Title
Spatial-Temporal Hierarchical Model for Joint Learning and Inference of Human Action and Pose

Author
Nie, Xiaohan

Publication Date
2017

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Spatial-Temporal Hierarchical Model for Joint Learning and Inference of Human Action and Pose

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Xiaohan Nie

2017

ABSTRACT OF THE DISSERTATION

Spatial-Temporal Hierarchical Model for Joint Learning and Inference of Human Action and Pose

by

Xiaohan Nie

Doctor of Philosophy in Statistics

University of California, Los Angeles, 2017

Professor Song-Chun Zhu, Chair

In the community of computer vision, human pose estimation and human action recognition are two classic and also of particular important tasks. They always serve as basic preprocessing steps for other high-level tasks such as group activity analysis, visual search and human identification and they are also widely used as key components in many real applications such as intelligent surveillance system and human-computer interaction based system. The two tasks are closely related for understanding human motion, most methods, however, learn separate models and combine them sequentially.

In this dissertation, we build systems for pursuing a unified framework to integrate training and inference of human pose estimation and action recognition in a spatial-temporal And-Or Graph (ST-AOG) representation. Particularly, we study different ways to achieve this goal:

(1) A two-level And-Or Tree structure is utilized for representing action as animated pose template (APT). Each action is a sequence of moving pose templates with transition probabilities. Each Pose template consists of a shape template represented by an And-node capturing part appearance, and a motion template represented by an Or-node capturing part motions. The transitions between moving pose templates are governed in a Hidden Markov Model. The part locations, pose types and action labels are estimated together in inference.

(2) In order to tackle actions from unknown and unseen views we present a multi-view

spatial-temporal And-Or Graph (MST-AOG) for cross-view action recognition. As a compositional model, the MST-AOG compactly represents the hierarchical combinatorial structures of cross-view actions by explicitly modeling the geometry, appearance and motion variations. The model training takes advantage of the 3D human skeleton data obtained from Kinect cameras to avoid annotating video frames. The efficient inference enables action recognition from novel views. A new Multi-view Action3D dataset has been created and released.

(3) To further represent part, pose and action jointly and improve performance, we represent action at three scales by a ST-AOG model. Each action is decomposed into poses which are further divided into mid-level spatial-temporal parts (ST-parts) and then parts. The hierarchical model structure captures the geometric and appearance variations of pose at each frame. The lateral connections between ST-parts at adjacent frames capture the action-specific motions. The model parameters at three scales are learned discriminatively and dynamic programming is utilized for efficient inference. The experiments demonstrate the large benefit of joint modeling of the two tasks.

(4) The last but not the least, we study a novel framework for full-body 3D human pose estimation which is a essential task for human attention recognition, robot-based human action prediction and interaction. We build a two-level hierarchy of Long Short-Term Memory (LSTM) network with tree-structure to predict the depth on 2D human joints and then reconstruct the 3D pose. Our two-level model utilizes two cues for depth prediction: 1) the global features from 2D skeleton. 2) the local features from image patches of body parts.

The dissertation of Xiaohan Nie is approved.

Hongjing Lu

Luminita Vese

Ying Nian Wu

Song-Chun Zhu, Committee Chair

University of California, Los Angeles

2017

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Song-Chun Zhu, for guiding me to the area of computer vision and giving me the opportunity to conduct research in VCLA of UCLA. His high-level vision, passion and deep insight are always inspiring me during my Ph.D study. I'm very grateful for his helpful and painstaking guidance over the past few years. The most valuable lesson I learned from him is the spirit of never stoping to get better not only in academia but also in common life. I also thank other committee members: Hongjing Lu, Luminita A. Vese and Ying Nian Wu.

Many gratitude also goes to my excellent group members at VCLA lab of UCLA. I appreciate so much for their friendship, collaboration and support that helped me overcome setbacks and frustrations during my graduate study. Particularly I'd like to thank Benjamin Yao, Caiming Xiong and Seyoung Park for their wonderful collaborations with me. Many thanks also go to Tianfu Wu, Yibiao Zhao, Brandon Rothrock, Ping Wei, Bo Li, Joey Yu, Yang Lu, Meng Meng, Tianmin Shu, Wenguan Wang, Siyuan Qi, Yixin Zhu, Tian Han, Yang Liu , Hang Qi, Dan Xie and Yuanlu Xu.

Finally and most importantly, I cannot thank my family enough for their help during these years. None of this would have been possible without their countless love, support and patience.

VITA

2012–2017 Ph.D. Candidate (Statistics), UCLA, Los Angeles, CA

2009–2012 M.S. (Computer Science), Beijing Institute of Technology, Beijing, China

2005–2009 B.S. (Computer Science), Zhengzhou University, Zhengzhou, China

PUBLICATIONS

[1] Bruce Xiaohan Nie, Ping Wei, Song-Chun Zhu, *"Monocular 3D Human Pose Estimation by Predicting Depth on Joints"* Under review for ICCV, 2017

[2] Seyoung Park, Bruce Xiaohan Nie (equal contribution), Song-Chun Zhu, *"Attributed Grammar for Joint Estimation of Human Attributes, Part and Pose"* Under review for PAMI, 2017

[3] Bruce Xiaohan Nie, Caiming Xiong, Song-Chun Zhu, *"Joint Action Recognition and Pose Estimation From Video"*, CVPR, 2015

[4] Jiang Wang, Bruce Xiaohan Nie, Yin Xia, Ying Wu, Song-Chun Zhu, *"Cross-view Action Modeling, Learning and Recognition"*, CVPR, 2014

[5] Benjamin Yao, Bruce Xiaohan Nie, Zichen Liu, Song-Chun Zhu, *"Animated Pose Templates for Modelling and Detecting Human Actions"*, PAMI, 2013

CHAPTER 1

Introduction

1.1 Motivation

In the research area of computer vision, human pose estimation and action recognition are two very classic and of particular important tasks. As shown in Figure 1.1 the goal of pose estimation is to localize the human joints from images and action recognition is targeting at classifying human actions from images or videos. The two tasks always serve as the pre-processing steps for other high-level tasks such as group activity analysis, human attention recognition and human tracking. They are also key components for many real applications in industry such as intelligent surveillance system, human-computer interaction based system, content-based image and video retrieval. The researchers in computer vision have paid a lot of attention to these two tasks over the past few years because of the easy acquisition of big data and rapid development of computer hardware. The robust and high-performance algorithms for the two tasks are urgently needed not only in research area but also in industry.

Despite their different goals they are highly related. However, most previous works train models for the two tasks separately and combine them in a sequential order: using pose estimation as an input for action recognition or estimating poses for each action respectively. We believe that it is desirable to study them in a unified framework due to the following reasons:

(1) Many human actions are defined on human postures like running, walking, jumping, bending, pointing and so on. For those actions, the human poses naturally provide enough information for inferring action labels which means the actions can be recognized easily solely based on human poses. There are some actions defined on the interaction between

people and objects such as typing keyboard, reading book, throwing ball and so on. Those actions can be recognized by human poses combined with contextual objects. A robust pose estimation method will help action recognition a lot undoubtedly.

(2) Human actions also provide strong priors on the placement and movement of human poses, especially in video based systems. The action label from video provides strong constraints on the geometric relations of human parts in each frame and also defines the movement of poses among several frames. In real applications, the number of interesting actions is limited which enables us to discover the action-specific motion information to benefit pose estimation.

(3) Many previous methods take the pose estimation as a preprocess step for action recognition. The main drawback of such methods is that the performance of action recognition highly relies on the output of pose estimation. However, the most discriminative parts such as arms, hands, legs and feet are often miss-detected in pose estimation due to the large pose variation and complex background, thereby the subsequent action recognition is deteriorated. The action-specific large motion of those parts are critical cues for the part localization. There are also some methods of action recognition bypass body poses and only use coarse/mid-level features which are not explainable and hard to be analyzed.

Taking the above reasons into consideration, a unified framework is needed to represent human pose and action explicitly and joint the learning and inference of pose estimation and action recognition. In this dissertation, we propose a Spatial-Temporal And-Or Graph model to integrate the two tasks so they can benefit each other during training and testing. Specifically we studied several ways to achieve this goal:

1. We represent action as animated pose template (APT) by utilizing the two-level And-Or Tree structure. Each action is composed by a sequence of moving pose templates (MPT) with transition probabilities. Each MPT consists of a shape template represented by an And-node capturing appearance and a motion template represented by an Or-node capturing motions. The transitions between MPT are controlled by a Hidden Markov Model. The part locations, pose types and action labels are inferred together.

2. A multi-view Spatial-Temporal And-Or Graph model is presented for cross-view action recognition. In the MST-AOG, the geometry, appearance and motion variations are explicitly modeled. Each action is composed of 3D poses each of which can be projected into arbitrary views. In training, we take advantages of the 3D human skeleton data obtained from Kinect cameras to avoid cumbersome annotation of human poses. The efficient inference enables action recognition from novel viewpoints. We also collect a new Multi-view action3D dataset and release it to public.

3. We represent action at three scales by a Spatial-Temporal And-Or Graph model: coarse level capturing overall motion and appearance of the video, middle level capturing the action-specific motions among the spatial-temporal parts (ST-parts), fine level representing appearance of small parts at highest resolution. To further improve the performance, we also integrate deep learned features into this hierarchical structure.

The last but not the least, we propose a novel framework based on deep networks for full-body 3D human pose estimation. Specifically, a two-level LSTM network is utilized to predict the depth on human joints, thus the 3D pose is reconstructed from predicted depth values. Our deep network has tree-structure defined on the skeleton kinematic relation and can help broadcast the information between different joints in a top-down manner.

1.2 Overview of the Dissertation

Our research in this dissertation aims to provide a framework to unify the two tasks which are naturally coupled and should be studied together: human pose estimation and human action recognition. The proposed unified framework should enable the joint training and inference of the two tasks so they can complement each other. Towards this goal, we study several ways by using ST-AOG models.

Firstly, in chapter 2, we characterize each action as just an evolution of different poses and we represent animated pose template (APT) using a two-level And-Or tree structure for detecting short-term, long-term and contextual actions in videos. Each APT is a sequence of motion pose templates (MPTs) with transitions between the neighbors. Each MPT consists

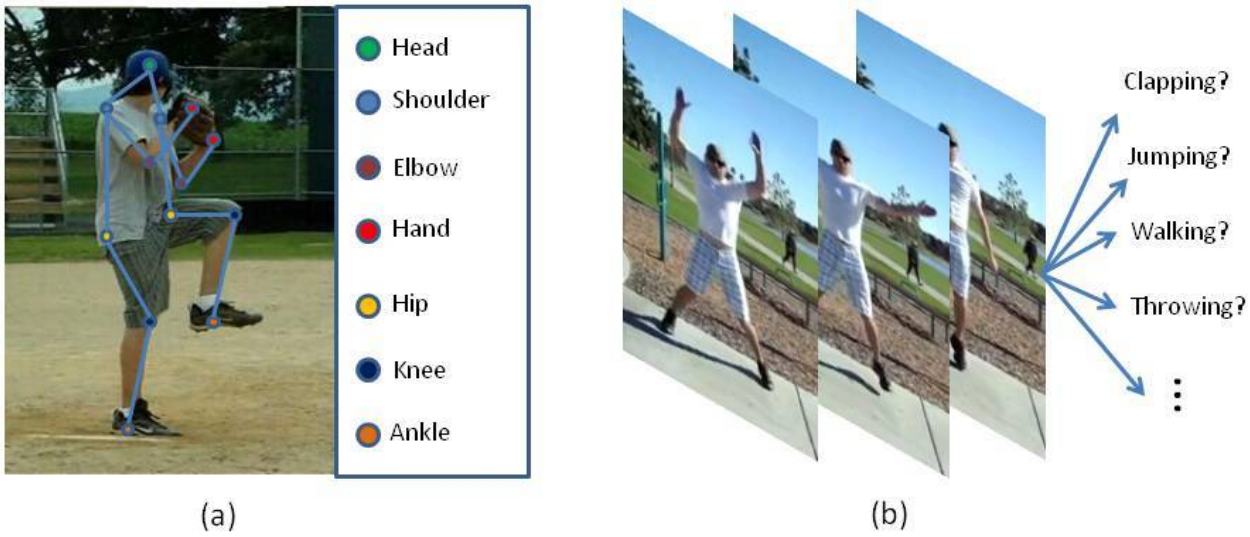

Figure 1.1: The goal of (a) pose estimation and (b) action recognition. Circles with different colors represent different human joints. We draw edges connecting joints only for better visual effect.

of two components: i) a motion template specifying the motion of the parts by the Histogram of Optical-Flows (HOF) features. ii) a shape template with deformable parts represented in an And-node whose appearance are represented by the Histogram of Oriented Gradient (HOG) features. Each Shape template may have several motion templates represented by an Or-node capturing different motion directions of the same part. The MPT is suitable for detection short-term action snippets in 2-5 frames. We extend MPTs to APT by adding the temporal constraints in a Hidden Markov Model (HMM). In order to detection action with contextual objects, we treat objects as additional parts of the MPT and also add spatial constraints between parts. To train the model, we manually annotate part locations on several key frames of each video and cluster them into pose templates using EM. We learn model parameters by applying a Semi-Supervised Structural SVM algorithm (SS-SVM) that iterates between two steps: i) updating model parameter using labeled data. ii) imputing the unlabeled data with parameters learned from the previous step. In inference, we use dynamic programming to efficiently compute the best sequence of pose templates for each action. We evaluate our method on several public dataset and the results demonstrate that our method achieves comparable or better performance on both action recognition and pos

estimation compared to state-of-the-art methods.

Secondly, in chapter 3, we propose a multi-view spatial-temporal And-Or graph (MST-AOG) representation for cross-view action recognition. One big limitation of the proposed APT and most current methods is the unpredictable performance in the situation that the testing actions are from viewpoints which are totally different from the training data because the visual features are very different from different views. As a compositional model, MST-AOG compactly represents the hierarchical combinatorial structures of cross-view actions by explicitly modeling the geometry, appearance and motion variations. In MST-AOG, each action is composed of a sequence of 3D key poses each of which can be projected onto 2D image under a certain viewpoint. Another challenge of modeling action by poses is the cumbersome annotations of poses. In the training, we take the advantage of the 3D human skeleton produced by Kinect sensors for avoiding the error-prone and time-consuming human annotations. This 3D information is only available in training and not be used in testing. Our method uses a set of discrete views in training to interpolate arbitrary novel views in testing. The extensive experiments demonstrate that our new action representation significantly improves the accuracy and robustness for cross-view action recognition on 2D videos. We also crate and release a new Multi-view 3D action dataset.

In our proposed APT and MST-AOG model, the specific motion information of different actions are characterized by the transitions of key poses which is too coarse to prevent miss-detecting of parts with small resolutions such as arms, legs and ankles. To further represent the relations of actions, pose and parts compactly and jointly, we represent actions at three scales by a Spatial-Temporal And-Or Graph (ST-AOG) model in chapter 4. Each action is decomposed into poses which are further divided into mid-level ST-parts and then fine-level parts. The ST-parts is a combination of three parts at highest resolution and discretized into several components by clustering. The ST-parts within the same component have small variation of appearance and deformation, thus they are much easier to be detected than parts at fine-level. The geometrical deformations between different ST-parts are treated independently in single frame. To capture the specific action motion we add lateral connections between ST-parts at adjacent frames in the ST-AOG. The lateral edges represent both

deformation and transition of ST-parts. The model parameters for three scales are learned discriminatively by S-SVM. In inference we use dynamic programming to infer the best ST-part sequence of each action because of the independency of ST-parts. To further improve the performance we ground our model on deep learned features. The ST-parts candidates are then extracted from the deep convolutional network. The experiment results show that our approach achieves state-of-art accuracy in action recognition while also improving pose estimation, thus demonstrate the huge benefit of joint modeling of the two tasks.

In chapter 5, we propose a novel framework focusing on estimation of full-body human 3D pose. A two-level hierarchy of Long Short-Term memory (LSTM) network is proposed to predict the depth on human joints and then reconstruct the 3D human pose. The first level of our model contains two key components which captures different information from two data sources: 1) the skeleton-LSTM network which takes the predicted 2D joint locations to predict joint depth. The global skeleton feature can help to remove the physically implausible 3D joint configurations. 2) the patch-LSTM network which uses the local image patches of body parts to predict depth. Considering using the kinematic relation of human skeleton, we have manually defined tree-structure in our deep network so the information at different joints are broadcasted in top-down manner during training. Two LSTM networks at the first level are aggregated in the second level for final depth prediction. The extensive experiments on two public 3D datasets demonstrate our better qualitative and quantitative performance compared to state of the art methods.

1.3 Literature Review

Since our goal is the joint modeling of pose estimation and action recognition, our work is related to these two streams of research in literatures and we will review them respectively. Researchers have made a lot of progress in the two areas during the past few years and we select some classic and popular articles to review. We begin by reviewing literatures of pose estimation.

1.3.1 Pose Estimation

The pose estimation methods can be categorized by two different types of input data: single RGB image and video. Most video pose estimation methods are based on single image pose estimation with additional temporal constraints added among poses at different frames, so we focus on reviewing single RGB image based pose estimation. The methods of pose estimation on single image are reviewed with two aspects: pose estimation with pictorial structure, pose estimation by deep convolutional neural networks (ConvNets).

The pictorial structure has achieved notable success in the area of object detection [FMR08, FGM10b] and is applied successfully to many other applications [YR12, LHW13, YWZ14]. The main idea of pictorial structure is to model the deformable parts of the object. Each part is always parameterised by its position, orientation and scale. The likelihood function of a configuration of parts consists of unary potentials measuring the local image evidence given the position of parts and binary potentials evaluating the relative deformation of two parts. The relative deformation of parts are usually modeled as a tree structure which enable the efficient inference by dynamic programming. As shown in Figure 1.2 the pictorial structure can be naturally applied to human pose estimation by representing human pose as deformable human parts. Yang and Ramanan [YR12] build a tree-structure spring model to capture both spatial and co-occurrence relations between parts. Each part has several components which represents different orientations. The deformation of a part is only conditioned on its parent. The model is trained in max-margin framework and dynamic programming is used for inference. Brandon et al. [RPZ13] uses a compositional And-Or Graph (AOG) model to represent the large appearance and geometry variation of pose. The large appearance variation among people is handled by substituting parts with their variants. Each variant has its own deformation and context-sensitive compatibility with neighboring part variants. They incorporate the background region segmentation to better estimate the contrast of a part region from its cluttered surroundings. The Structure-SVM (SSVM) is applied for training model parameters. Pinshchulin et al. [PAG13] extends the basic pictorial model to a more flexible structure with stronger local appearance representations including

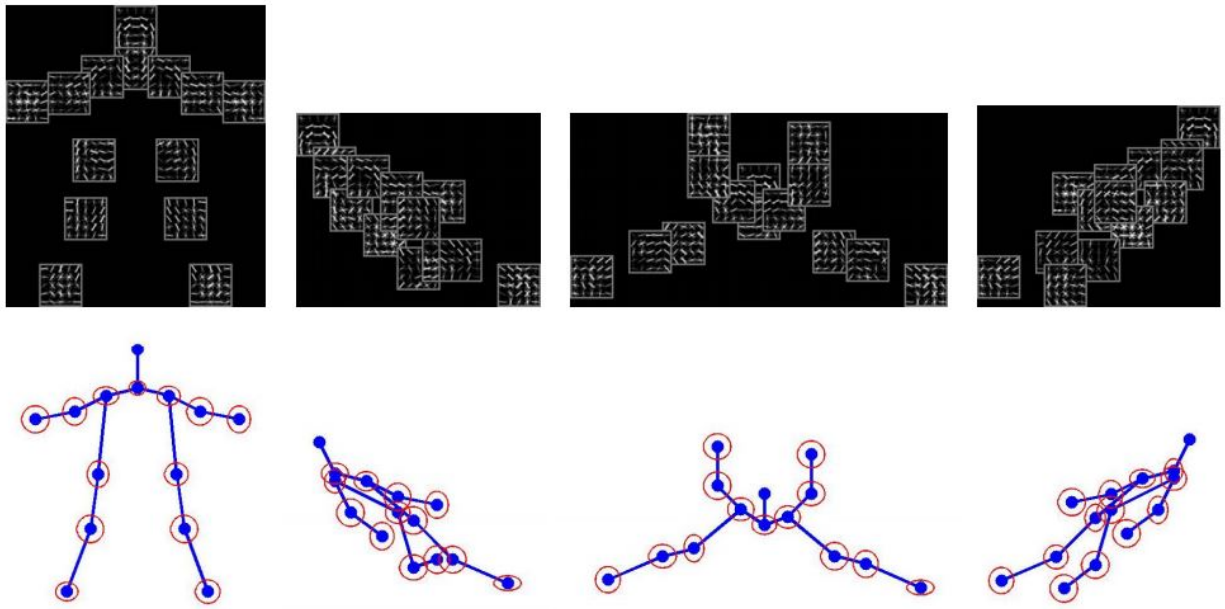

Figure 1.2: A visualization of four different mixtures of human pose from [YR12]. The appearance of each part is represented by Histogram of Gradients (HOG). The edges connecting parts represents the location deformation and part type compatibility. The tree structure enables efficient inference by dynamic programming.

rough part detectors and mixtures of several specific part detectors. They combine the mid-level representation based on semi-global poselets [BM09] with local appearance templates. The poselets can capture the appearance of different spatial configuration of parts.

