of unaries and binaries and showed that learning the right amount of appearance sharing among action classes improves the pose estimation accuracy. While we expect further improvements by using a more sophisticated CNN architecture as baseline and by including a temporal model, the proposed method has already shown its effectiveness on two challenging datasets for pose estimation and action recognition.

Acknowledgements: The work was partially supported by the ERC Starting Grant ARCA (677650).

REFERENCES

[1] X. Chen and A. L. Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*, 2014.
[2] A. Cherian, J. Mairal, K. Alahari, and C. Schmid. Mixing Body-Part Sequences for Human Pose Estimation. In *CVPR*, 2014.
[3] G. Chéron, I. Laptev, and C. Schmid. P-cnn: Pose-based cnn features for action recognition. In *CVPR*, 2015.
[4] M. Dantone, J. Gall, G. Fanelli, and L. Van Gool. Real-time facial feature detection using conditional regression forests. In *CVPR*, 2012.
[5] M. Dantone, C. Leistner, J. Gall, and L. Van Gool. Body parts dependent joint regressors for human pose estimation in still images. *TPAMI*, 2014.
[6] C. Desai and D. Ramanan. Detecting actions, poses, and objects with relational phraselets. In *ECCV*, 2012.
[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010.
[8] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 2005.
[9] K. Fragkiadaki, H. Hu, and J. Shi. Pose from flow and flow from pose. In *CVPR*, 2013.
[10] G. Gkioxari, A. Toshev, and N. Jaitly. Chained predictions using convolutional neural networks. In *ECCV*, 2016.
[11] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869*, 2014.
[12] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, 2016.
[13] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. Black. Towards understanding action recognition. In *ICCV*, 2013.
[14] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.
[15] B. X. Nie, C. Xiong, and S.-C. Zhu. Joint action recognition and pose estimation from video. In *CVPR*, 2015.
[16] D. Park and D. Ramanan. N-best maximal decoders for part models. In *ICCV*, 2011.
[17] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *ICCV*, 2015.
[18] L. Pishchulin, M. Andriluka, P. Gehler, and B. Schiele. Poselet conditioned pictorial structures. In *CVPR*, 2013.
[19] L. Pishchulin, M. Andriluka, and B. Schiele. Fine-grained activity recognition with holistic and pose based features. In *GCPR*, 2014.
[20] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, and B. Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *CVPR*, 2016.
[21] U. Rafi, J. Gall, and B. Leibe. An efficient convolutional network for human pose estimation. In *BMVC*, 2016.
[22] V. Ramakrishna, T. Kanade, and Y. Sheikh. Tracking human pose by tracking symmetric parts. In *CVPR*, 2013.
[23] B. Sapp, C. Jordan, and B. Taskar. Adaptive pose priors for pictorial structures. In *CVPR*, 2010.
[24] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, 2011.
[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.
[26] V. K. Singh and R. Nevatia. Action recognition in cluttered dynamic scenes using pose-specific part models. In *ICCV*, 2011.
[27] M. Sun, P. Kohli, and J. Shotton. Conditional regression forests for human pose estimation. In *CVPR*, 2012.
[28] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *CVPR*, 2015.
[29] J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014.
[30] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, 2014.
[31] N. Ukita. Iterative action and pose recognition using global-and-pose features and action-specific models. In *ICCV Workshop*, 2013.
[32] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action Recognition by Dense Trajectories. In *CVPR*, 2011.
[33] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.
[34] J. Wang, X. Nie, Y. Xia, Y. Wu, and S.-C. Zhu. Cross-view action modeling, learning, and recognition. In *CVPR*, 2014.
[35] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016.
[36] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Convolutional channel features for pedestrian, face and edge detection. In *ICCV*, 2015.
[37] W. Yang, Y. Wang, and G. Mori. Recognizing human actions from still images with latent poses. In *CVPR*, 2010.
[38] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *TPAMI*, 2013.
[39] A. Yao, J. Gall, and L. Van Gool. Coupled action recognition and pose estimation from multiple views. *IJCV*, 2012.
[40] B. Yao and L. Fei-Fei. Recognizing human-object interactions in still images by modeling the mutual context of objects and human poses. *TPAMI*, 2012.
[41] M. Ye, Q. Zhang, L. Wang, J. Zhu, R. Yang, and J. Gall. A survey on human motion analysis from depth data. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, 2013.
[42] T.-H. Yu, T.-K. Kim, and R. Cipolla. Unconstrained monocular 3d human pose estimation by action detection and cross-modality regression forest. In *CVPR*, 2013.
[43] D. Zhang and M. Shah. Human pose estimation in videos. In *ICCV*, 2015.
[44] W. Zhang, M. Zhu, and K. G. Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *ICCV*, 2013.
[45] S. Zuffi, J. Romero, C. Schmid, and M. J. Black. Estimating human pose with flowing puppets. In *ICCV*, 2013.