Supplemental Materials

Anonymous CVPR submission

Paper ID 481

1. Sample Data

We include some sample videos in our Multiview Action3D dataset in the supplemental materials, and the detetion results of the propsoed model.

The same action "pick up with one hand" taken from three different viewpoints. We can see that the spatio-temporal patterns of these three viewpoints differ significantly. This is a major challenge which we want to tackle in this paper. We showed that the proposed MST-AoG model can characterize the association of the spatio-temporal patterns across different viewpoints and achieve high accuracy and robustness for cross-view action recognition.

We have three experimental settings: cross-view setting uses training data and test data taken from different viewpoints; cross-subject setting uses training data and test data of different subjects; cross-environment setting uses training data and test data taken in different environments.

Our method not only classify action but also parse action according to the And-Or graph using DP algorithm. For each frame, a parse graph is derived naturally representing the action category, pose and view angle in current frame. With the progress of the video, the parse graph may change and different nodes will be selected. We include some deteciton and parsing results of the propsoed method in the demo video. The detetion windows are obtained by using temproal median filters with frame 5 on the detecion windows on all the frames. There are some failure parsing results due to big variation of appearance, but the classification of whole action is robust to those failures. Though currently we didn't evaluate parsing performance quantitatively, we plan to evaluate it in the future.

2. Inference in AND/OR Graphs

The inference of a MST-AoG model calculates the the states of the nodes and evaluates their scores to generate a parse graph, and uses the parse graph for action recognition. Since this MST-AoG model is tree-structured, we generate the parse graph using dynamic programming. The general dynamic programming process contains bottom-up phase and top-down phase, which is similar to sum-product and max-product algorithm in graphical model.

The bottom-up phase computes the node scores in a bottom-up manner. In the bottom-up phase, the score of an AND-node is computed according the scores of its children:

$$s(n_{AND}, t_{n_{AND}}) = \max_{t_{n_1},\cdots,t_{n_l}} S_{n_{AND}}(s(n_1, t_{n_1}), \cdots, s(n_l, t_{n_1}), t_{n_{AND}}) \quad (1)$$

where $S_{n_{AND}}$ is the score function of the node n_{AND}, t_{n_i} is the state of the node n_i. $s(n_i, t_i)$ is the score of the state t_i of a the node n_i, and $n_1, \cdots n_l$ are the children of n_{AND}.

The score of an OR-node is the maximum scores of its children for the given state:

$$s(n_{OR}, , t_{n_{OR}}) = \max_i S_{n_{OR}}(s(n_i, t_{n_i}), t_{n_{OR}}) \quad (2)$$

where $S_{n_{OR}}$ is the score function of the node n_{OR}, and $n_1, \cdots n_l$ are the children of n_{OR}.

We can compute the node scores bottom up from the leave nodes to the root node by iteratively applying Eq. (1) and Eq. (2).

The top-down phase computes the node states by passing the information back from the top root node down to the intermediate nodes. In the top-down phase, the state of a child of an AND-node can be updated as:

$$t_{n_i} = \arg \max_{t_{n_1},\cdots,t_{n_l}} S_{n_{AND}}(s(n_1, t_{n_1}), \cdots, s(n_l, t_{n_1}), t_{n_{AND}}) \quad (3)$$

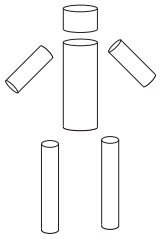


Figure 1. The configuration of the human parts.

Part	Joints
Head	head, left shoulder, right shoulder
Torso	left hip, right hip, spine, left shoulder, right
Left leg	left hip, left knee, left ankle, left foot
Right leg	right hip, right knee, right ankle, right foot
Left arm	left shoulder, left elbow, left wrist, left hand
Right arm	right shoulder, right elbow, right wrist, right hand

Table 1. The relationship between joints and parts.

and the state of a child of an OR-node can be updated as

$$t_{n_i} = \arg\max_{t_{n_i}} S_{n_{OR}}(s(n_i, t_{n_i}), t_{n_{OR}}) \tag{4}$$

In this way, we can compute the optimal state for the nodes in the AND-OR graph from the root node to the leave nodes by iteratively applying Eq. (3) and Eq. (4), and generate a parse graph.