Over past few years the powerfulness of deep ConvNets have been demonstrated in many computer vision tasks such as object classification [KSH12, SZ14b] and detection [GDD14, Gir15, RHG15], video analysis [SZ14a, WQT15], image to text [KL15, MXY15] and so on. The networks used in those works are consists of many linear and non-linear layers so they are much deeper than conventional methods which are shallow and use handcrafted features. Recently many works have explored deep ConvNets for human pose estimation. Toshev and Szegedy [TS14a] formulate the pose estimation as a regression problem based on the cascade of Deep CNN networks. The initial joint locations are predicted by a DCNN and then a few subsequent DCNNs are applied to refine the predictions gradually. The input of the refining DCNN at each stage is the sub image cropped based on the predictions from last stage.

Xianjie et al. [CY14] combines the pictorial graphical model with DCNN. Both the unary and binary potential in their model are learned with a DCNN. The pairwise relationship between parts are predicted from the local image measurement. Tompson et al. [TJL14] combines a ConvNet part detector with a part-based spatial model into a unified learning framework. The multi-resolution feature representation with overlapping receptive fields is utilized in their ConvNet architecture. The spatial model is applied to approximate MRF loopy belief propagation. The two models are trained jointly by back-propagation. Newell et al. [NYD16] propose a novel convolutional network architecture to directly predict the probability maps of parts end to end. As shown in Figure 1.3 the network has repeated bottom-up and top-down structures with intermediate supervision. The features are processed across different scales and consolidated to best capture the various spatial relationships of parts. Wei et al. [WRK16] implicitly model long range dependencies between parts by designing a multi-stage architecture based on several convolutional networks. The network at each stage directly operates on the confidence maps computed from previous stages and produces refined estimates for the next stage. Cao et al. [CSW17] represent a framework with two ConvNets to do multi-person pose estimation. One ConvNet is applied for generating candidates for each part and another ConvNet aims to estimate the limb orientation densely at each position on the human skeleton. The likelihood of connecting two parts to form a limb is the summation of likelihoods of points on this limb. The inference of multiple poses is formulated as a set of bipartite matching problems over a tree skeleton.

We also review two works of pose estimation from videos. Cherian et al. [CMA14] extends the pictorial model with temporal edges between parts at adjacent frames. The geometric, appearance and motion compatibility between parts are captured by different cues from local image features and optical flows between two frames. The approximate inference is performed on the highly loopy graphical model. Instead of using a graphical model Shen et al. [SYM14] formulates the problem as matching dense trajectories from 2D video with the projection of 3D trajectories from a 3D human motion library. The 3D trajectories are projected onto 2D space under different viewpoints during inference.

With the success of deep networks on a wide range of computer vision tasks and especially

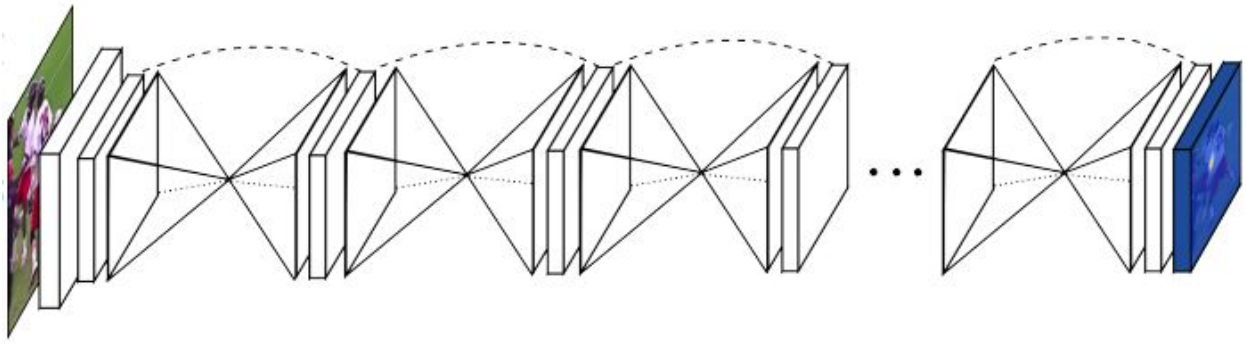

Figure 1.3: A visualization of deep network for end-to-end pose estimation from [NYD16]. The multiple stacked hourglass modules with repeated bottom-up top-down layers allow for prediction from features at multiple scales.

2D human pose estimation, the 3D pose estimation from monocular image using deep networks [LC14, LZC15, RS16, YIK16, ZZL16] have received lots of attentions recently. Some approaches [LC14, LZC15] directly predict the 3D pose from images so their training and testing are restricted to the 3D MoCap data in a constrained environment. Li et al. [LC14] applies a deep network to regress 3D pose and detect 2D body parts simultaneously. In this method there is no explicit constraint to guarantee that the predicted 3D pose can be projected to the detected 2D part locations. Li et al. [LZC15] learn the common embedding space for both image and 3D pose using a deep network. Some methods [YIK16, ZZL16] use two different data sources for training 2D pose estimator and 3D pose predictor. The benefit is that their 2D pose estimators can be trained from another data source instead of the 3D Mocap data which is captured in a constrained environment. Zhou et al. [ZZL16] predict 3D poses from a video sequence by using temporal information. The 3D pose estimation is conducted via an EM type algorithm over the entire sequence and the 2D joint uncertainties are marginalized out during inference. Yasin et al. [YIK16] propose a dual-source approach to combine 2D pose estimation with 3D pose retrieval. The first data source only contains images with 2D pose annotations for training 2D pose estimator and the second source consists of 3D MoCap data for 3D pose retrieval. Another work worth mention is [KG15] which use regression forest to infer the depth information and estimate 3D joint location probabilities from image patches. The independent joint probabilities are used with the pictorial

structure model to infer the full skeleton.

1.3.2 Action Recognition

Here we divide the methods of action recognition into three categories: coarse/mid-level feature based methods, pose feature based methods and end-to-end action recognition by deep learning.

Coarse/mid-level feature based action recognition. Aaron et al. [BD01] proposes a whole template representation of human movement and formate the action recognition as a matching problem. The video is encoded by two images: one is the motion energy image which captures whether the pixel has motion and another is the motion history image which captures how long the motion of the pixel has. Lena et al. [GBS05] represents video as a whole spatial-temporal shape generated by the moving foreground. The local and global features are extracted from the spatial-temporal shape for matching actions in a database. Another popular stream of framework for action recognition is to build classifiers on bag-of-words (BoW) representation on spatial-temporal interest points such as cubiods [DRC05] and 3D Harris corner [LC05]. This category of method can be thought as the direct extension of object detection using 2D spatial interest points. With the interest points detected, the appearance and motion features like Histogram of Gradients (HOG) [DT05] and Histogram of Optical Flows (HOF) [LMS08] are extracted and used for clustering to get bag-of-words representation. Finally a discriminative classifier is trained on BoW. Instead of using features around interest points, Wang et al. [WKS13] extracts dense trajectories by optical flow and builds a bag-of-words representation on trajectory aligned features. Although the coarse/mid-level feature based methods have achieved quite high performance on some datasets, they offer no intuition about the relations between pose and action and cannot be analyzed easily.

Pose feature based action recognition. Due to the great progress made in pose estimation on single image, many action recognition approaches apply the pose estimation as the preprocessing step and extract pose features for the subsequent action recognition.

Yao et al. [YL12] first estimates the 2D poses from single images and then matches the estimated poses with a set of representative poses from actions in the training data. Desai et al. [DR12] proposes phraselet to representation actions. Each phraselet is a combination of human pose and objects. The DPM [FMR08] framework is applied for detecting actions, poses and objects together. Maji et al. [MBM11] uses distribution of activations from poselet [BM09] detectors to represent actions. Yao et al. [YGG12] tries to couple action recognition and pose estimation. The pose estimation is formulated as an optimization over a set of action specific manifold. The two tasks are inferred iteratively. However, they require that each video is captured from multiple views simultaneously which limits the applicability of this method in common environment.

End to end action recognition by deep learning. With the successfulness of deep learning on many computer vision tasks, the trend of designing end to end action recognition method becomes very popular. Karpathy et al. [KTS14] provides extensive empirical evaluation of deep CNNs with different structures for large-scale video classification on a new dataset of 1 million videos collected from YouTube. They study different approaches for extending the connectivity of CNN in time domain to capture the local spatial-temporal information. Simonyan et al. [SZ14a] propose a two-stream deep CNN architecture which incorporates spatial and temporal networks. The inputs for spatial network are still frames and for temporal network are optical flows. They also demonstrate that the multi-task learning on two different action datasets can largely increase the amount of training data and improve the performance of both. Donahue et al. [DHG15] proposes a novel recurrent convolutional architecture for large-scale action recognition. The recurrent structure consists of a ConvNet processing single frame and a Long-Short Term Memory (LSTM) network processing a sequence of frames. Each single frame is first encoded into a deep feature by the ConvNet and the LSTM aims to deal with the long-term dependencies in actions. This framework can also predict the action label for each frame instead of the whole video so it can be applied for action parsing.

CHAPTER 2

Animated Pose Templates for Modeling and Detecting Human Actions

2.1 Introduction

2.1.1 Backgrounds and Motivations

In recent years, human action recognition has attracted a lot of research attentions from computer vision because that it is needed in a wide range of applications from intelligent video surveillance system, human computer interaction to content based video retrieval. There are challenges at multiple levels for building a robust system for real-world human action understanding: 1) classifying the action category; 2) localizing the area of interest; 3) explaining the interactions between contextual objects and action agents. Recent works has made huge progress on the first challenge in the datasets which only contain one person doing action with static or uncluttered background, however, the action understanding from in-the-wild videos is still a hard problems because of the cluttered background and background motion. In general, actions have diverse complexities in space and time: 1) In space, actions can be defined by the movement of body parts such as clapping and waving, or by the human-object interactions such as making coffee and washing dishes. In the later case, the contextual information from objects are critical for classifying the action; 2) In time, actions can be defined on a single frame such as sitting and standing or 2-5 frames such as waving hand and pushing bottom, or a few seconds such as making coffee and mopping floor.

We review the related literatures in four categories in the following.

i) **Action recognition by template matching.** The idea of template matching is ex-

ploited in the early stage of action recognition. There approaches are intuitive and attempt to represent the motion by 2D templates or 3D volumes from video sequences. For example, Bobick et al. [BD01] use motion history images to capture both motion and shape information to represent actions, The global descriptors motion energy image and motion history image are introduced and used as templates which can be matched to stored templates of known actions. Gorelick et al. [GBS07] extend the concept of template matching from 2D motion templates to 3D space-time volumes. The space-time features such as local space-time saliency, action dynamics, shape structures and orientations are extracted for template matching. The common disadvantage of those template matching methods is that they rely on the static and simple background which allows for segmenting out the foreground and they are sensitive to appearance and view-point variations. In our method, we don't need the foreground segmentation.

ii) **Action recognition by spatial-temporal interest points.** In order to capture the large appearance and geometric variations from action videos, researchers extracted spatial-temporal interested points (STIP) and HoG and HOF features around them, and train classifier in max-margin framework [LMS08]. They either pooled the features in a bag-of-word (BoW) representation [DRC05, SLC04] or pyramid structure [KG10]. There are some shortcomings for these methods: 1) BoW cannot represent human body parts explicitly and the K-mean clustering for quantization rely on low-level features and is often unreliable. 2) the spatial relations between body parts are missing. 3) The motion information for each part and pose is not modeled.

iii) **Action recognition by pose estimation.** Due to the great progress made by human detection and pose estimation, recently people attempt to use pose estimation for action recognition. Mori et al [YYM10] use HoG features and SVM classifiers for action recognition and they show that it is beneficial to treat poses as latent variables in the SVM training because the same type of action may contain multiple poses. Yang and Ramanan [YR12] propose a mixture-of-parts model for pose estimation. In their work, the strong supervision of body parts is reported to be better than laten part models.

iv) **Action recognition by scene context.** Some actions are defined by human-object

Figure 2.1: action snippets contain rich appearance, geometry and motion information in 3 frames. 1) the static pose; 2) the short-term motion velocities (blue arrows)

interactions instead of poses. Lan et al. [LWY10] utilizes contextual information between single person and a group for better action and group activity recognition. Most recent works [PJZ11, YF10] try to model the body body configuration and object location jointly. Most of these works detect objects and poses in static images and the motion patterns are not considered.

2.1.2 Method Overview

In order to overcome the shortcomings of current methods and represent actions of different space-time complexity, we propose the Animated Pose Templates (APT) to classify the short-term, long-term and contextual actions from videos. We overview our model and method in the following.

i) **Short-term actions as moving pose templates.** We propose moving pose templates (MPT) to represent the short-term actions or the so-called action snippets which refer to actions with length 3-5 frames. Two examples for clapping and drinking are shown in Figure []. The long term action is always composed of several short-term actions. We learn the dictionary of MPT by clustering the action snippts. Each MPT consists of 1) a shape template (ST) which captures the body part appearance by HOG features and 2) a motion template (MT) specifying the motion information of part by the HOF features. Note that the different action snippet may share the shape template and differ in the motion template.

ii) **Long-term actions as animated pose templates.** We propose animated pose template (APT) to represent the long-term action which is composed of a sequence of moving

pose templates. The APT is a generative model based on the MPTs. We track the bounding boxes for the root node and parts over time by HMM model which captures the spatial constraints of parts and also the transition between different type of MPT.

iii) **Animated pose template with contextual objects.** In order to recognize the actions which are defined on the contextual objects. We annotate the objects with bounding boxes and treat them as additional parts in moving pose templates.

iv) **Inference by dynamic programming.** Since our MPT has an And-Or tree structure and APT is modeled by HMM in temporal transition, the dynamic programming can be naturally adopted for effective inference in two steps: 1) Detecting the human pose candidates for the MPT by dynamic programming. 2) Computing the APT with contextual objects in time by beam search using the pose candidates from the first step.

v) **Learning by semi-supervised structural SVM (SS-SVM).** We annotate body parts and contextual objects as several key-frames for each video clip of each action category. We have two sets of unknown parameters: 1) the latent variables for the unannotated frames including pose labels and part locations. 2) model parameters such as the weights for HOG and HOF features, coefficients for pose transitions and part deformations. We use the SS-SVM [TJH05] for training which iterates between two steps: 1) updating model parameters by solving the structural SVM optimization; 2) imputing missing variables with parameters learned from the previous step. This algorithm belongs to a family of optimization methods known as the concave-convex procedure (CCCP) which converge to a local optimal.

We evaluate our method on several public action datasets and another challenging our-door action dataset collected by ourselves. The results demonstrate that our method can discover the key poses of actions as well as the contextual objects, and achieve comparable or better performance relative to state-of-the-art methods.

Our main contribution is a comprehensive model - animated pose templates (APT) to represent actions with diverse space-time complexities. The And-Or tree structure in space and the HMM structure in time enables fast inference by dynamic programming in time and space. The results show that we achieve comparable or better performance than the

state-of-the-arts in several challenging datasets.

The remainder of the chapter is organized as follows: In Section 2.2, we introduce the formulations of animated pose templates. In Section 2.3, several inference strategies are introduced for detecting short-term action snippets and long-term actions. Section 2.4 presents the learning algorithm. In Section 2.5, we present the experimental results and comparison with other state-of-the-art methods on 4 public datasets and a contextual action dataset we collected.

2.2 Representation

In this section, we present formulates for moving pose templates (MPT), animated pose templates (APT) and APT augmented with contextual objects.

2.2.1 Moving Pose Templates (MPT)

Each action consists of a sequence of key poses or action snippets each of which is represented by a moving pose template. Figure 2.2 illustrates 3 poses for one example of hand-clapping action from the MSR dataset. Each key pose is composed of a shape template and two motion template depending on the motion direction of the arms.

The moving pose templates are denoted by

$$\Omega_{mpt} = MPT_i = (ST_i, MT_i) : i = 1, ..., n \tag{2.1}$$

where each shape template ST_i contains a root templte ST_{i0} for the whole person and m templates ST_{ij} for body parts: $ST_i = (T_{i0}, T_{i1}, ..., T_{im})$. This is similar to DPM model for human detection [FGM10b]. Each template T_{ij} has the following components to describe the geometry and appearance:

i) the body label a_{ij} indicates the body part category. The parts are different in different actions and $a_{ij} \in \Omega_{part}, \Omega_{part} = {}'figure', 'head', 'torso', ...$ In training videos, the bounding boxes for the root and parts are annotated in key frames.

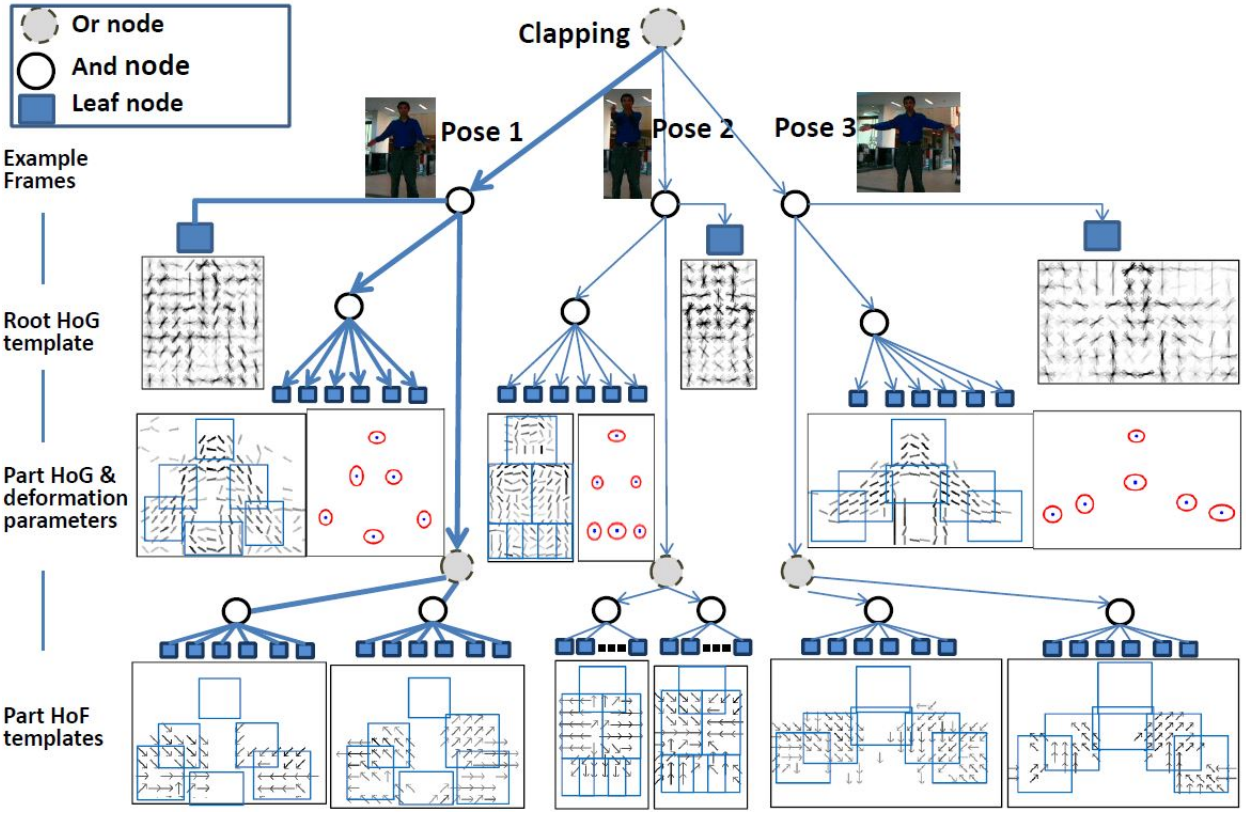

Figure 2.2: Each moving pose template is composed of a shape template of HOG feature and a motion template of HOF feature. The shape template has a root template for entire bounding box and several part templates. The deformation of part template relative to the root template is governed by 2D gaussian function whose mean and variance are visualized by ellipses. Each shape template is associated with 2 motion templates represent the different movement of body parts. The small arrows represent the dominant flow directions. The 3 shape templates multiplied by 2 motion templates represent total 6 action snippets in the And-Or tree where and-node represents composition and or-nodes represents selections between alternative choices.

ii) Z_{ij} represents the image domain for the root or part templates. It includes the upper-left corner, window width and height. We allow each part to rotate in $-20^o, 0, +20^0$.

iii) X_{ij} represents the HOG feature vector extracted from Z_{ij}.

iv) h_{ij} represents vector of latent variables of the template. It includes the displacement $d_{ij} = (dx, dx^2, dy, dy^2)$ relative to the anchor point. Similar to DPM, we use a 2D quadratic

function to penalize the d_{ij}, then the deformation is governed by a 2D gaussian function which is illustrated as ellipses in Figure 2.2. The rotation is not penalized.

The motion template MT_i is a vector for m parts: $MT_i = (V_{i1}, V_{i2}, ..., V_{im})$ where V_{ij} represents motion for each part. We use the variation of the HOF features [LMS08]. Instead of projecting velocities of each each pixel onto 4 bins in [LMS08], we use 8 bins and find empirically that the finer orientations and continuous scale works better. For certain parts which do not have motions, the motion features will be all zeros.

We use l_i to denote the label of MPT_i with different ST and MT. The model parameters for each MPT_i is a vector w_i with the same length as the feature vector for the appearance, deformation and motion features, $w_i = (w_i^A, w_i^D, w_i^M)$. The following score function is used for evaluating the mpt^t at frame t,

$$S(mpt^t) = \sum_{j=0}^{m} < \omega_{ij}^A, X_{ij}^t > + \sum_{j=1}^{m} < \omega_{ij}^D, d_{ij}^t > + \sum_{j=1}^{m} < \omega_{ij}^M, V_{ij}^t > \tag{2.2}$$

The score function can be interpreted as a log-posterior probability up to a constant. In inference, the scores for all possible mpt at each time frame are computed and only the top candidates kept.

2.2.2 Animated Pose Templates (APT)

The animated pose template (APT) is a sequence of moving pose templates with the transition probability p. Each possible APT in the time interval $[t^s, t^e]$ is written as a stochastic set,

$$\Omega_{apt} = apt[t^s, t^e] = (mpt^{(t^s)}, ..., mpt^{(t^e)}) \tag{2.3}$$

The markov chain probability controls the transition in the sequence,

$$p(mpt^{(t^s)}) \prod_{t=t^s}^{t^e} p(mpt^{(t+1)} | mpt^{(t)}) \tag{2.4}$$

where the initial probability $p(mpt^{(t^s)})$ is uniform over all MPTs. The trainsition probability $p(mpt^{(t+1)}|mpt^{(t)}$ includes two components:

i) The transition probability $p(l^{t+1}|l^t)$ between MPT label l^t and l^{t+1}. As Figure [] shows, we use two types of HMM model: 1) an ergodic HMM which allows flexible transition between any MPT label. 2) a left-right HMM which only allow the state to either transit to a certain state or stay in the current state. The probability of staying at current state indicates the speed of the action.

ii) The tracking probability $p(Z^{t+1}|Z^t)$ between parts at adjacent frames. We assume that the parts are conditionally independent given the root template position, therefore the probability can be factorized as

$$p(Z^{t+1}|Z^t) = \prod_{j=0}^{m} p(Z^{t+1}_{l^{(t+1)j}}|Z^t_{l^{(t)j}}) \tag{2.5}$$

where $p(Z^{t+1}_{l^{(t+1)j}}|Z^t_{l^{(t)j}})$ is governed by a gaussian distribution on the position (x,y) and scale s. We use a six-dimensional vector to express the quadratic function: $\phi^t_{ij} = (dx_{ij}, dx^2_{ij}, dy_{ij}, dy^2_{ij}, ds_{ij}, ds^2_{ij}$

We use a score function to represent the logarithm of the transition probability between two MPTs,

$$S(mpt^{t+1}|mpt^t) = log(p(l^{t+1}|l^t)) + \sum_{j=0}^{m} < \omega^T_{l^t j}, \phi^t_{l^t j} > \tag{2.6}$$

In summary, an animated pose template hypothesis $apt[t^s, t^e]$ is evaluated in the following score function,

$$S(apt[t^s, t^e]) = \sum_{t=t^s}^{t^e} S(mpt^t) + \sum_{t=t^s}^{t^e-1} S(mpt^{t+1}|mpt^t) \tag{2.7}$$

The inference algorithm searches for the APT which maximizes the above score function. This is equivalent to maximizing the posterior probability except that the model parameters are learned discriminatively.

2.2.3 Animated Pose Templates with contextual objects

In order to represent actions which are defined by contextual objects, we augment the MPT by adding contextual objects as additional human parts. We use $\Omega_{obj} = '\mathit{torch}', '\mathit{button}', '\mathit{cup}', ...$ to denote the set of possible objects in actions. For example, Figure 2.3 illustrates one instance of an action interacts with objects: the person walks on the 'ground' and then push 'button', and pick up the merchandise at the 'outlet'.

As illustrated in Figure 2.4, we treat objects in the same way as body parts except that they don't have motion features. We use $CO = (T_{im+1}, ..., T_{im+m'})$ to denote the m' objects. Each object template T_{ij} has the same variables as the body parts such as a_{ij} for object name, Z_{ij} for the object bounding box, X_{ij} for the HOG feature inside Z_{ij} and d_{ij} for the deformation relative to the body part. The bounding boxes and labels of objects are annotated in key frames of training data. We add the following term into the score function $S(mpt^t)$,

$$S(CO^t) = \sum_{j=m+1}^{m+m'} (< \omega_{ij}^A, X_{ij}^t > + < \omega_{ij}^D, d_{ij}^t >) \tag{2.8}$$

The positions and labels of objects are inferred together with body parts and action labels.

2.3 Inference

2.3.1 The Objectives of Inference

Our inference algorithm do the following three tasks in a single framework by dynamic programming in space and time.

i) Firstly we detect the action snippets at each frame using the learned moving pose templates. This includes localizing the body parts Z_t and classify the MPT label l^t. The contextual objects are also localized if the action has any object.

ii) Secondly, with detected candidates of MPTs, we recognize the animated pose templates

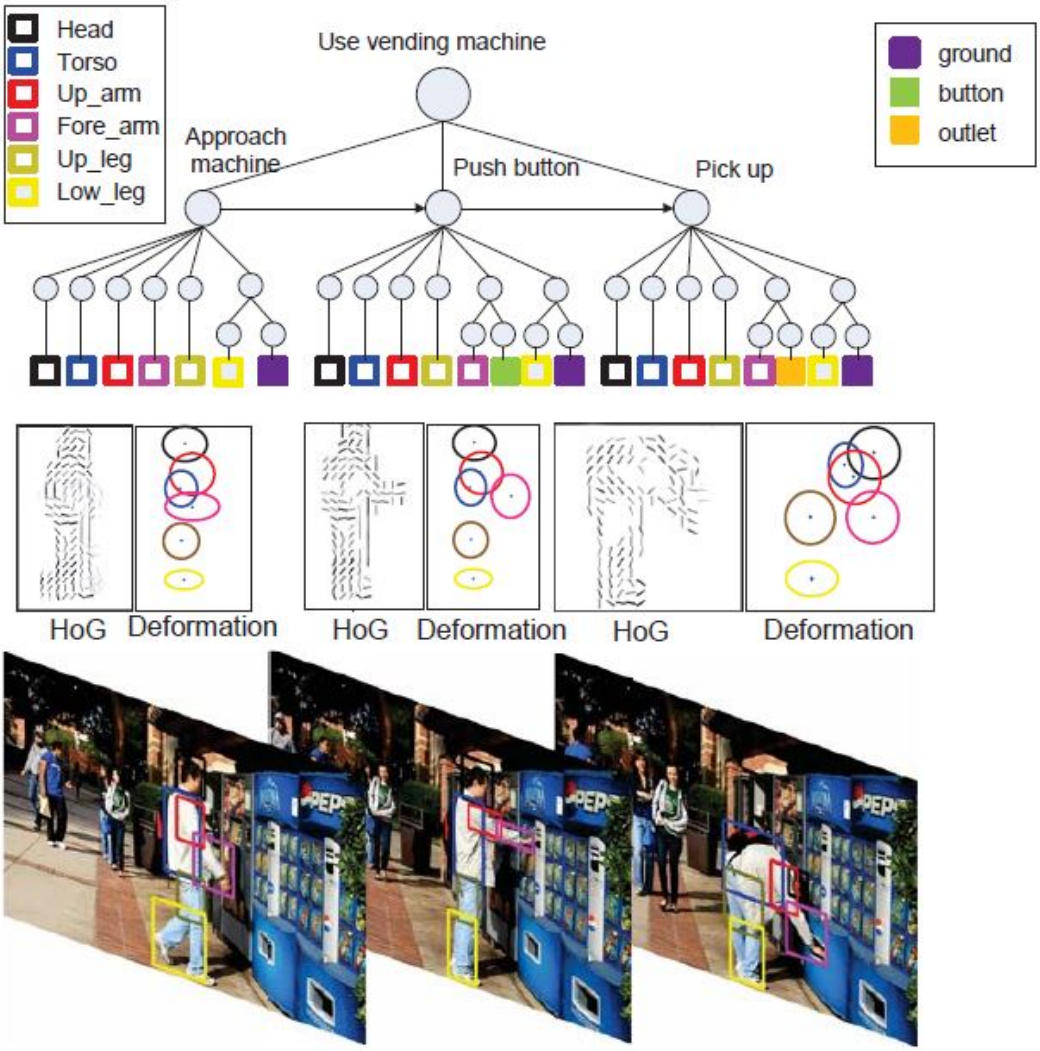

Figure 2.3: The APT includes three MPTs: 1) walking on the 'ground' to approach the vending machine; 2) pushing 'button' at the vending machine; 3) picking up the merchandise at the 'outlet'. In the And-Or structure, open squares are the body parts and solid squares are added to represent the objects: ground (purple), button (green), outlet (yellow). These objects have spatial relations with body parts. Below the And-Or tree, we show learned HOG template for body parts and their deformations. The bottom row of the figure shows the actual detected poses on a video sequence.

based on the MPT scores and the transition probability.

All variables are included in APT thus the objective is to optimize the score function in

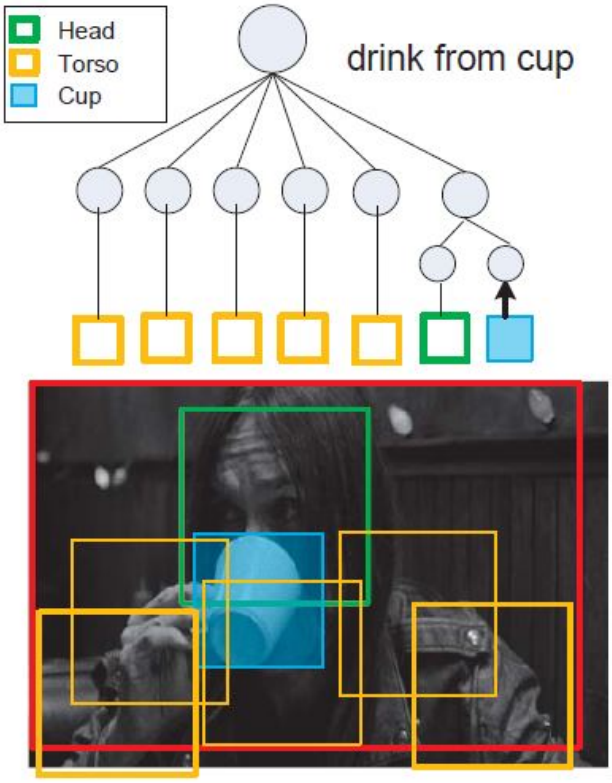

Figure 2.4: The drinking action contains six body parts and one additional part which represents the cup interacting with the head.

Eqn 2.7 for a given time interval $[t^s, t^e]$,

$$apt^*_{[t^s,t^e]} = argmaxS(apt[t^s, t^e]) \tag{2.9}$$

We may interpret the score function S as a log-posterior probability, however, it is different from the log-posterior probability in the Bayesian network because it only explains the images inside the bounding box in a certain time interval $[t^s, t^e]$.

iii) Finally we detect multiple action instances in the video. We use a greedy way to find multiple APTs each of which corresponds to an action instance. The sliding window is applied in the image pyramid over all time interval lengths to calculate MPTs, then the best APT is calculated based on the MPT candidates. After selecting the APT with the highest score, we remove all MPTs in this APT and calculate the next best APT until the score of APT is below a threshold.

2.3.2 DP for detecting MPT in space

We maximize the score function in Eqn 2.2 to detect the action snippets,

$$mpt^t = argmaxS(mpt) \tag{2.10}$$

For every 3 image frame I^t, I^{t+1}, I^{t+2}, we extract the HOG feature on the image pyramid at 18 orientations and 3 octaves similar to the DPM model in the literature. We also calculate the HOF feature from by the Lucas-Kanade [LK81]. The two optical flow maps between frames $[I^t, I^{t+1}]$ and frames $[I^{t+1}, I^{t+2}]$ are calculated respectively, then the HOF features are derived by projecting optical flows over an 8x8 image window in space and 2 frames in time. We use 8 bins for 8 directions and the value of each bin is the sum of optical flow vectors. The sigmoid function is used to normalize the value to [0,1].

Since the MPT is represented by the And-Or Tree structure, the MPT with highest score can be inferred by standard dynamic programming algorithm. The speed for detecting a single action snippets is around 2 frames per second for the video with resolution 240x320 using an i7 PC.

2.3.3 DP for detecting APT in time

At each image frame, we output several MPT candidates by the dynamic programming. Each time the algorithm will find the best MPT if score $S(mpt)$ is above the threshold τ_1, then then window of the current candidate will be blocked and the next best MPT candidate will be calculated. The threshold τ_1 is determined using the approximately admissible threshold strategy similar to [FGM10a]. It is actually the lowest score for the person to be detected in the training data.

Suppose we detect several MPT candidates $\alpha(t)$ in each time frame between t^s and t^e and n^t is the number of MPT candidates at time t. We illustrate those candidates in Figure 2.5. For those candidates, we compute two quantities for action detection: 1) the total score of the MPT candidate $\alpha(t)$: $s(\alpha(t)) = S(mpt_{\alpha(t)})$. If there is MPT detected then the score

is set to -inf which means that we do not go back to adjust the MPT detection because it is very expensive and do not improve the performance too much; 2) The transition scores between two candidates at two adjacent frames: $s(\alpha(t), \alpha(t+1)) = S(mpt_{\alpha(t+1)}|mpt_{\alpha(t)})$.

We plug the two quantities into Eqn 2.9 and get,

$$apt^*[t^s, t^e] = argmax \sum_{t=t^s}^{t^e} s(\alpha(t)) + \sum_{t=t^s}^{t^e-1} s(\alpha(t), \alpha(t+1)) \tag{2.11}$$

This can be solved by standard dynamic programming. The algorithm for detecting action is outlined below,

Algorithm 1 Action detection algorithm

Input: Video $I[t_i : t_j]$, thresholds τ_1 and τ_2;

Output: Detected actions in a list $List_A$;

Perform MPT detection on each frame and add a candidate $\mathbf{mpt}_k$ to a candidate list $List_C$ if $S(\mathbf{mpt}_k) > \tau_1$ by Eqn. (2.2)

Connect all candidates at consecutive frames $\mathbf{mpt}_i^{(t)} \leftrightarrow \mathbf{mpt}_j^{(t+1)}$ of the same action type, and compute their transition cost $S(\mathbf{mpt}_j^{(t+1)}|\mathbf{mpt}_i^{(t)})$ by Eqn. (2.6).

repeat

Find an optimal path **apt** using DP, and compute its $S(\mathbf{apt}) \Rightarrow s$ by Eqn. (2.7).

if $s < \tau_2$ **then**

return $List_A$;

else

Remove all the MPTs in **apt** from $List_C$ and add them into $List_A$

end if

until $List_C = \emptyset$.

return $List_A$;

The threshold τ_2 is determined from all positive training examples. For each action, τ_2 is the lowest score of the positive APT on the positive example so that it does not prune any optimal configuration.

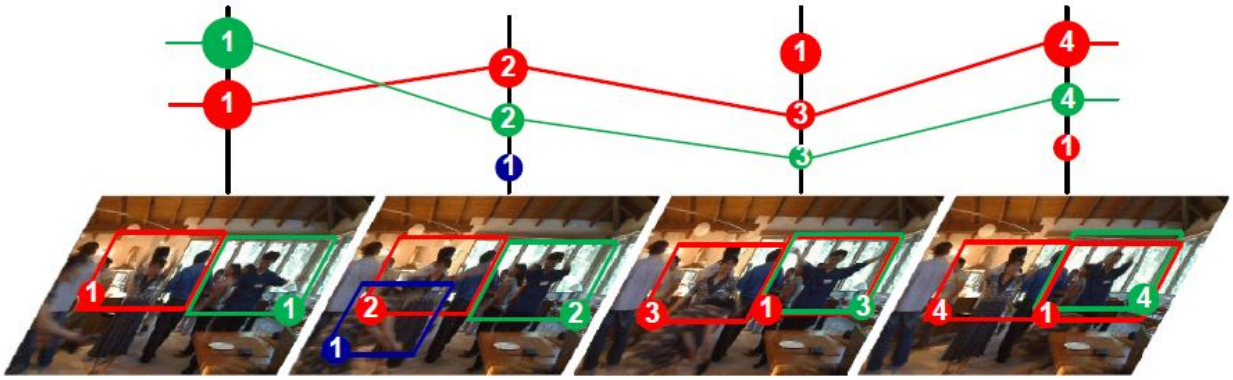

Figure 2.5: At each frame we detect several MPT candidates which are visualized with colored balls. Green stands for 'clapping', red stands for 'hand waving' and blue is 'boxing'. The lines connecting balls are detected APTs.

2.4 Learning

We employ the semi-supervised structure SVM method for learning the modeling parameters in MPT and APT. Suppose we have the first n frames annotated with structured labels $y_i, i = 1, 2, ..., n$ where the label y includes the action label, MPT label and bounding boxes for the whole person and body parts, thus the labels of remaining frames are hidden: $h_i, i = n+1, n+2, ..., N$. We use D to denote our training data, $D = (\{x_i, y_i\}_{i=1}^{n}, \{x_i, h_i\}_{i=n+1}^{N})$ where x is the feature at frame i including HOG and HOF features.

The learning algorithm includes three steps in the following,

i) **Initialize the MPT and APT models.** We cluster the annotated frames to get the dictionary of MPT using EM in the joint space of HOG and HOF features. Each MPT corresponds to a pose template with a certain viewpoint, part configuration and motion velocity. With MPT clusters, we also initialize the transition probabilities between two MPT $A(l^{t+1}|l^t)$.

ii) **Train the MPT parameters by structure SVM.** With annotated frames and MPT labels, we train the MPT parameters $\omega = (\omega^A, \omega^D, \omega^M)$ for appearance, deformation and motion by structure SVM. We treat it as a multi-class classification problem and minimize the following function,

$$\min_{\omega} \frac{1}{2}\|\omega\|_2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \tag{2.12}$$
$$s.t. \quad \max_{\hat{y}\in\mathcal{Y}} \omega^T(\phi(x_i, y_i) - \phi(x_i, \hat{y}_i)) \geq \Delta(y_i, \hat{y}_i) - \xi_i,$$

This optimization problem can be solved by the cutting-plane algorithm [TJH05].

iii) **Train the APT model by semi-supervised structure SVM.** We add unlabeled frames into the training process and train the transition probabilities between MPTs. The latent SVM learning process optimize the following objective function,

$$\omega = \arg\min_{\omega\in\mathcal{R}^d} \left(\frac{1}{2}||\omega||_2 + \sum_{i=1}^{n} g(x_i, y_i; \omega) + \sum_{i=n+1}^{N} \max_{h\in\mathcal{Y}} g(x_i, h_i; \omega) \right) \tag{2.13}$$

where $g(x, y; \omega)$ is the upper bound function for the risk of the latent structure SVM,

$$g(x, y; \omega) = max\{0, \Delta(y, \hat{y}) + \omega^T(\phi(x, \hat{y}) - \max_{h^*} \phi(x, h^*))\} \tag{2.14}$$

The Eqn 2.13 is a sum of convex and concave function and the local optimal can be found by the CCCP procedure iteratively.

In experiments we find that it is not good to add all unlabeled examples into the training process at the beginning because that the incorrect label may lead to a very bad local minimal. Inspired by the curriculum learning strategy in [KPK10], we gradually add high scored examples into training and update the parameters ω. This helps the algorithm to converge to a good local minimum smoothly.

2.5 Experiments

2.5.1 Datasets

In this section, we evaluate our method on four datasets, three of which are public datasets: the KTH dataset [SLC04], Microsoft Research Action II dataset [CLH10] and The Coffee & Cigarette dataset [LMS08]. The fourth dataset is collected by ourselves at UCLA campus

and can be downloaded at: http://vcla.stat.ucla.edu/dataset/animated pose.html. Note that some datasets do not provide the bounding box annotations for body parts, therefore, we manually add the annotations to all training data.

2.5.2 Action Classification on KTH dataset

The KTH dataset has six types of human actions and each action is performed multiple times by 25 actors. We use the same experimental setting as in [SLC04], so 16 of 25 persons are used for training and the rest 9 are used for testing. The training data has total 2391 action clips. We only test the classification accuracy because the background of this dataset is quite clean, thus the detection is very easy.

We use 40 key-frames for each training video clip. 10 parts are annotated in each key-frame: head, torso, upper/lower arms and upper/lower legs. We cluster the key-frames to get 3 MPTs for each action class. We also test APT model with latent parts instead of annotated parts. The appearance and motion template for each part are initialized in the same way as DPM [FGM10b].

The results are reported in Table 2.1. Clearly the latent parts do not perform as well as the annotated parts which we believe is due to the bad local minimum by poor initialization. With only 20 annotated key-frames, our method achieves the best performance. The performance is not improved much with doubled number of annotations.

2.5.3 Action Detection on the MSR dataset

The MSR dataset has 54 video clips and three actions: hand waving, clapping and boxing. These videos have cluttered background such as walking people and outdoor traffic. Multiple action instances may happen in one video clip, therefore it is necessary to localize the action of interest in each frame. The bounding boxes of person are annotated in all videos. Only 6 upper body parts are annotated because the three action are only related to upper body. We follow the same experiment setting as in Cao et al. [CLH10] so half of the videos are used for training and the rest half for testing.