3. The Human Part Configuration

In this paper, the human is represented as a set of human parts. Each human part consists of multiple human joints. We will describe the human part configuration in detail in this section.

The 3D joint positions are employed to characterize the pose of the human body. For a human subject, 20 joint positions are tracked by the skeleton tracker provided by Kinect. Each joint i has a 3-dimensional location coordinates vector $\boldsymbol{p}_j(t) = (x_j(t), y_j(t), z_j(t))$, a 3-dimensional motion vector $\boldsymbol{m}_j(t) = (\Delta x_j(t), \Delta y_j(t), \Delta z_j(t))$ as well as a visibility label $h_j(t)$ at a frame t. $h_j(t) = 1$ denotes that the j-th joint is visible in frame t and $h_j(t) = 0$ otherwise.

In this paper, we manually group the joints into the following six parts: head, torso, left arm, right arm, left leg and right leg. Each part is a cylinder that contains a set of tracked joints, illustrated in Fig. 1. The details of the part configuration are shown in Table 1.

In order to achieve invariance to absolute body position, the initial body orientation and the body size, we normalize the location and motion of the human joint. In the experiments we found the joints "head", "neck", "hip", "left shoulder", "right shoulder" to be the most stable. Thus, we use these joints to estimate an affine transformation matrix for each frame. This affine transformation matrix transforms the planes fitted by the 5 joints to the x-y plane and the joint "neck" to the origin. We can also obtain the azimuth rotation angle $\theta(t)$ from the affine transformation matrix for each frame by factorizing the affine transformation matrix.

4. Samples of the Discriminative Poses

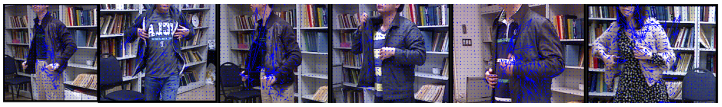
Some examples of the discriminative poses discovered with the proposed data mining algorithm is shown in Fig. 2. One subfigure in this figure corresponds to a discriminative pose for a specific action. We can see that one pose contains frames that are captured from different viewpoints and performed by different subjects. As a result, the frames for one pose can differ significantly in their appearance and motion. However, the proposed MST-AoG model can successfully represent the spatio-temporal patterns of one pose across different viewpoints.

(a) pick up with one hand

(b) pick up with two hands

(c) walk

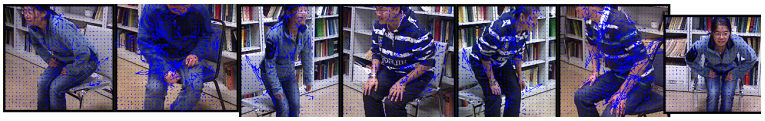
(d) trash



(e) donning

(f) doffing

(g) sit down



(h) stand up

(i) throw

(j) carry

Figure 2. Samples of the discovered discriminative poses. Each row shows a discriminative pose for one class, captured from different viewpoints. We can observe that one pose captured from different viewpoints have huge difference in the appearance and motion.

5. Representative and Discriminative Pose Mining Algorithm

The detail of the representative and discriminative pose mining algorithm is shown in Alg. 1. The definitions of the symbols in this algorithm can be found in the full paper.

Take a set of parts, and their candidate part items $\mathcal{T}_k$, the number of classes C, threshold T_{supp} and T_{disc}.
for *Class* $c = 1$ *to* C **do**
  Set P_c, the discriminative pose pool for class c to be empty : $P_c = \{\}$. Set $k = 1$.
  repeat
    1) Generate the candidates poses by augmenting the poses in the pool P_C with the part items of the k-th part $\mathcal{T}_k$.
    2) Add the candidate poses whose support is larger than T_{supp} to the pool P_c.
    3) Remove the poses in P_c that is non-maximal.
    4) $k = k + 1$
  until *no discriminative pose is added to* P_c;
  remove the poses whose discriminative scores are less than T_{disc} in the pool P_c.
end
return the discriminative poses pool for all the classes.

Algorithm 1: Discriminative Pose Mining