Supervision	Work	Average
weakly-sup	Schuldt *et al.* [SLC04]	71.71%
weakly-sup	Dollar *et al.* [DRC05]	80.66%
weakly-sup	Niebles and Fei-Fei [NWF08]	83.92%
weakly-sup	Laptev *et al.* [LMS08]	91.81%
weakly-sup	Liu and Shah [LLS09]	94.16%
weakly-sup	Cao *et al.* [CLH10]	95.02%
weakly-sup	**APT with latent parts**	84.70%
semi-sup	**APT, 10 annotated frames**	92.70%
semi-sup	**APT, 20 annotated frames**	94.24%
semi-sup	**APT, 40 annotated frames**	94.53%

Table 2.1: Comparison of classification on the KTH dataset.

We first evaluate the detection performance for action snippets using moving pose templates. As shown in Figure 2.6(a), the action 'boxing' achieves the best performance because the poses of 'waving' and 'clapping' are quite similar.

Secondly we evaluate the detection performance for action clips. We use a sliding window with 15 frames to search for the action instances. Following the same criterion in [CLH10], we denote the bounding boxes for a ground truth action as $Q^g = Q_1^g, Q_2^g, ..., Q_m^g$ and for a detected action as $Q^d = Q_1^d, Q_2^d, ..., Q_n^d$. We use $HG(Q_i^g)$ to denote if the ground truth Q_i^g is detected, and $TD(Q_j^d)$ to denote if the detected Q_j^d is correct,

$$HG(Q_i^g) = \begin{cases} 1, & \exists Q_k^d, s.t. \frac{|Q_k^d \cap Q_i^g|}{|Q_i^g|} > \delta_1 \\ 0, & \text{otherwise} \end{cases}$$

$$TD(Q_j^d) = \begin{cases} 1, & \text{if } \exists Q_k^g, s.t. \frac{|Q_k^d \cap Q_j^d|}{|Q_j^d|} > \delta_2 \\ 0, & \text{otherwise} \end{cases}$$

where $|.|$ denotes the area of the bounding box and δ_1, δ_2 are set to $1/4$. Based on HG and TD we can compute the precision and recall for action detection. The precision-recall curve is reported in Figure 2.6 (b). Our APT model outperforms others when full supervision is used. We also test the performance when using different amount of annotated key-frames. The

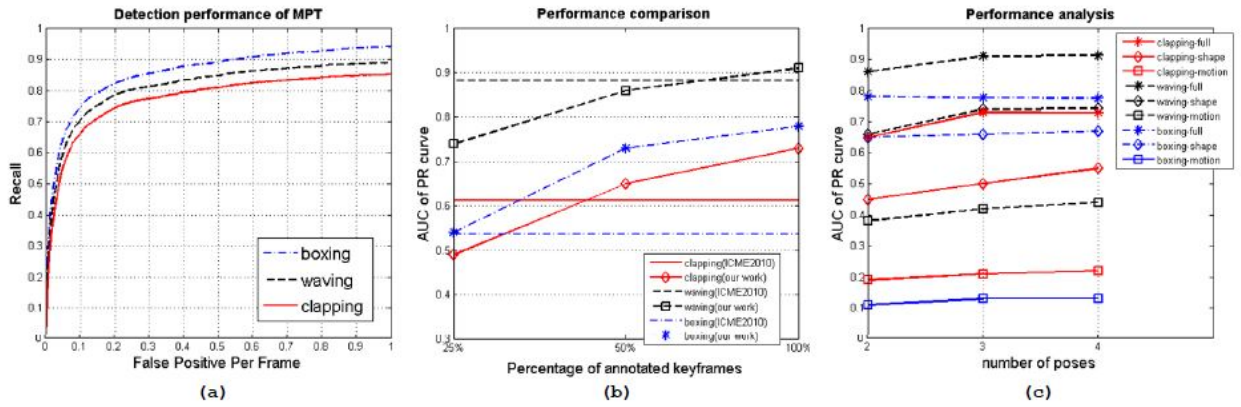

Figure 2.6: (a) Detection performance for action snippets using MPT model.(b)Performance comparison with different amount of annotated key-frames used for initialization. (c) Performance comparison with different number of MPTs used for each action class. The comparison between final system and using only shape or motion template are also included.

improvement with more annotations is diminishing which indicates that 50% of annotations are enough to get a good result and more annotations are not very helpful. As shown in Figure 2.6 (c), we consider the effect of the number of poses. Using only two MPTs hurts the performance a lot, but using 4 or 5 MPTs does not improve the performance in all actions, therefore we use 3 MPTs in the experiments. We also compare the effect of different features in Figure 2.6 (c). From the results we can see that the shape template achieve better results than motion template. It is worth noting that the shape template is much more important than motion template in the action 'boxing'. With only shape templates, our method already outperforms others.

2.5.4 Coffee and Cigarette dataset

The Coffee & Cigarette dataset is collected from a movie and has 11 short videos with different scenes and actors. There are two action classes: 'drinking' and 'smoking'. For the action drinking, there are 106 training samples and 38 testing samples. For the action smoking, there are 78 training samples and 42 testing samples. For each example, we annotate 40 key-frames with 6 upper body parts and one contextual object.

Figure 2.7: We only show the red bounding boxes for body head and blue bounding boxes for the contextual objects.

We first show some detection results in Figure 2.7. The action instance is correctly detected if the predicted spatial-temporal volume overlaps with the ground-truth volume at least 20%. The average precision of our method and others are reported in Table 2.2.

Three method variations are compared: 1) only using MPT without transitions between MPTs. 2) using the APT with latent parts instead of annotated parts. 3) using APT with annotated contextual parts. The results show that our APT model with objects achieves better performance than others on this task. The same as the MSR dataset, the APT model with latent parts performs worse than the strong supervision with annotated parts. Without supervision, the object cannot be detected well during the learning process because they have larger appearance variations than body parts.

2.5.5 UCLA contextual action detection dataset

Our dataset consists of videos of 10 scenes taken from everyday living places such as campus plaza, food court and so on. Each video contains about 10 instances from 6 action categories: purchasing from a vending machine, using elevator, throwing trash into a can, using water dispenser , picking up newspapers from a paper stand and sitting down on a chair then get up and leave. There are six body parts annotated in each frame: 'head', 'torso', 'upper-arm',

	Drinking	Smoking
MPT w/ contextual parts	29%	14%
APT w/ contextual parts	58%	31%
APT w/ latent parts	43%	26%
Laptev *et al.* [LP07]	43%	-
Willems *et al.* [WBT09]	45%	-

Table 2.2: Results on Coffee & Cigarettes. AP performance for spatio-temporal localizations in percent. The first two rows report the performance of our algorithm. The remaining results from recent literatures.

'lower-arm', 'upper-leg' and 'lower-leg'.

To minimize the effect of over-fitting, we apply a 5-fold cross-validation by randomly choosing different combinations of training and testing actions. The average detection precision is measured for six event classes as shown in Table 2.3. Since the APT with latent parts method does not use the contextual information, it is much worse than the full APT model.

Event	APT-full	APT w/ latent parts
vending machine	82%	43%
elevator	92%	67%
throw trash	86%	58%
water dispenser	87%	62%
news-stand	89%	74%
sit down then get up	90%	66%

Table 2.3: Detection performance on UCLA vending machine dataset.

2.6 Summary

In this chapter, we study the joint representation of action and pose using moving pose template (MPT) which models a short duration action snippets and animated pose template (APT) which is based on a sequence of MPTs and models a long duration action. Each MPT contains a shape template capturing appearance feature and a motion template capturing motion features. Due to the And-Or tree structure of MPT and the chain structure of APT, the dynamic programming us used for effective inference. The parameters are learned discriminatively through latent structure SVM. This method has two main drawbacks and can be improved in the following aspects: 1) it is view-dependent. For different views, more pose templates are needed. The model which is capable of recognizing actions under unseen viewpoints is needed. 2) The temporal transition is modeled on the level of MPT which is too rough to capture the important motion information with body parts. A deeper And-Or Graph representation is needed for modeling the detailed motion changing and appearance variations.

CHAPTER 3

Cross-view Action Modeling, Learning and Recognition

3.1 Introduction

3.1.1 Backgrounds and Motivations

In the research area of video-based action recognition, most existing works focus on recognizing actions from viewpoints which are similar to training data, therefore their general limitation is the unpredictable performance in the situation where the actions need to be recognized from a novel view. Due to the large difference between the visual appearances from different views and the difficulty to find view-invariant features, it is desirable to build models for cross-view action recognition, i.e., recognizing video actions from views that are unseen in the training videos. Despite some recent attempts [JDL08, LSK11], this problem has not been well explored.

One possible approach for cross-view action recognition is to enumerate a sufficiently large number of views and build dedicated feature and classifier for each view. Clearly this approach is too time consuming because it requires annotating a large number of videos for all views and all action categories. Another possible approach is to interpolate across views via transfer learning [LSK11]. This method learns a classifier from one view, and adapts the classifiers to new views. The performance of this approach is largely limited by the discrimination power of the local spatio-temporal features in practice.

In this section, we tackle this problem from a new perspective: creating a cross-view video action representation by exploiting the compositional structure in spatio-temporal patterns and geometrical relations among views. We call this model multiview spatio-temporal AND-

OR graph model (MST-AOG), inspired by the expressive power of AND-OR graphs in object modeling [SZ13]. This model includes multiple layers of nodes, creating a hierarchy of composition at various semantic levels, including actions, poses, views, body parts and features. Each node represents a conjunctive or disjunctive composition of its children nodes. The leaf nodes are appearance and motion features that ground the model to images. An important feature of the MST-AOG model is that the grounding does not have to be at the lowest layer (as in conventional generative models), but can be made at upper layers to capture low resolution spatial and temporal features. This compositional representation models geometry, appearance, and motion properties for actions. Once the model is learned, the inference process facilitates cross-view pose detection and action classification.

The AND/OR structure of this MST-AOG model is intuitive and simple, but the major challenges lie in the learning of geometrical relations among different views. This paper proposes novel solutions to address this difficult issue by taking advantage of the 3D human skeleton produced by Kinect sensors. This 3D skeleton information is only needed for training, and not used for cross-view action recognition. The projection of the 3D poses enables explicit modeling of the 2D views. Our model uses a set of discrete views in training to interpolate arbitrary novel views in testing. The appearances and motion are learned from the multiview training video and the 3D pose skeletons.

In order to learn the hierarchical structure of our MST-AOG, we design a new discriminative data mining method to discover the discriminative poses for each action automatically. This data driven method provides a very effective way to learn the structure for the action nodes. Since this hierarchical structure enables information sharing (e.g., different view nodes share certain body part nodes), MST-AOG largely reduces the enormous demands on data annotation, while improving the accuracy and robustness of cross-view action recognition, as demonstrated in our extensive experiments.

3.1.2 Contributions

Compare to other existing methods for action recognition, our proposed cross-view action recognition method makes the following contributions:

- The proposed spatial-temporal and-or graph structure is a expressive and compact model for representing action with multiple viewpoints and unifies the modeling of geometry, appearance and motions.
- The human skeleton information is not need in testing. Once trained, our model only needs 2D videos to recognize actions from novel views.
- We propose a novel and effective methods to learn the nodes in our MST-AOG model discriminatively. The different upper-level nodes share some low-level nodes which enables effective computation.

3.2 Multi-view Spatial Temporal And-Or Graph (MST-AOG)

3.2.1 Model Overview

We propose a spatial-temporal And-Or Graph model for representing multi-view actions. As a multi-layer hierarchical compositional model, it is able to compactly accommodate the combinatorial configurations for cross-view action modeling. There are three types of nodes in our ST-AOG and each node is associated with a score: 1) The AND node models the conjunctive relationship of its children nodes, therefore the score takes the summation over its children nodes. 2) The OR node captures the disjunctive relationship or the mixture of possibilities of its children nodes, thus its score takes the maximum over its children. 3) The LEAF node or terminal node is observable and is associated directly with images, thus grounds the model. We visualize the structure of model in Figure 3.1. The root node on the top layer is an OR node representing the mixture of the set of different actions. Each action is decomposed into a sequence of discriminative 3D poses. A 3D pose exhibits a mixture of its projections on a set of 2D views. A 2D view includes a set of spatio-temporal parts, and

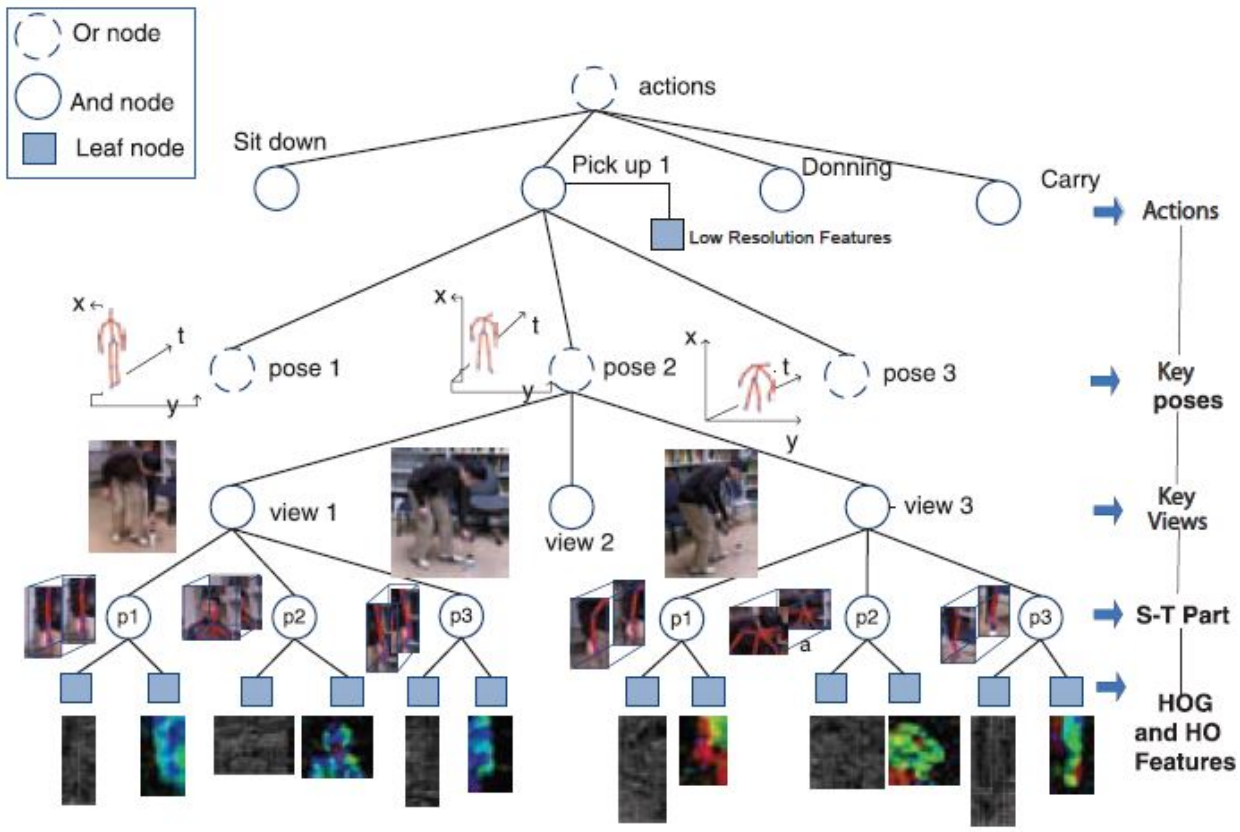

Figure 3.1: The MST-AOG action representation. The geometrical relationship between parts in different views are modeled jointly by projecting the 3D poses into the given view. The parts are discriminatively mined and shared by all actions.

each part is associated with its appearance and motion features. Thus, the action nodes, view nodes and part nodes are AND nodes, and pose nodes are OR nodes. We will discuss the scores and parameters for these nodes in the following subsections.

The strong expressive power of an AOG lies in the structure of layered conjunctive and disjunctive compositions. Moreover, MST-AOG shares the part nodes across different views via interpolation. An example will be given when discussing the action node in Section 3.2.4.

3.2.2 Pose/View Nodes and 3D Geometry

We first introduce the pose and view nodes in our MST-AOG for cross-view action modeling. A pose node is an OR node that models the association of spatio-temporal patterns to a 3D

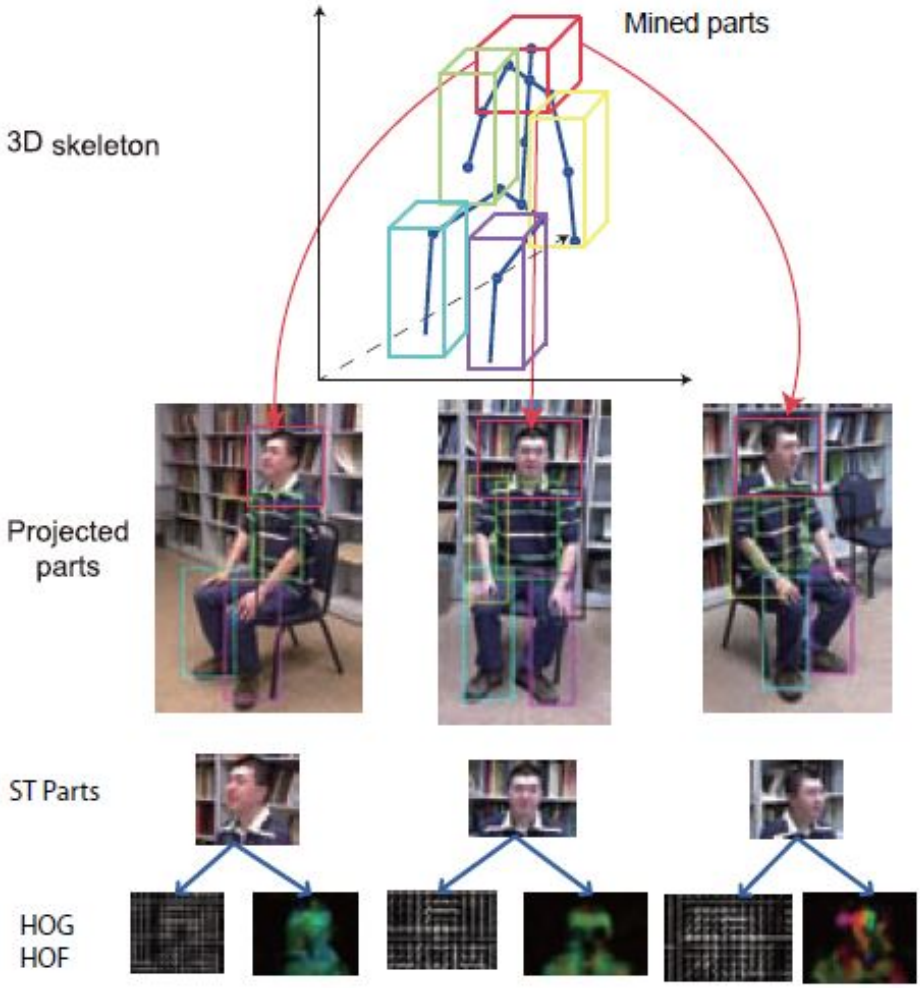

Figure 3.2: 3D parts and projected parts in different views.

pose which can be projected to various views each of which is represented by a view node. For each view node, it captures the AND relationship of a number of parts (i.e., the limb of the human). Each part node captures its visual appearance and motion features under a specific view θ. Specifically, we use a star-shaped model for the dependencies among body parts, inspired by the DPM model [FGM10b], as Figure 3.2 shows. Their 2D locations are denoted by $V = v_0, v_1, , v_N$, where v_0 is for the root part (the whole pose). I denotes the image frame. We define the score associated with the i_{th} part node to be $S_R(v_i, I, \theta)$.

Two factors contribute to the score of a view node: the score of its children part nodes $S_R(v_i, I, \theta)$ and the spatial regularization among them $S_i(v_0, v_i, \theta)$ that specifies the spatial

relationship between the root part and each child part. Such spatial regularization measures the compatibility among the parts under view θ. Here we only consider the rotation angle. The total compatibility score of a view node is written as:

$$S_V(v_0, \theta) = \sum_{i=0}^{N} S_R(v_i, I, \theta) + \sum_{i=1}^{N} S_i(v_0, v_i, \theta) \tag{3.1}$$

where v_i denotes the location of part i, N is the total number of parts and θ indicates the current view. The 2D global location of a 2D pose is set to be the location of the root part v_0. As the pose node is an OR node, the score for a pose node is computed by maximizing the scores from its children nodes which are view nodes:

$$S_p(v_0) = \max_{\theta} S_v(v_0, \theta) \tag{3.2}$$

The evaluation of the spatial regularization of the parts needs a special treatment, because a pose node represents a 3D pose and it can be projected to different views which lead to different part relationships explicitly, as illustrated in Figure 3.2.

The 3D geometrical relationship of the parts is modeled as the 3D offsets of the i_{th} part u_i with respect to the root part u_0. Each offset can be modeled as a 3D Gaussian distribution with the mean u_i as well as diagonal covariance matrix Σ_i.

$$logP(\Delta_{p_i}) \propto -\frac{1}{2}(\Delta_{p_i} - u_i)^T \Sigma_i^{-1} (\Delta_{p_i} - u_i)) \tag{3.3}$$

where $\Delta_{p_i} = (\Delta_{x_i}, \Delta_{y_i}, \Delta_{z_i})$ is the 3D offset between the part i and root part. Here u_i can be estimated from the 3D training data captured by Kinect sensor. The Σ_i can also be learned from training data.

We project the 3D parts into 2D parts under a certain viewpoint. The projected 2D parts also follow a Gaussian distribution. Here we assume the projection is scaled orthographic and denote it by Q_i^{θ},

$$Q_i^\theta = \begin{bmatrix} k_1 cos\theta & 0 & -k_1 sin\theta \\ 0 & k_2 & 0 \end{bmatrix} \tag{3.4}$$

where θ is the rotation angle under the view, and k_1, k_2 are scale factors for two image axes. In training, we take advantage of the 3D skeleton data from Kinect sensors. Since we have the ground truth 3D positions of parts and their projected 2D positions, these parameters can be easily estimated. We use the orthographic projection to approximate the perspective projection because that the actors are sufficiently far way from the camera when performing actions. Since Q_i^θ is linear transformation, the projected 2D offset also follows a Gaussian distribution with mean $v_i^\theta = Q_i^\theta u_i$ and variance-covariance matrix $\Sigma_i^\theta = Q_i^\theta \Sigma (Q_i^\theta)^T$. Thus the 2D spatial pairwise relationship score $S_i(v_0, v_i, \theta)$ can be written as,

$$\begin{aligned} S_i(v_o, v_i, \theta) = \quad & ((\Sigma_i^\theta)_{11}^{-1}, (\Sigma_i^\theta)_{22}^{-1}, (\Sigma_i^\theta)_{12}^{-1})^T \cdot \\ & (-(\Delta v_i^x)^2, -(\Delta v_i^y)^2, -2\Delta v_i^x \Delta v_i^y) \end{aligned} \tag{3.5}$$

where $(\Delta v_x = v_i^x - v_0^x, \Delta v_y = v_i^y - v_0^y)$ is the 2D deformation between i-th part v_i and the root part v_0. This 3D geometrical relationship is shared and can be learned cross different views. The 2D geometrical relationship in novel views can be obtained by projecting the 3D geometrical relationship to the novel views under the rotation angle θ.

3.2.3 Part Node and Motion/Appearance

The spatial-temporal patterns of a part under a view are captured by its motion and appearance features. Each part has an appearance node with score $A_i(v_i, I, \theta)$, and a motion node with score $M_i(v_i, I, \theta)$. They capture the likelihood (or compatibility) of the appearance and motion of part i located at v_i under view θ at image I respectively. Thus the score associated with a part node is written as,

$$S_R(v_i, I, \theta) = A_i(v_i, I, \theta) + M_i(v_i, I, \theta) \tag{3.6}$$

We apply the commonly used HOG [FGM10b] and HOF [LMS08] features to represent the appearance and motion of a given part respectively. In order to model the difference and correlation of the appearance and motion for one part in different views, we discretize the view angle into M discrete bins each of which corresponds to a view node. We use exponential interpolation to obtain the appearance and motion features in the view bins which do not appear in training data. The appearance score function $A_i(v_i, I, \theta)$ is defined as,

$$A_i(v_i, I, \theta) = \frac{\sum_{m=1}^{M} e^{-d^2(\theta,\theta_m)} \phi_{i,m}^T \phi(I, v_i, \theta)}{\sum_{m=1}^{M} e^{-d^2(\theta,\theta_m)}} \tag{3.7}$$

where $e^{-d^2(\theta,\theta_m)}$ denotes the exponential angular distance between the view θ and the view at bin m, $\phi(I, v_i, \theta)$ is the HOG feature at location v_i in image I under the view θ. $\phi_{i,m}$ is the HOG template of view bin m, and can be learned from the training data with view at bin m. The motion score function $M_i(v_i, I, \theta)$ is defined and learned in the same way as the appearance features.

The part nodes of different actions are shared across different views via interpolation, therefore we can learn the appearance and motion templates of the part nodes for the novel views via interpolation.

3.2.4 Action Node

In the second layer of our ST-AOG model, an action is decomposed into a number of N_p 3D discriminative poses, however, it is insufficient for an action node to include only a set of pose nodes for two reasons. First, when the image resolution of the human subject is low, further decomposing the human into body parts is not plausible, as detecting and localizing such tiny body parts will not be reliable. Instead, low resolution visual features may allow the direct detection of rough poses. Suppose we have N_L low resolution features, denoted by $\phi_i, i = 1, 2, , N_L$. We simply use a linear prediction function $\sum_i^{N_L} \omega_i^T \varphi_i$ to evaluate low-resolution-feature action prediction score. The weights ω_i can be learned for each low-resolution features. We use two low-resolution features: intensity histogram and size of the

bounding boxes of the foreground.

Therefore, an action node consists of two types of children nodes: a N_p number of pose nodes and a N_L number of leaf nodes for low-resolution grounding. With low-resolution features, we rewrite the score of an action node as,

$$S_A(l) = \sum_{i}^{N_p} S_P^i(v_0) + \sum_{i}^{N_L} \omega_i^T \varphi_i \tag{3.8}$$

where $S_p^i(v_0)$ is the score of the i-th pose node as we defined in Equation 3.2. ω_i is the weights for the low-resolution features φ_i.

3.3 Inference

Given an input video from a novel view, the inference of MST-AOG calculates the scores of all the nodes to achieve cross-view action classification. Since this MST-AOG model is tree-structured, inference can be done effectively via 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.

3.3.1 Cross-View Pose Detection

The states of the pose nodes, view nodes, and part nodes are their locations and scales. The score for a view node is defined in Equation 3.1, and the score for a pose node is defined in Equation 3.2. The inference of a pose node is simply comparing the scores of all the child view nodes at each location and scale, and finding the maximum score.

For a view node, since the score function in Equation 3.1 is convex, we can maximize the score in terms of the locations of the parts $v_0, v_1, ..., v_N$ very efficiently using distance transform [FGM10b]. The inference step can be achieved by convolving HOG and HOF features of the input image with the appearance and motion templates of all parts from different views and obtain the response maps. Then for each view bin, we can compute its projected part offset relationship. Using the distance transform, we can efficiently calculate

the response map for the poses under this view bin. This also enables the estimation of the novel view by finding the view bin that has the largest view score.

3.3.2 Action Classification

We apply the spatio-temporal pyramid to represent the spatio-temporal relationship of poses and low-resolution features for action recognition. The scores of the pose nodes and the low-resolution feature nodes at different locations and frames constitute a sequence of response maps. We apply the max-pooling over a spatio-temporal pyramid. The response of a cell in the pyramid is the maximum among all responses in this cell.

We divide one whole video into 3-level pyramid in the spatial-temporal dimensions. This yields $1 + 8 + 64 = 73$ dimensional vector for each response map. Then, we can use the linear prediction function defined in Equation 3.8 to compute the score of an action. The action node with the maximum score corresponds to the predicted action. Although this representation only acts as a rough description of the spatial-temporal relationships between the poses, we find it achieves very good results on our experiments.

3.4 Learning

The learning process has two tasks. The first task is learning the MST-AOG parameters, e.g., the appearance and motion templates of each part in the part nodes, 3D geometrical relationship in the view and pose nodes, and the classification weights in the action nodes. The second task is discovering a dictionary of discriminative 3D poses to determine the structure of the MST-AOG model. We describe the details of the two tasks in the following.

3.4.1 Learning MST-AOG Parameters

Learning MST-AOG parameters for the part and view nodes can be formulated as a latent structural SVM problem. The parameters of the latent SVM include: the variance Σ_i in Equation 3.3, the appearance and motion templates in Equation 3.7.

Although we have all part locations and the view labels in the training data, since we are more interested in predicting the action rather than the precise location of each part and the view, we treat the locations of the parts v_j and the view θ as latent variables. And we apply a latent SVM to learn the our model using the labeled location of the parts and the view angle as initialization. This treatment is more robust to the noise in the training data.

For each example x_n, we have its class label $y_n \in \{-1, +1\}, n \in \{1, 2, ..., N\}$. The objective function is,

$$\min_{\omega} \frac{1}{2}||\omega||^2 + C\sum_{n=1}^{N} max(0, 1 - y_n S_p(v_0|x_n)) \tag{3.9}$$

where $S_p(v_0|x_i)$ is defined in Equation 3.2, which is the total score for example x_i. ω is the concatenation of all model weights.

The learning is done by iterating between optimizing weights ω and computing the part locations and views of the positive examples in training data.

For each key-pose, we use the samples whose distances are less than a certain threshold to this pose in the positive data as positive examples and randomly sample 5000 negative training examples from negative data. Two bootstrapping is applied for mining hard negatives during the learning process. Since the action score in Equation 3.8 is a linear function, the parameters can be easily learned by a standard linear SVM solver.

3.4.2 Mining 3D Pose Dictionary

In order to learn the structure of our MST-AOG, we propose an effective data mining method to discover the 3D poses which are discriminative to different actions. Each 3D pose is a specific spatial configuration of a subset of body parts.

3.4.2.1 Part Representation

We employ the 3D joint positions to characterize the 3D pose of the human body. The 3D skeleton information from Kinect sensor has total 21 joint positions and each join-

t j has 3 location coordinates $p_i(t) = (x_i(t), y_i(t), z_i(t))$ and a motion vector $m_j(t) = (\Delta x_j(t), \Delta y_j(t), \Delta z_j(t))$. Each joint has a visibility label $h_j(t)$ which indicates that j-th joint is visible in frame t with value 1 and invisible with value 0. The location coordinates are normalized so that they are invariant to absolute position, orientation and scale. The joints are grouped into multiple parts manually.

3.4.2.2 Part Clustering

Since the poses in one action are highly redundant, we cluster the examples of each part to reduce the size of the search space, and to enable part sharing. Let part k be one of the K parts of the person and J_k be the set of the joints of this part. For each joint $j \in J_k$ in this part, we have $p_j = (x_j, y_j, z_j), m(j) = (\Delta x, \Delta y, \Delta z)$, and $h_i \in \{0, 1\}$ as its 3D position, 3D motion and visibility map, respectively. For a certain part, given the 3D joint positions of the two examples s and r, we can define their distance:

$$\begin{aligned} D_k(s,r) = \quad & \textstyle\sum_{j \in J_k} (1 + h_{s,r}(t))(||p_j^s(t) - Sp_j^r(t)||_2^2 \\ & + ||m_j^s(t) - Sm_j^r(t)||_2^2) \end{aligned} \tag{3.10}$$

where S is a similarity transformation matrix which minimizes the distance between the part k of the example s and the example r. The term $h_{s,r}$ is a penalty term based on the visibility of the joint j in two examples: $h_{s,r}(j) = a$ if $v_s(j) = v_r(j)$ and is 0 otherwise. Since this distance is non-symmetric, we define the final symmetric distance as $\bar{D}(s,r) = (D(s,r) + D(r,s))/2$.

Spectral clustering is performed on the distance matrix. We remove the clusters that have too few examples, and use the rest of the clusters as the candidate part configurations for mining. We denote the set of all candidates part configurations for the part k as: $T_k = t_{1k}, t_{2k}, ..., t_{N_k k}$, where each t_{ik} is called a part item represented by the average joint positions and motions in the cluster.

3.4.2.3 Mining Representative and Discriminative Poses

The discriminative power of a single part is usually limited. We need to discover poses (the combinations of the parts) that are discriminative for action recognition.

For a pose P that contains a set of part items $T(P)$, with each part item in this set belonging to different part. we define the spatial configuration of a poses as the 3D joint positions and motions of all the part items in this pose.

We first compute the activations of each pose P at all videos. The activation of a pose P with configuration P_p in a video v_i is defined as,

$$a_p(i) = min_t e^{-D(P_p, P_p^t)} \tag{3.11}$$

where P_p^t is the 3D joint positions of the poses P in the t-frame of the video i, and D(...) is the distance defined in Equation 3.10. If very similar poses exist in this video, the activation is high. Otherwise, the activation is low. One discriminative pose should have large activation in the videos in a given category, while having low activation vector in other categories. We define the support of the pose P for category c as:

$$Supp_p(c) = \frac{\sum_{c_t=c} a_p(i)}{\sum_{c_t=c} 1} \tag{3.12}$$

where c is the category label for the video i. We define the discrimination of the pose p as,

$$Disc_p(c) = \frac{Supp_p(c)}{\sum_{c' \neq c} Supp_p(c')} \tag{3.13}$$

Our goal is to discover the poses with large support and discrimination, therefore these poses can distinguish different action categories well. Since adding one part item into a pose always creates another pose with lower support, i.e. $Suppp(c) < suppp'(c)$ if $T(P) \supset T(P')$. Thus we use the A prior like algorithm to find the discriminative poses. Specifically, we remove the non-maximal poses from the discriminative pose pool. For a pose P, if there

exist a pose P' such that $T(P) \subset T(P')$ and both P and P' are in the set of discriminative and representative poses, then P is a non-maximal pose.

This algorithm usually produces an excessive large number of poses, we prune the sets of discriminative poses with the following criteria. Firstly, we remove poses that are similar to each other. This can be modeled as a set-covering problem, and can be solved with a greedy algorithm. We choose a pose P with highest discrimination, and remove the poses whose distance is less than a given threshold. Secondly, we remove the poses with small validation scores for the detectors trained for these poses.

3.5 Experiments

We evaluate our method and compare it with other stat-of-the-art methods on two datasets: the MSR-DailyActivity3D dataset [WLW12] and the Multiview Action3D dataset which is collected by ourselves. In all experiments we only use the videos from a single view for testing and do not use the skeleton information and videos from multiple views.

3.5.1 Northwestern UCLA Multiview Action3D Dataset

Northwestern-UCLA Multiview 3D event dataset contains RGB, depth and human skeleton data captured simultaneously by three Kinect cameras. Ten action categories are annotated in this dataset: pick up with one hand, pick up with two hands, drop trash, walk around, sit down, stand up, donning, doffing, throw, carry. Each action is performed by 10 actors. Figure 3.4 shows some example frames of this dataset. The view distribution is shown in Figure 3.3. This dataset contains data taken from a variety of viewpoints.

The comparison of the recognition accuracy of the proposed algorithm with the baseline algorithms is shown in Table 3.1. We compare with virtual views [LZ12], Hankelet [LCS12], Action Bank [SC12] and Poselet [MBM11]. For Action Bank, we use the actions provided by [SC12] as well as a portion of the videos in our dataset as action banks. For Poselet, we use the Poselets provided by [MBM11]. We also compare our model with training one

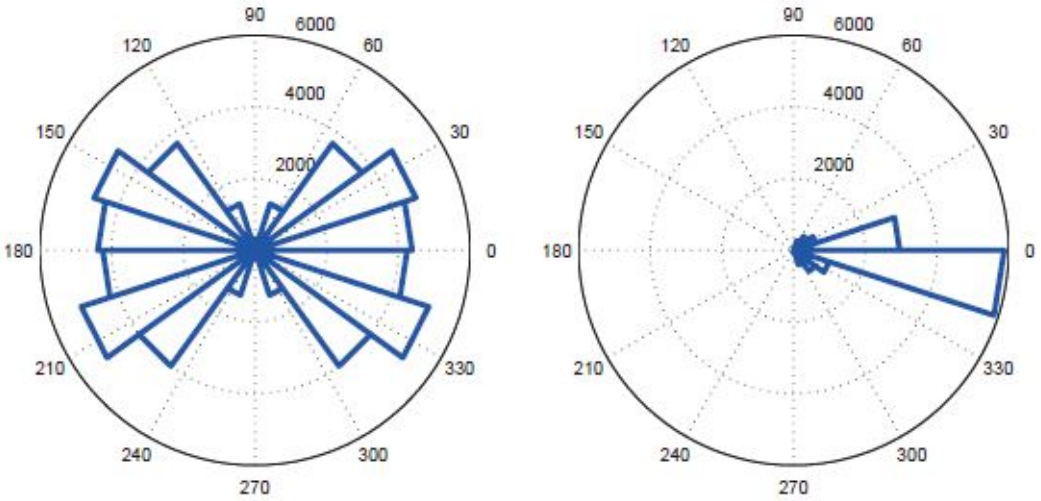

Figure 3.3: The view distribution of the Multiview-Action3D dataset(left) and MSR Daily-Activity3D dataset (right).

dedicated model for each view, which is essentially a mixture of deformable part models (DPM), to compare the robustness of the proposed method under different viewpoints with DPM model. We have 50 pose nodes for all the actions and 10 child view nodes for one pose node for both mixture of DPM and MST-AOG. The number of the part nodes in DPM and MST-AOG is both 1320 (different poses can have different number of parts). MST-AOG also has 2 child low-resolution feature nodes for each action node. These parameters are chosen via cross-validation. In MST-AOG, the appearance/motion and geometrical relationship of the part nodes are shared and learned across different view nodes, but the mixture of DPM treats them independently.

The recognition experiments are conducted under the following three settings:

1) cross-subject setting: We use the samples from 9 subjects as training data, and leave out the samples from 1 subject as testing data.

2) cross-view setting: We use the samples from 2 cameras as training data, and use the

Figure 3.4: Sample frames from Multiview Action3D dataset and MSR DailyActivity3D dataset [WLW12].

samples from 1 camera as testing data.

3) cross-environment setting: We apply the learned model to the same action but captured in a different environment. Some of the examples of the cross environment testing data are shown in Figure 3.4.

These settings can evaluate the robustness to the variations in different subjects, from different views, and in different environments.

The proposed algorithm achieves the best performance under all three settings. Moreover, the proposed method is rather robust under the cross-view setting. In contrast, although the state-of-the-art local-feature-based cross-view action recognition methods [LZ12, LCS12] are relatively robust to viewpoint changes, the overall accuracy of these methods is not very high, because the local features are not enough to discriminate the subtle differences of the actions in this dataset. Moreover, these methods are sensitive to the changes of the environment. The Poselet method is robust to environment changes, but it is sensitive to viewpoint changes. Since the mixture of DPM does not model the relations across different

Method	C-Subject	C-View	C-Env
Virtual View [LZ12]	0.507	0.478	0.274
Hankelet [LCS12]	0.542	0.452	0.286
Action Bank [SC12]	0.246	0.176	N/A
Poselet [MBM11]	0.549	0.245	0.485
Mixture of DPM	0.748	0.461	0.688
MST-AOG w/o low-S	0.789	0.653	0.719
MST-AOG w Low-S	**0.816**	**0.733**	**0.793**

Table 3.1: Recognition accuracy on Multiview-3D dataset.

view, its performance degrades significantly under cross-view setting. The comparison of the recognition accuracy of the different methods under crossview setting is shown in Figure 3.5. We also observe that utilizing low-resolution features can increase the recognition accuracy, and the proposed method is also robust under cross environment setting.

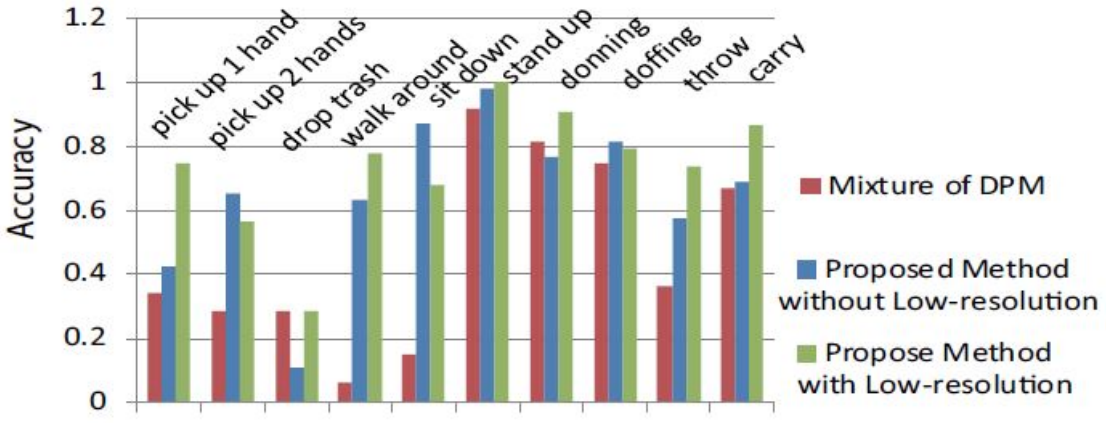

Figure 3.5: The recognition accuracy under cross-view setting on The recognition accuracy under cross-view setting.

The confusion matrix of the proposed methods with low resolution features under cross-view setting is shown in Figure 3.6. The actions that cause most confusion are pick up with one hand versus pick up with two hands, because the motion and appearance of these two

actions are very similar. Another action that causes a lot of confusion is drop trash, because the movement of dropping trash can be extremely subtle for some subjects.

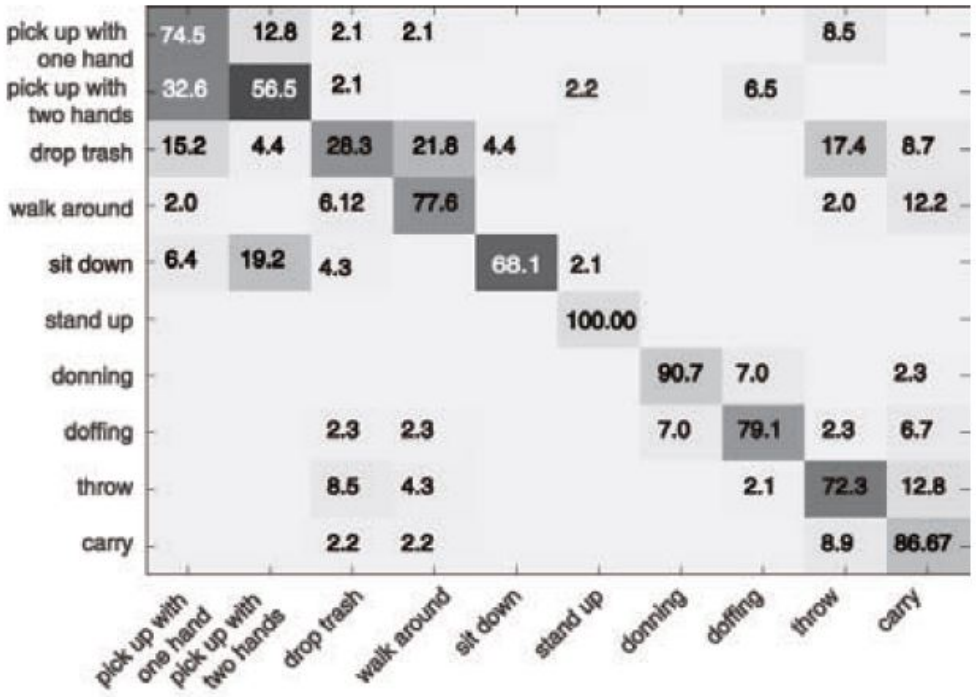

Figure 3.6: The confusion matrix of MST-AOG on multiview data under cross-view setting (with low-resolution features).

3.5.2 MSR Daily Activity3D Dataset

The MSR-DailyActivity3D dataset is a daily activity dataset captured by a Kinect device. It is a widely used as a Kinect action recognition benchmark. There are 16 activity types: drink, eat, read book, call cellphone, write on a paper, use laptop, use vacuum cleaner, cheer up, sit still, toss paper, play game, lay down on sofa, walk, play guitar, stand up, sit down. If possible, each subject performs an activity in two different poses: sitting on sofa and standing. Some example frames are shown in Figure 3.4. The view distribution of this dataset can be found in Figure 3.3. Although this dataset is not a multiview dataset, we compare the performance of the proposed method with the baseline methods to validate its performance on single view action recognition.

Method	Accuracy
STIP [LC05]	0.545
Action Bank [SC12]	0.230
Poselet [MBM11]	0.238
Actionlet Ensemble [WLW12]	0.835
MST-AOG	**0.731**

Table 3.2: Recognition accuracy for DailyActivity3D dataset. The result of [WLW12] is not directly comparable with MST-AOG, because it uses 3D skeleton.

We use the same experimental setting as [WLW12], using the samples of half of the subjects as training data, and the samples of the rest half as testing data. This dataset is very challenging if the 3D skeleton is not used. The results are reported in Table 3.2. The Poselet method [MBM11] achieves 23.75% accuracy, because many of the actions in this dataset should be distinguished with motion information, which is ignored in the Poselet method. STIP [LC05] and Action Bank [SC12] do not perform well on this dataset, either. The proposed MST-AOG method achieves a recognition accuracy of 73.5%, which is much better than the baseline methods.

Note that the accuracy of Actionlet Ensemble method in [WLW12] achieves 85.5% accuracy. However, the proposed method only needs one RGB video as input during testing, while Actionlet Ensemble method requires depth sequences and Kinect skeleton tracking during testing.

The confusion matrix of the proposed method on MSRDailyActivity3D dataset is shown in Figure 3.7. We can see that the proposed algorithm performs well on the actions that are mainly determined by poses or motion, such as stand up, sit down, toss paper, cheer up, call cellphone. However, recognizing some actions requires us to recognize objects, such as playing guitar and play games. Modeling the human-object interaction will improve the recognition accuracy for these actions.

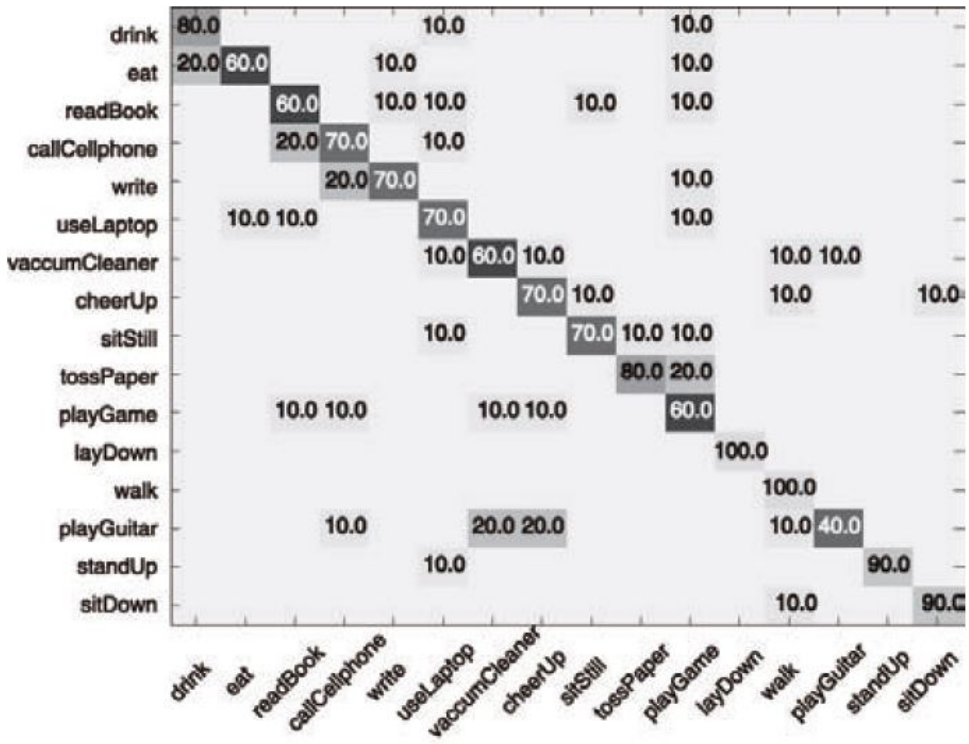

Figure 3.7: The confusion matrix of MST-AOG on MSRDailyActivity3D dataset.

3.6 Summary

In this section, We study a new cross-view action representation and propose the MST-AOG model which can effectively express the geometry, appearance and motion variations across multiple view points with a hierarchical compositional model. It takes advantage of 3D skeleton data to train, and achieves 2D video action recognition from unknown views. Our extensive experiments have demonstrated that MST-AOG significantly improves the accuracy and robustness for cross-view, cross-subject and cross-environment action recognition. The proposed MST-AOG can also be employed to detect the view and locations of the actions and poses. This will be our future work.

CHAPTER 4

Joint Pose Estimation and Action Recognition from video

4.1 Introduction

In chapter 2, we discussed our method for action detection and body part detection using moving pose template and animated pose template. In this chapter, we extend the model in chapter 2 to a more sophisticate model to incorporate appearance information at different scales and detailed motion information of fine-level body part movement. The spatial-temporal And-Or graph model is introduced to joint the learning of pose estimation and action recognition more closely.

4.1.1 Motivation and Objective

As we mentioned in our introduction, action recognition and pose estimation are important tasks for vision-based human motion understanding, and they are highly coupled because that most action are defined by poses and also the pose configuration and movement is controlled and guided by actions. Despite they have different goals, it is desirable to study them in a common framework. However, existing methods train the models for the two tasks separately and combine them sequentially.

The main drawback of existing methods is that the accuracy of action recognition relies on the performance of pose estimation because the latter always provides part locations for the former. Due to the large pose variation and complex background in action datasets, the most discriminative parts (arms, hands, legs, feet, etc.) for action recognition are always

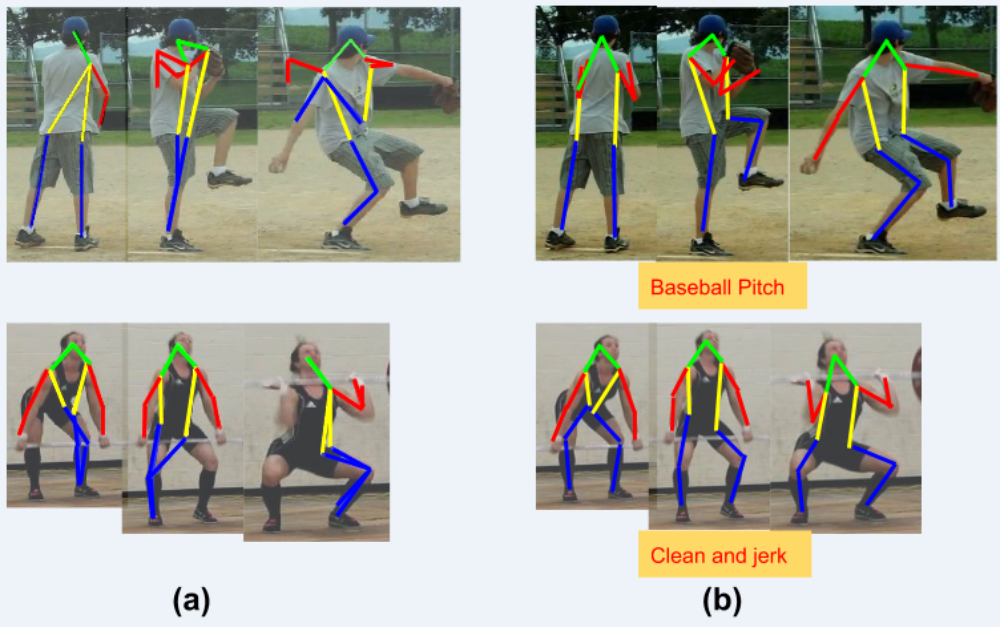

Figure 4.1: (a) Single frame human poses estimated by [YR12]. (b) The action label and poses estimated by our approach.

miss-detected in pose estimation, thus the subsequent action recognition is deteriorated by the bad body part localization result. Once the pose estimation is done, the missed body parts cannot be recovered by action recognition in this sequential way. However, those human parts often have large motions and can be recovered by action-specific motion information to some extent. Figure 4.1 shows that the arms and legs miss-detected by the pose estimation are successfully detected by our method. Besides the motion information on arms and legs, actions also provide strong priors on the movement of poses and also provide strong constraints on the possible poses in space and time.

The prevailing methods for pose estimation from static images employ probabilistic and compositional graphical models in which nodes represent body part appearance and edges represent geometrical deformation and part type compatibility. The errors of pose estimation mainly come from small parts like forearms and wrists due to the large appearance variations and similarity with cluttered backgrounds. For pose estimation in videos, the human motion can become much larger and the changing of viewpoints make appearance inconsistent at adjacent frames. As shown in Figure 4.1, we improve the estimation of part locations by

considering action specific information.

Most methods for action recognition bypass body poses and achieves promising results by using coarse/mid-level features for action classification on public datasets. In this chapter, we study the joint learning of action and pose by combing coarse/mid-level features with find-level body part features.

4.1.2 Method Overview

In this chapter, We focus on designing a common framework to integrate the training and testing of video pose estimation and action recognition. In training, the information from both tasks are utilized for optimizing the model parameters and in testing the human pose at each frame and the action label of the whole video are inferred jointly in our framework.

We start with building a spatial-temporal And-Or graph model [LWZ14, SWJ13] to represent actions and poses jointly. The hierarchical structure of our model can represent the top-down decomposition of human pose in a single frame and temporal transitions between mid-level/fine-level body parts in subsequent frames. On the top layer, the information of action in low resolution is captured by coarse-level features and then action is decomposed into poses at each frame. Each pose is further decomposed into five mid-level spatial-temporal parts (ST-parts) which can cover a large portion of human body. The five ST-parts can deform independent of each other. They are more robust to image variations than fine-level parts. All ST-parts are quantized into several components by clustering. The ST-parts in the same component can be seen as a poselet [BM09] which has small appearance and geometric variations. Each component is represented by mid-level and fine-level features.

In order to capture the specific motion information of each action and detect fine-level body parts with detailed motion, we connect ST-parts at subsequent frames in the And-Or graph and model their temporal co-occurrence relationship of part types and deformations. The model parameters at three levels are first trained separately by structure SVM (SSVM) and then combined by a mixture of experts method, therefore the features at different levels has different importance for action recognition. Due to the independence between ST-parts

of each pose, we effectively infer both action label and poses by dynamic programming.

4.1.3 Our Contributions

To our best knowledge, Yao et al. [YGG12] is the only paper trying to couple action recognition and pose estimation. They formulate the pose estimation as an optimization problem over a set of action specific manifold and conducts the two tasks iteratively, however, we represent action and human pose in the hierarchical structure and do the training and inference simultaneously. They also need training videos captured from multiple views but we can work on videos which are captured from a single view.

In this chapter we study the unified framework to combine action recognition and video pose estimation. Specifically we employ a spatial-temporal And-Or Graph model to represent the two tasks in a hierarchical manner. We make three contributions to both action recognition and video pose estimation:

i) We propose a spatial-temporal AOG model to integrate action recognition and video pose estimation. The two tasks are mutually benefit from each other in both training and inference.

ii) We represent actions at three scales. Coarse and middle level features are trained jointly with pose features, and our method give different weights to the features at different scales through the mixture of expert method.

ii) We outperform state-of-art action recognition and pose estimation methods on two action datasets: Penn Action dataset and sub-JHMDB dataset. The experiments also demonstrate the large benefit of modeling the two tasks together.

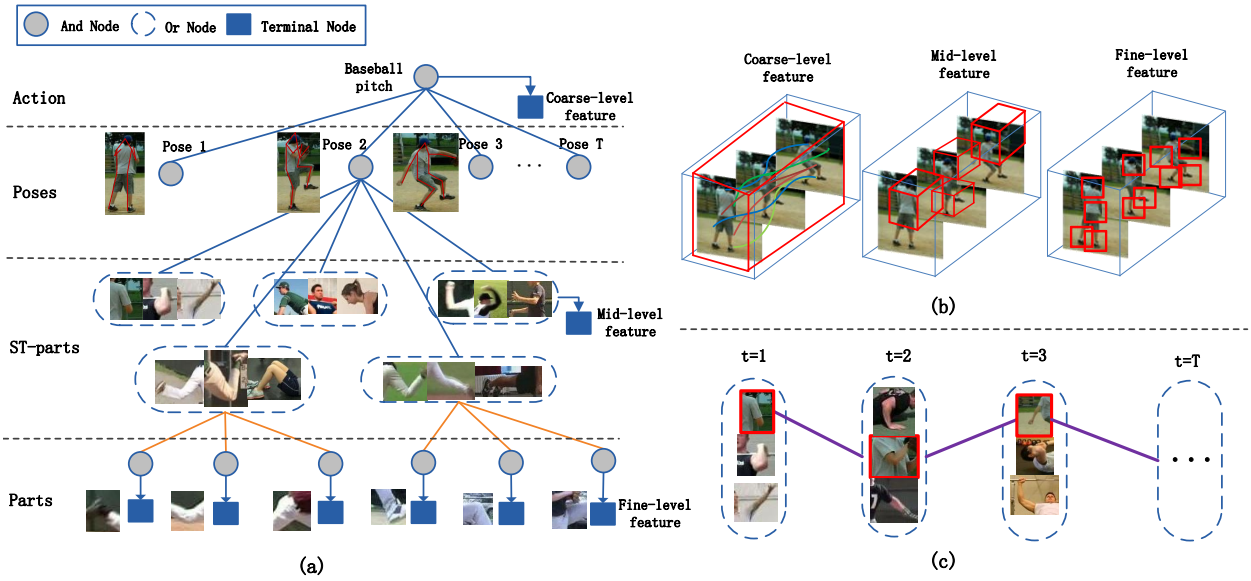

Figure 4.2: (a) Our spatial-temporal AOG model for action "Baseball pitch". The action is decomposed into poses, ST-parts, and parts. Each ST-part is an Or-Node that represents the mixture components. For simplicity we only draw all nodes at the second frame. The orange edges represent geometric deformations between ST-parts and parts. (b) The three feature levels. Action nodes, ST-part nodes and part nodes connect to terminal nodes that represent coarse-level, mid-level and fine-level features respectively. (c) An example of temporal relation on ST-part 'left arm'. The purple edges connecting five ST-parts at adjacent frames capture the temporal co-occurrence and deformation relations. During inference we select the best component (red rectangle) for each ST-part.

4.2 Representation and Modeling

4.2.1 Spatial Temporal And-Or Graph Model

Figure 4.2 shows our spatial-temporal And-Or Graph model for representing action and poses. There are three kinds of nodes: And nodes, Or nodes and Terminal nodes. The And nodes capture the decomposition of a large entity. In our case the action and poses are represented by And nodes because they are decomposed into several small entities. The Or nodes represent structural variations and alternative choices. Here each ST-part is an Or node since it could belong to different component. The Terminal node is observable and here we directly associate them with image evidence. We define three kinds of terminal nodes to represent actions at three scales. The terminal nodes associated with action and ST-parts represent coarse and mid-level features respectively. The terminal nodes at bottom level represent fine-level body part features.

In the top level of our AOG model, each action instance A is decomposed to a sequence of poses p_t,

$$A = \{p_1, p_2, ..., p_T\} \tag{4.1}$$

where T is the number of frames of this actio instance. Each pose p_t is represented by an And node and decomposed into several Spatial-Temporal mid-level parts l_i (Figure 4.2(a)),

$$p_t = \{l_1, l_2, ..., l_M\} \tag{4.2}$$

where M is the number of ST-parts of pose p_t. Each ST-part l_i is associated with its label c_i and several fine-level parts o_j,

$$l_i = \{o_0, ..., o_{N_i-1}, c_i\} \tag{4.3}$$

where $o_j = (x_j, y_j)$ denotes the location of the part which could be one of the human joints, $o_j \in \Omega_{part}, \Omega_{part} = \{'head', 'torso', 'leftarm', 'rightarm', ...\}$. N_i is the number of

fine-level parts in this ST-part and o_0 is the root part. c_i is the label of this ST-part and $c_i \in \{1, 2, ..., z_i\}$, z_i is the number of labels. The ST-parts with the same label have small appearance and deformation variations. Each ST-part also represent a movement phase of the action. The learning of ST-parts will be discussed later in this chapter.

We divide the feature vectors of the nodes in our And-Or Graph into two categories: **classification features** which focus on distinguishing different action categories and **detection features** which focus on recognizing actions from background (non-action video clips).

Classification feature includes two terms: $\psi(l_i)$ and $\psi(c_i)$. $\psi(l_i) = [d_1\, d_2 \,...\, d_{z_i}]^T$ captures the deformation of ST-parts relative to root part. $d_j = (o_0 - u_j)$ is the normalized Euclidean distance between the root part and the center of ST-part. $\psi(c_i) = [0, 0, \mathbf{1}(c_i), ..., 0, 0]$ represents the label of ST part and it is a z_i dimension indicator where the entry corresponding to component c_i is one and the others are zero.

Detection feature contains two terms: the part score $\sum_{j=0}^{N_i} S(o_j)$ and the deformation score $\sum_{j=1}^{N_i} S(o_j, o_0)$. The two scores can be directly obtained from any off-the-shelf single image pose estimation. Here we use the method from [YR12]. Those features can be treated as the regularization for action classification.

There are two types of edges capturing binary relationship in our And-Or graph model: the edges with orange color represent the geometric deformation of ST-parts/find-level parts in a singe frame and edges with purple color represent the smoothness and temporal co-occurrence of ST-parts at subsequent frames. We use three kinds of features for those binary relations:

Deformation feature is a four-dimensional vector which models the deformation between ST-part and fine-level part as a 2D gaussian distribution: $\psi(E_d) = [dx, dy, dx^2, dy^2]^T, E_d \in \Omega_D$.

Temporal co-occurrence feature at ST-part i of frame t is a $z_i \times z_i$ dimensional indicator: $\psi(E_o) = [0, 0, \mathbf{1}(c^t)\mathbf{1}(c^{t+1}), ..., 0], E_o \in \Omega_O$ which means that only the entry corresponding to labels c^t and c^{t+1} is set to one and the others are zeros.

Smoothness feature $d(l_i^t, l_i^{t+1})$ is the negation of Euclidean distance between root parts l_i^t and l_i^{t+1}. This feature is designed for giving penalty to large movement of poses at subsequent frames.

Although the action is represented by a sequence of poses, it is insufficient to only use pose features for action recognition because low resolutions makes part detection unreliable. As shown in Figure 2.2 we directly ground the action node at top-level and ST-part nodes onto image evidence which means that the coarse-level and mid-level features are also utilized for action recognition. For the coarse feature ψ_L, we follow the framework of [WKS13] to extract the bag-of-words features based on dense trajectories. For the mid-level feature ψ_M, we extract HOG and HOF features in the window of each selected ST-part and concatenate them to a long vector.

4.2.2 Score Functions

In this section we introduce the score functions of our hierarchical model in a bottom up manner from fine-level parts to ST-parts, and then pose and actions. For simplicity we ignore the action label because all score functions are the same for different action labels.

The terminal nodes in our model ground nodes to image data. Note that besides the terminal nodes at the bottom layer like previous grammar models, we also have terminal nodes at other layers for coarse-level action features and mid-level ST-part features. Instead of training fine-level part templates with action jointly, we train them independently by the off-the-shelf pose estimation method [YR12], thus the fine-level part scores and their deformation scores are obtained directly from [YR12].

We first define the score for a ST-part i,

$$S(l_i) = S_d(l_i) + S_h(l_i) + \lambda \sum_{j=0}^{N_i} S(o_j) + \lambda \sum_{j=1}^{N_i} S(o_j, o_0) \tag{4.4}$$

There are four terms contributing to the ST-part score. The first two terms are scores based on classification features and the last two terms are based on detection features.

$S_d(l_i) =< \omega_d^{l_i}, \psi(l_i) >$ measures the compatibility of deformation of current ST-part l_i and current action label. $S_h(l_i) =< \omega_h^{l_i}, \psi(c_i) >$ measures the compatibility of the ST-part label c_i and current action label. $S(o_j)$ is the score of fine-level part j and $S(o_j) = log(P(o_j))$ where $P(o_j)$ is the marginal probability from pose estimation. $S(o_j, o_0) =< \omega^{ij}, \psi(E_d^{ij}) >$ is the deformation score of part j related to the root part. Parameter λ is the weight for detection score. The inference algorithm will search all possible ST-parts in the feature pyramid and output a top candidate list at each image frame.

Each pose is consists of M ST-parts, therefore the score for a ST-part is the summation of score of its children,

$$S(p_t) = \sum_{i=1}^{M} S(l_i^t) \tag{4.5}$$

We assume that all ST-parts in each image frame are independent and we don't consider their geometric relationships. This assumption help us avoid the loopy graph structure which is a common case in video pose estimation. In inference, we use dynamic programming to infer the best ST-part sequence for each type of ST-part effectively. The details of inference is discussed in Section 4.3.

As shown in the top level of Figure 2.2, each action instance is decomposed to a sequence of full-body poses and their transitions at adjacent frames, thus the total score of an action is formulated as,

$$S_H(A) = \sum_{t=1}^{T} S(p_t) + \sum_{t=1}^{T-1} S(p_{t+1}|p_t) \tag{4.6}$$

where $S(p_t)$ is the score for ST-part p_t and $S(p_{t+1}|p_t)$ evaluate the transition between two poses. The relation between two poses is characterized by the transitions between their child ST-parts, thus it is thus written as,

$$S(p_{t+1}|p_t) = \sum_{i=1}^{M} S(l_i^{t+1}|l_i^t) \tag{4.7}$$

The transition score between two ST-parts is defined as,

$$S(l_i^{t+1}|l_i^t) = S(c_i^t, c_i^{t+1}) + \omega_{l_i} d(l_i^t, l_i^{t+1}) \tag{4.8}$$

where $S(c_i^t, c_i^{t+1}) =< \omega_o^{l_i}, \psi(E_o^{c_i^t, c_i^{t+1}}) >$ is the score based on temporal co-occurrence feature and $\omega_{l_i} d(l_i^t, l_i^{t+1})$ is the score based on smoothness feature.

We rewritten the score of the action based on notations of ST-parts,

$$S_H(A) = \sum_{i=1}^{M} (\sum_{t=1}^{T} S(l_i^t) + \sum_{t=1}^{T-1} S(l_i^{t+1}|l_i^t)) \tag{4.9}$$

From the above equation we can clearly see that the action score is only related to the ST-parts. Our inference algorithm search for the positions and labels of ST-parts to maximize this score.

4.2.3 Mixture of Experts

As we discussed in Section 4.1, using only pose features for action recognition is not robust because of the low resolution images which make part detection difficult. The Equation 4.9 can be seen as the classifier with information from poses. Here we also utilize the coarse-level feature $S_L(A)$ and mid-level features $S_M(A)$ for classifying action at low resolutions. We first train independent classifiers for $S_H(A)$, $S_M(A)$ and $S_L(A)$ and apply the mixture of experts framework on top of the three classifiers. With coarse-level and mid-level scores, the action score can be written as,

$$S(A) = \pi_L(A) S_L(A) + \pi_M(A) S_M(A) + \pi_H(A) S_H(A) \tag{4.10}$$

$S_L(A) =< \omega_L, \psi_L(A) >$ is coarse-level score and $S_M(A) =< \omega_M, \psi_M(A) >$ is mid-level score. The weights $\pi_L(A) =< \omega'_L, \phi'_L(A) >$, $\pi_M(A) =< \omega'_M, \phi'_M(A) >$ and $\pi_H(A) =< \omega'_H, \phi'_H(A) >$ are linear functions on features of action example A. Equation 4.10 will decide how much information from each classifier will be used for the action recognition based on the observable features.

4.3 Inference

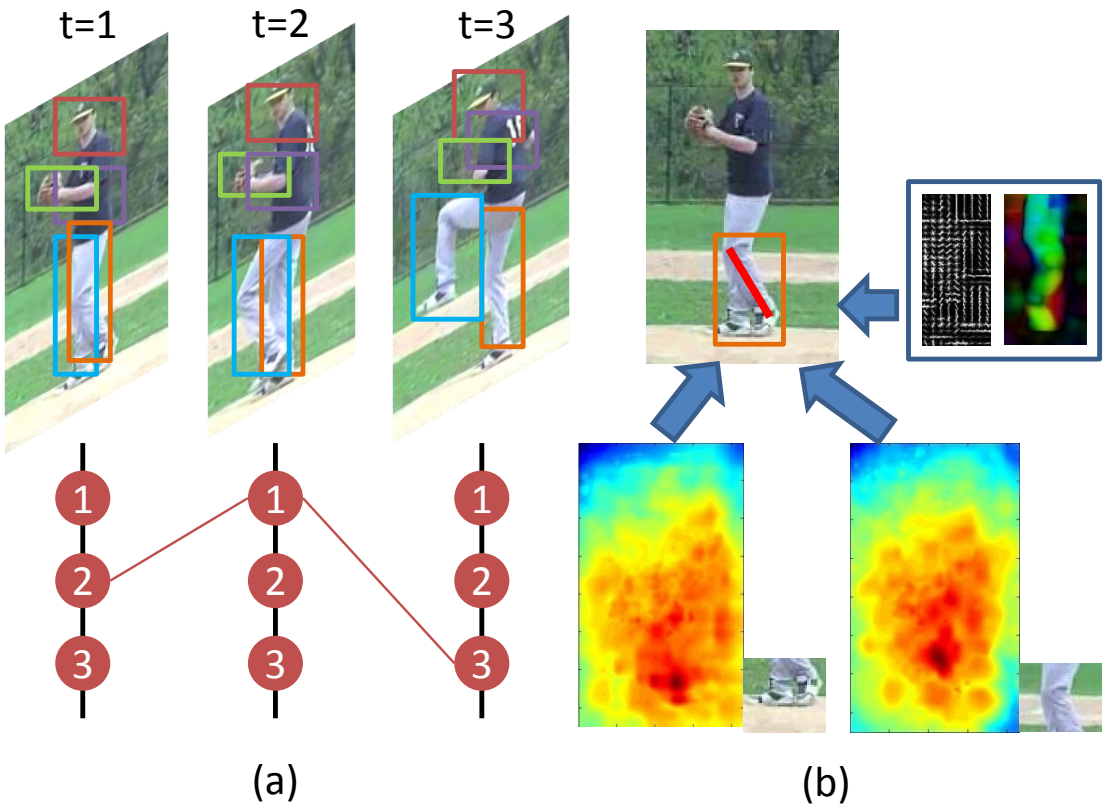

Figure 4.3: An example of our inference method. (a) For each frame we generate a list of ST-part candidates and compute the best sequence for action label by dynamic programming. (b) The ST-part is represented by the mid-level features (HOG and HOF template) and fine-level features (scores of knee and ankle).

Since we joint action recognition and pose estimation, the objective of our inference is to find the action label of the whole video and body part locations at each frame. The inference of part locations is conducted by optimizing the fine-level action score $S_H(A)$ in equation 4.9. The inference of action label is done by computer the $S(A)$ in equation 4.10. The coarse-level score $S_L(A)$ and mid-level score $S_M(A)$ are trained by linear-SVM so the inference is the dot-product between SVM weights and corresponding features. As illustrated in Figure 4.3, the fine-level action scoe $S_H(A)$ is divided into M independent terms, each of which corresponds to the summation of unary and binary scores of one spatial-temporal

part. Due to the independence of M ST-parts, we apply dynamic programming to compute the optimal sequence of each type of ST-part,

$$[l_i^1, l_i^2, ..., l_i^T] = argmax \sum_{t=1}^{T} S(l_i^t) + \sum_{t=1}^{T-1} S(l_i^{t+1}|l_i^t) \tag{4.11}$$

The above equation is repeated M times to find the total M optimal sequence for each action label. Finally the action score is computed by Equation 4.10 with coarse-level score and mid-level score.

The final action label is obtained by finding the maximum score in Equation 4.10 and we trace back to the best ST-part sequence for this action and then get all joint locations. Note that the joint 'left shoulder' can be either from the ST-part 'head shoulder' or 'left arm' and we pick it from 'head shoulder' which is easier to be detected than 'left arm'. The joint 'right shoulder' is also shared by 'head shoulder' and 'right arm' and we pick it from 'head shoulder'.

Searching ST-parts over the image pyramid is too time consuming because we need to computer pose at each image frame. As shown in Figure 4.3, to speed up computation, we first run [PR11] for each frame and compute response maps for all ST-parts. After non-maximum suppression we pick the ST-part candidates which have scores greater than τ, then we connect all candidates on consecutive frames and compute their binary transition scores. To determine the optimal threshold τ, we compute the scores on the ground truth ST-parts for all training images and pick the highest value as the threshold that does not prune any ground truth ST-part.

4.4 Learning

Our learning process contains two main stages: 1) learning type of ST-parts for each action label. 2) learning the model parameters for coarse-level, mid-level and fine-level action. Specifically we need to learn weights for unary ST-part scores , temporal transition scores between ST-parts in subsequent frames and classification weights for each action.

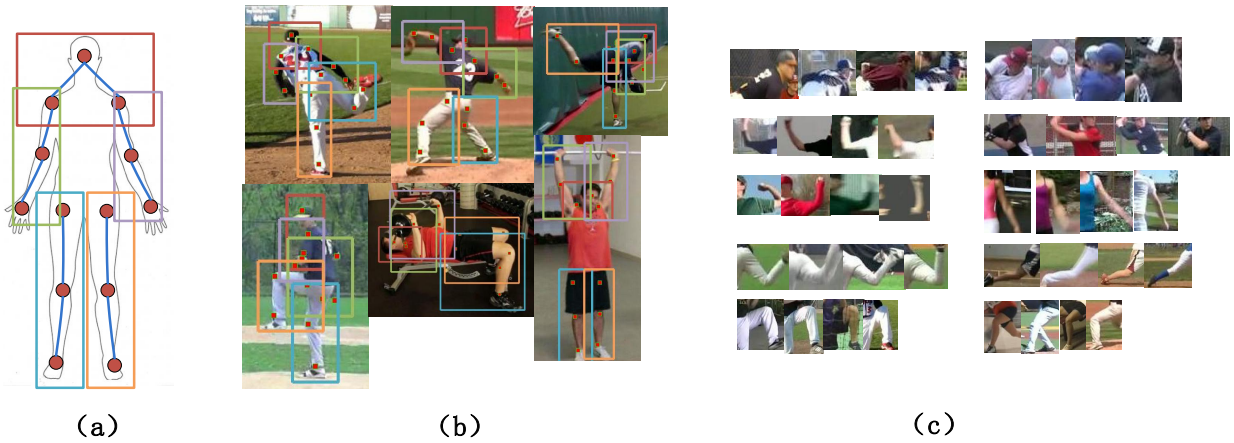

Figure 4.4: (a) The 13 parts used in our model. They are divided into five ST-parts each of which contains three parts. (b) Some examples of pose annotations in training data and their generated ST-parts. (c) Some examples for five ST-parts with two types.

As a mid-level representation of human pose, ST-parts are much more robust to image variations than fine-level parts, especially on action datasets containing large appearance, geometric and motion variations which are main challenges for detecting body parts. We have annotations of part locations in the training data and we can learn configurations of ST-parts from those annotations.

4.4.1 ST-part Representation

We use 13 fine-level parts to represent the human pose. As shown in Figure 4.4(a) the 13 parts are divided into 5 groups each of which corresponds to one spatial temporal part: 'head-shoulder','left arm','right arm','left leg','right leg'. Each ST-part includes three fine-level parts. In order to compute deformation we define root part for each ST-part: head, left elbow, right elbow, left knee, right knee. Each ST-part is encoded by a feature vector:

$$f(l_i^t) = [\Delta p_1, \Delta p_2, \Delta p_0^t, \Delta p_1^t, \Delta p_2^t] \tag{4.12}$$

where $\Delta p_1 = p_1 - p_0$ and $\Delta p_2 = p_2 - p_0$ are the offsets of parts relative to the root parts in one frame. $\Delta p_0^t = [p_0^{t-1} - p_0^t, p_0^{t+1} - p_0^t]$ is the temporal offset of root part at two subsequent

frames. Δp_1^t and Δp_2^t are defined in the same way as Δp_0^t. Using the temporal offset as a feature is important for getting diverse ST-parts because some of them have the same joint configuration and can only be distinguished by motions. To make the feature invariant to scale, we use head length to estimate the full-body pose scale s_t at each frame, and then normalize the feature vector: $f(l_i^t) = f(l_i^t)/s_t$.

4.4.2 ST-part Clustering

To capture the large appearance and motion variations and let actions share ST-parts, we represent each ST-part as a mixture of components model and the components are obtained by doing k-means clustering on the features $f(l_i^t)$. Each component indicates one type of ST-part. In order to make the ST-part component compact in appearance and motion, we first run k-means on the training examples with same action label and view label to get many small clusters, hence each cluster has small variation in appearance and motion. Clusters that have few examples and belong to only one video are removed as annotation errors. Finally we combine these clusters according to their distance to let them to be shared by different actions and viewpoints. We show some examples of ST-parts in Figure 4.4(c). We can see that the examples of the same ST-part component have similar configurations of fine-level parts.

4.4.3 Learning Model Parameters

The coarse-level template ω_L is learned by linear-SVM using the dense trajectory features[WKS13]. These features don't use any pose information and they try to capture the appearance and short-term motion on the moving blocks which are tracked by optimal flows. The mid-level information is captured by HOG/HOF features of ST-parts. Following[FGM10b], we first train HOG/HOF templates on our ST-part components with linear-SVM, then convolute those templates on image pyramid to extract response maps of ST-part candidates. The feature vector is constructed by performing spatial-temporal max-pooling on response maps, and the template ω_M is learned by another linear-SVM.

The parameters we need to learn for the fine-level action score include $\omega_d^{l_i}$ and $\omega_h^{l_i}$ for the deformation compatibility score and label compatibility score of each ST-part, $\omega_o^{l_i}$ for the ST-part co-occurrence score. We adapt latent Structure-SVM for learning those parameters with regularization. Although all training data have part annotations, ground truth for part locations and ST-part components, only using ground truth may hurt performance because that it is very difficult to annotate all parts precisely at each frame in challenging action datasets. Thus we allow the parts to move between the top N detected parts which are within a certain distance of the ground truth part locations. The Learning process iterates between the following two steps until convergence:

i) To train parameters $w = [\omega_d^{l_1}\, \omega_h^{l_1}\, \omega_o^{l_1} ... \omega_d^{l_M}\, \omega_h^{l_M}\, \omega_o^{l_M}]$, we ignore the detection scores $\lambda \sum_{j=0}^{N_i} S(o_j)$ and $\lambda \sum_{j=1}^{N_i} S(o_j, o_0)$ and the smoothness score $\beta d(l_i^t, l_i^{t+1})$ and train the parameters with detected poses h_i. For the first iteration, h_0 is set to ground truth poses. This is formulated as a supervised multi-class classification problem,

$$\begin{aligned} &\min_{\omega_t} \tfrac{1}{2}\|\omega_t\|_2 + \tfrac{C}{n}\textstyle\sum_{i=1}^{n} \xi_i, \\ s.t. \quad &\max_{\hat{y}\in\mathcal{Y}} \omega_t^T(\phi(x_i, y_i^t) - \phi(x_i, \hat{y}_i^t)) \geq \Delta(y_i, \hat{y}_i) - \xi_i, \end{aligned} \tag{4.13}$$

Here $y_i^t = (a_i, h_i^t)$ where a_i is action label. $\Delta(y_i, \hat{y}_i)$ is 1 if $a_i = \hat{a}_i$and 0 otherwise. t indexes the iteration.

ii) After computing parameters at iteration t, we add the detection score and the smoothness score back into the fine-level score function and infer the poses for each training example. λ and β are determined by cross-validation on training data. Similar to the inference process, we first generate the top N ST-parts candidates within a certain distance around the ground truth and find the best sequence for each ST-part among those candidates under the ground truth action label by Equation 4.11. Then we get the hidden variables h_i^{t+1} from the inferred ST-parts and go back to step 1.

After learning the parameters for the three levels, we obtain the scores from the classifiers of the three levels separately. Finally we learn the weights $\pi_L(A)$, $\pi_M(A)$ and $\pi_H(A)$ to combine the three classifiers for the final action recognition. We formulate the combination

of three different classifiers in the mixture of experts framework [JJ93] where each expert corresponds to a classifier in each level. The weights are computed based on each action example, so the values of weights indicate which expert the example prefers to use for classification. Here we concatenate scores of different categories at each level as features to learn the weights.

4.5 Experiments

We test our method on two public action datasets: the Penn Action dataset [ZZD13] and the sub-JHMDB dataset [JGZ13]. Both datasets are proposed for the purpose of action recognition but they also provide annotations of human joints which are required by our training approach. The performance of both action recognition and pose estimation are evaluated on each dataset.

4.5.1 Evaluation on Penn Action Dataset

The Penn Action Dataset contains 15 action categories and the annotations include action labels, rough view labels and 13 human joints for each image. The occlusion label of each joint is also provided. We follow [ZZD13] to split the data into 50/50 for training/testing. The action 'playing guitar' and several other videos are removed because less than one third of a person is visible in those data. We find that there exist some un-annotated joints that always remain at the left-top corner of image. To correct those errors we train a regression model to predict positions of un-annotated joints by using the visible neighbor joints from videos with the same action and view label. In order to get diverse poses to train [YR12] we first cluster the training data based on whole pose features to get 500 clusters. Then we uniformly select total 5000 images from those clusters as training images. We use the code provided by [YR12], and we set part mixture number to 8: 6 for visible joints and 2 for occluded joints.

The number of mixture components of 5 S-T parts are 43, 37, 31, 56, 58. We find that more components does not improve performance but increase training burden greatly.

The hyper parameters $\lambda = 10$ and $\beta = 0.01$ for detection score and smooth score are determined by cross-validation on the training data. Training converges in only 3 iterations. The coarse-level action templates are trained by the code from [WKS13] and the mid-level action templates are trained by public library of svm. The number of candidates of the ST-part 'head-shoulder' is around 200 and of other ST-parts is around 1000 because the parts 'head' and 'shoulder' only have high scores on a few locations whereas other parts have much larger variations on the response map.

Method	Accuracy
STIP[ZZD13]	82.9%
Dense[WKS13]	73.4%
MST[WNX14]	74.0%
Action Bank[ZZD13]	83.9%
Actemes[ZZD13]	79.4%
Ours(fine)	73.4%
Ours(all)	**85.5**%

Table 4.1: Recognition accuracy on Penn Action dataset. Action Bank is not directly comparable since it uses other training dataset.

Table.4.1 compares the action recognition accuracy between previous methods and ours. We use the numbers of STIP, Dense, Action Bank and Actemes from [ZZD13]. Ours(fine) is trained by only fine-level features and Ours(all) is trained with all feature levels. With only fine-level features, the performance is not very good, but when coarse/mid-level features are added in the performance is improved due to the low resolution and heavy occlusion that make part detection unreliable and not good enough to classify actions.

The confusion matrix of Ours(all) is shown in Figure 4.5. Our approach performs well on the actions such as 'bowl', 'pull up', 'push up' and 'squat', however we achieve low accuracy on actions with fast movement such as 'tennis forehand' because the motion blur makes the positions of critical parts like wrists always wrong.

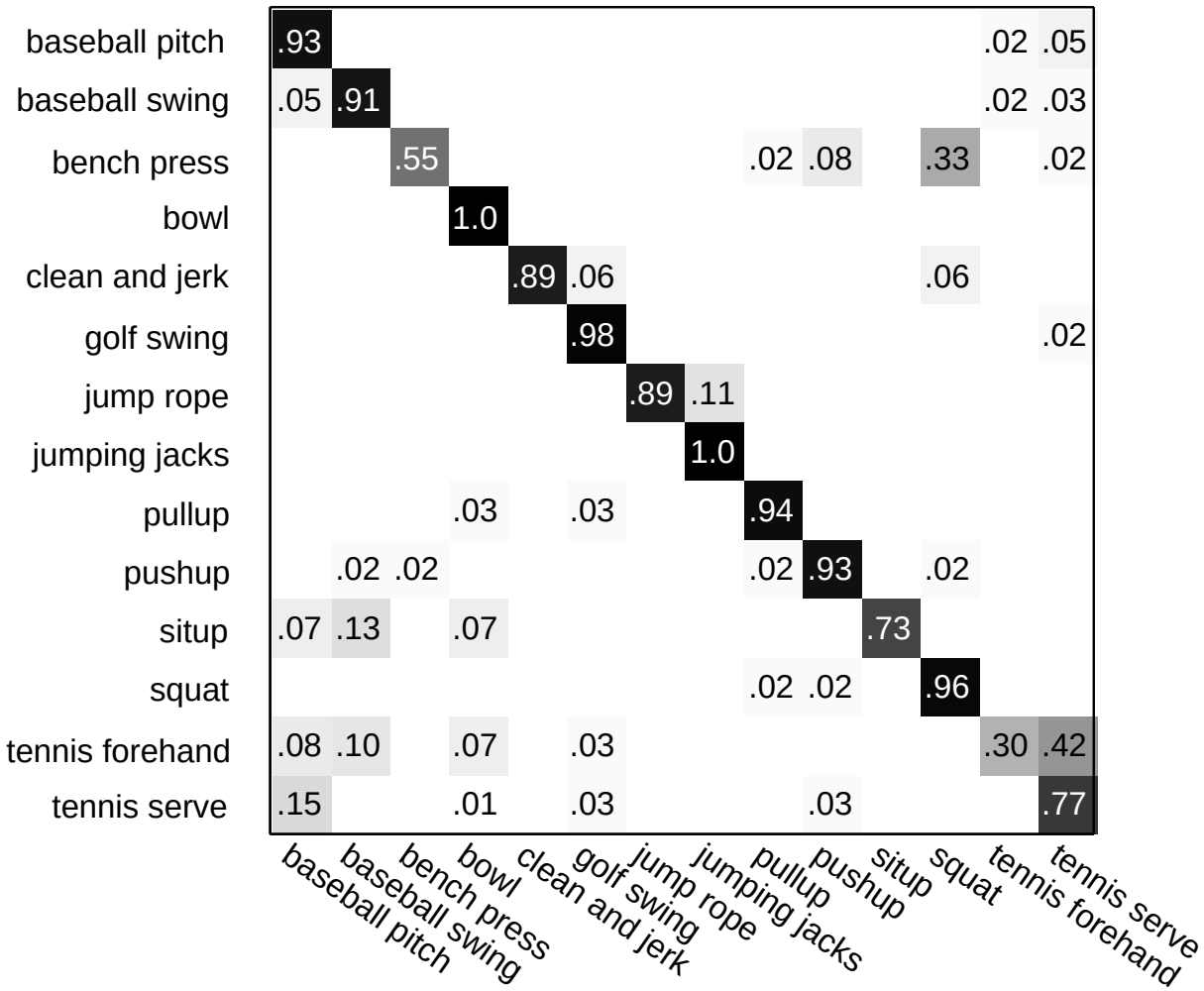

Figure 4.5: The confusion matrix of our method on Penn Action Dataset.

	Penn Action Dataset							
	Head	Shou	Elbo	Wris	Hip	Knee	Ankle	mean
[YR12]	57.9	51.3	30.1	21.4	52.6	49.7	46.2	44.2
[PR11]	62.8	52.0	32.3	23.3	53.3	50.2	43.0	45.3
[CMA14]	–	–	–	–	–	–	–	–
Ours	**64.2**	**55.4**	**33.8**	**24.4**	**56.4**	**54.1**	**48.0**	**48.0**

Table 4.2: Pose estimation accuracy in %.

	sub-JHMDB Dataset							
	Head	Shou	Elbo	Wris	Hip	Knee	Ankle	mean
[YR12]	73.8	57.5	30.7	**22.1**	69.9	58.2	48.9	51.6
[PR11]	79.0	60.3	28.7	16.0	74.8	59.2	49.3	52.5
[CMA14]	47.4	18.2	0.08	0.07	–	–	–	16.4
Ours	**80.3**	**63.5**	**32.5**	21.6	**76.3**	**62.7**	**53.1**	**55.7**

Table 4.3: Pose estimation accuracy in %.

We compare pose estimation accuracy with Yang et al. [YR12] and Park et al. [PR11]. We use their evaluation criteria and set the threshold to 0.2. The results are illustrated in Table 4.2 and Table 4.3. Our method outperforms theirs at every part. It is reasonable that the action specific motion information can help our method to select better parts which are not always detected with highest score the single image based pose estimation.

4.5.2 Evaluation on sub JHMDB Dataset

The sub-JHMDB dataset contains 316 clips with 12 action categories. It provides action labels, rough-view labels and 15 human joints for each frame. All joints are inside the image and there are no un-annotated joints. We use 13 human joints to train the single image pose estimation. We also do clustering on all frames using the whole pose features and select a total 1500 images from clusters for training. The part mixture number is set to 6.

We use the 3-fold cross validation setting provided by the dataset to do experiments.

The number of mixture components of 5 ST parts are 36, 42, 39, 64 and 64. The parameters $\lambda = 20$ and $\beta = 0.01$ for detection score and smooth score are decided by cross-validation and the training converges in 3 iterations.

Method	Accuracy
Dense[JGZ13]	46.0%
MST[WNX14]	45.3%
Pose[JGZ13]	52.9%
Ours(fine)	55.7%
Ours(all)	**61.2**%

Table 4.4: Pose estimation accuracy in %.

Table 4.4 compares our action recognition performance with others. We use the numbers of 'Dense' and 'Pose' from [JGZ13]. For Pose[JGZ13], we use the highest number they obtained by using pose features extracted from pose estimation. With only fine-level features our method already outperforms others. With coarse/mid features the accuracy is increased by nearly 6 percent because there are many low-resolution videos with large errors of pose estimation.

The comparison of pose estimation is illustrated in Table 4.2 and Table 4.3. Our method outperforms [YR12] a lot for parts 'Head' and 'Hip' by nearly 7%, however for the parts 'Elbows' and 'Wrists' our performance is similar to theirs which we believe is caused by that those parts are very subtle and the specific action motion information may prefer the distinguished part locations which are never in the right positions. To compare with [CMA14], we re-train their method on our dataset, and they only estimate the joints in upper body. Results demonstrate that the pairwise smoothness features used in their method are not working well in the action dataset because of the large motion and appearance variations.

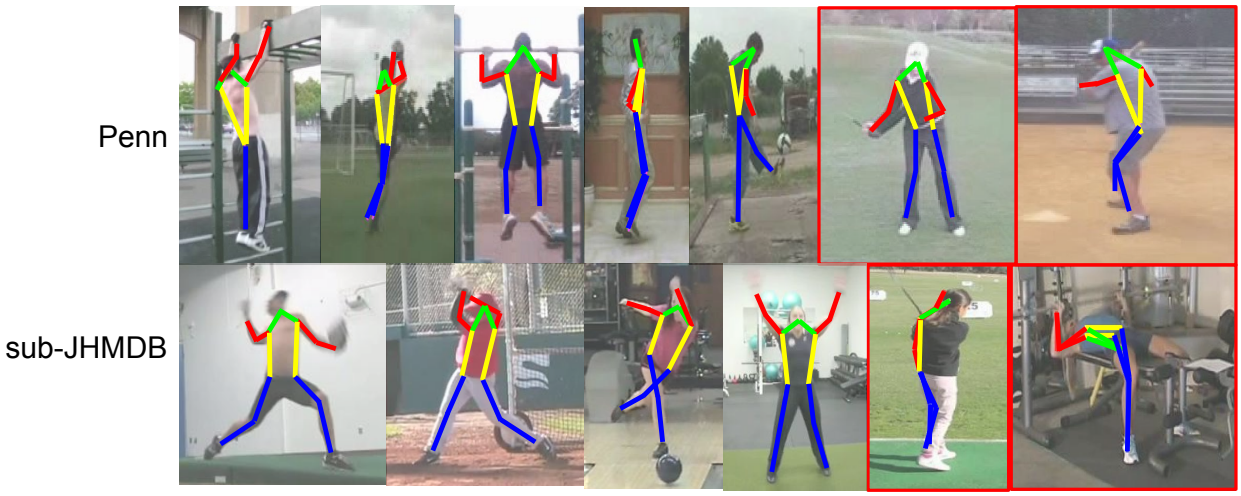

Figure 4.6: Some pose estimation results of our method on the two datasets. The last two columns show failure examples with red rectangle.

4.6 Incorporating Deep Neural Networks (DNN)

In this section, we integrate deep neural networks with our spatial-temporal and-or graph model to further improve the performance of pose estimation and action recognition. Most recently the method based on deep architectures have achieved huge progress in 2D human pose estimation [TJL14, CSW17, CY14, TS14a]. In those works, the appearance and deformation variation of human pose are implicitly modeled in feed-forward computations in networks with deep structures. The learned filters at different layers can also address the self-occlusion problem quite well. Another benefit of those methods is that they can train the models by back-prorogation and the inference can be done in the end-to-end manner. The GPU can be also utilized in those models for largely accelerating the computation speed. Recently there are many methods trying to recognize actions by deep networks and they achieve comparable or better performance than previous methods which use hand-crafted features and carefully designed models. They either train the action recognition with DNN in the end-to-end fashion [SZ14a, KTS14, DHG15] or combine deep features with other non-deep models [WQT15].

We borrow the strength from deep neural networks in the following way: We use the

output from DNN to represent the terminal nodes for fine-level parts of our ST-AOG model. Previously we use the method from [YR12] to compute the response map of each fine-level part, however, here we replace it with a deep network to directly output the response maps so the fine-level part score $S(o_j)$ in Equation 4.4 is from DNN. We generate our fine-level part and ST-part candidates from those response maps following the same way as we mentioned in Chapter 4.3.

We visualize the structure of the deep neural network used for our part localization in Figure 4.7. The feed-forward computation in this network can be divided into two processes based on the feature map size: 1) the top-down process which takes the whole image as input and use many convolutional layers to generate feature map with size 14x14x512. 2) In order to get high resolution part locations, the bottom up process generates enlarge the feature map to 56x56x384 by two deconvolutional layers. The last convolutional layer classifies the part types at each location so the response maps for all parts are computed. To train the network, we generate the ground-truth response maps by adding 2D gaussian centered on the joint location with standard deviation of 2 pixels. The mean squared error (MSE) loss is applied to compare the network output and ground-truth response maps. The training takes one day on a 12GB NIVIDA TitanX GPU.

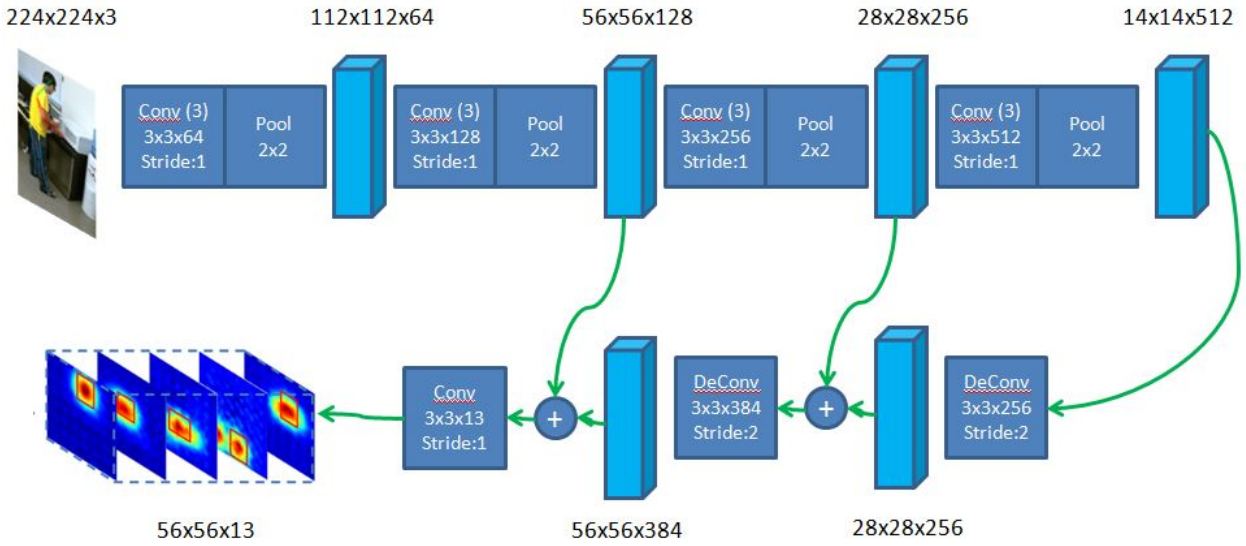

Figure 4.7: The structure of the deep neural network used for our part localization.

We evaluate our method with deep network on Penn Action dataset for both pose estimation and action recognition. We use the same training/testing split and evaluation metric from Section 4.5.1. The results are reported in Table 4.5 where 'ours(deep)' denotes our method with deep network. The action classification accuracy is further improved by absolute 5.8% which we believe is due to the much better part candidates extracted from the deep network instead of [YR12]. The results of pose estimation are reported in Table 4.6. Besides comparing with our previous method without DNN (denoted as 'ours'), we also compare with [GTJ16] which also utilizes deep networks. We can see that the deep network clearly improves part localization accuracy a lot over our previous method. We also outperform [GTJ16] which combines deep convolutional network with recurrent network for video pose estimation. The results also demonstrate the benefit of joint modeling of action and pose in our spatial-temporal and-or graph. Some qualitative results of part localization with deep network are shown in Figure 4.8.

Method	Accuracy
STIP[ZZD13]	82.9%
Dense[WKS13]	73.4%
MST[WNX14]	74.0%
Action Bank[ZZD13]	83.9%
Actemes[ZZD13]	79.4%
Ours(all)	85.5%
Ours(deep)	**91.3**%

Table 4.5: Recognition accuracy on Penn Action dataset. Action Bank is not directly comparable since it uses other training dataset.

4.7 Summary

In this section, we study a new framework to combine action recognition and pose estimation. The Spatial-Temporal And-Or Graph is proposed to represent action and pose jointly and the

	Penn Action Dataset							
	Head	Shou	Elbo	Wris	Hip	Knee	Ankle	mean
[GTJ16]	95.6	93.8	90.4	**90.7**	91.8	90.8	91.5	91.8
Ours	64.2	55.4	33.8	24.4	56.4	54.1	48.0	48.0
Ours(deep)	**97.4**	**96.0**	**91.0**	90.5	**94.5**	**92.8**	**93.2**	**93.3**

Table 4.6: Pose estimation accuracy in %.

Figure 4.8: Some examples of our part localization with deep network. The dots with different colors represent different body parts.

training and inference for the two tasks are conducted simultaneously. Our framework can also embrace the powerfulness of deep neural network (DNN) by replacing the terminal nodes with output from DNNs. The experiment results demonstrate our superior performance over other stat-of-the-art methods and also the benefits of joint modeling of action and pose. One limitation of our method is that we do not handle the self-occlusion explicitly which always appears in action dataset and also a big challenge for pose estimation. In the future, we plan to integrate the 3D pose estimation with current framework because that the occlusion issue can be solved to some extent with the help of 3D information.

CHAPTER 5

Monocular 3D Human Pose Estimation by Predicting Depth on Joints

5.1 Introduction

5.1.1 Motivation and Objective

In this Chapter, we aim at reconstructing full-body 3D human poses from a single RGB image. Specifically we want to localize the human joints in 3D space. Estimating 3D human pose is a classic task in computer vision and serves as a key component in many human related practical applications such as intelligent surveillance system, human-robot interaction system, human activity analysis, human attention recognition and so on. There are some existing works which estimate 3D poses in constrained environment from depth images [YWY11, SFC11] or RGB images captured simultaneously at multiple viewpoints [YGG12, HG12]. Different from them, here we focus on recognizing 3D pose directly from a single RGB image which is easier to be captured from general environment.

Estimating 3D human poses from a single RGB image is a challenging problem due to two main reasons: 1) the target person in the image always exhibits large appearance and geometric variation because of different clothes, postures, illuminations, camera viewpoints and so on. The highly articulated human pose also brings about heavy self-occlusions. 2) even the ground-truth 2D pose is given, recovering the 3D pose is inherently ambiguous since that there are infinite 3D poses which can be projected onto the same 2D pose when the depth information is unknown.

One inspiration of our work is the huge progress of 2D human pose estimation made by

recent works based on deep architectures [TJL14, NYD16, WRK16, CSW17]. In those works, the appearance and geometric variation are implicitly modeled in feed-forward computations in networks with hierarchical deep structure. The self-occlusion is also addressed well by filters from different layers capturing features at different scales. Another inspiration of our work is the effectiveness that deep CNN has demonstrated for depth map prediction from monocular image [EPF14, WSL15, LSL15] instead of stereo images or videos. Most of those approaches directly predict the pixel-wise depth map using deep networks and some of them build markov random fields on the output of deep networks. The largest benefit is that they are not bothered by designing geometric priors or hand-crafted features, and most models can be trained end-to-end using back-propagation. Based on the two above inspirations, in this paper, we propose a novel framework to address the challenge of lifting 2D pose to 3D pose by predicting the depth of joints from two cues: global 2D joint locations and local body part images. The 2D joint locations are predicted from off-the-shelf pose estimation methods.

5.1.2 Method Overview

Our approach is built on a two-level hierarchy of LSTM networks to predict the depth on human joints and then recover 3D full-body human pose. The first level of our model contains two key components: 1) the skeleton-LSTM network which takes the predicted 2D joint locations to estimate depth of joints. This is based on the assumption that the global human depth information such as global scale and rough depth can be inferred from the correlation between 2D skeleton and 3D pose. This global skeleton feature can help to remove the physically implausible 3D joint configuration and predict depth with considerable accuracy; 2) the patch-LSTM network which takes the local image patches of body parts as input to predict depth. This network addresses the correlation between human part appearance and depth information. To better model the kinematic relation of human skeletons, the two recurrent networks have tree-structures which are defined on human skeletons. During training, the features at different joints are broadcasted through the whole skeleton and in testing the depth are predicted for each joint in top-down fashion. The skeleton-LSTM is

first pre-trained on 2D-3D pose pairs without any image so infinite training examples can be generated by projecting 3D poses onto 2D space under arbitrary viewpoint. The patch-LSTM is pre-trained on human body patches extracted around 2D joints. To increase appearance variation and reduce overfitting we employ multi-task learning on the combination of two data sources: the MoCap data with the task of depth regression and in-the-wild pose data with the task of 2D pose regression. The two networks are aggregated in the second layer and finetuned together for final depth prediction. We evaluate our method extensively on Human3.6M dataset [IPO14] using two protocols. To test the generalization ability, we test our method on HHOI dataset using the model trained on Human3.6M dataset. The results demonstrate that we achieve better performance over state of the art quantitatively and qualitatively.

The contribution of our approach is three-fold:

i) We explore the ability of deep network for predicting the depth of human joints and then recover 3D pose. Our framework is more flexible than others because complex optimization is not needed and model can be trained end-to-end.

ii) We incorporate both global 2D skeleton features and local image patch features in a two-level LSTM network and the tree-structure topology of our model naturally represents the kinematic relation of human skeleton. The features at different joints are aggregated in top-down fashion.

iii) The extensive experiments demonstrate the superior quantitative and qualitative performance of our work relative to other state of the art methods.

5.2 Models

In this section, we will first describe the relationship between 3D pose estimation and depth prediction, and then introduce our model and its corresponding formulations.

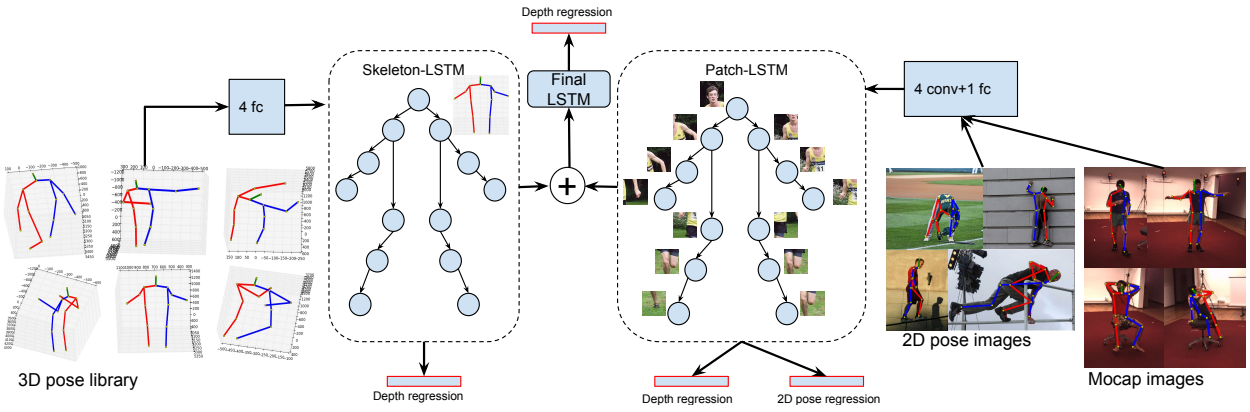

Figure 5.1: Overview of our model structure and training process. In the first level, the skeleton-LSTM is pre-trained with 2D-3D pose pairs to predict depth from global skeleton features. The patch-LSTM predicts depth from local image patch evidence of body parts. The tree-structure of two networks are defined on the kinematic relation of human joints, so the state of current joint is composed of the hidden states of its parents and the input feature of itself. The two networks are integrated into another LSTM at the second level for end-to-end training. To reduce overfitting of patch-LSTM, we borrow in-the-wild 2D pose images and train the network with multi-task loss: the depth prediction loss and 2D pose regression loss.

5.2.1 Recover 3D Pose by Depth Prediction

The 3D human pose is represented by a set of locations of human joints in 3D space. We use $W \in \mathbb{R}^{3\times N}$ to denote the 3D pose in the world's coordinate system where N is the number of joints. Each 3D joint location in W is denoted by the 3D coordinate $w_i = [X_i, Y_i, Z_i]$, $i \in 1, ..., N$. The 2D pose is defined in the same way as $S \in \mathbb{R}^{2\times N}$ and each 2D joint is denoted as $s_i = [x_i, y_i]$. The relationship between each 3D joint and 2D joint can be described as a perspective projection:

$$z_i \cdot \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = f \cdot [R|T] \cdot \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix}, i \in 1, ..., N \tag{5.1}$$

where $R \in \mathbb{R}^{3\times 3}$ denotes the camera rotation matrix, f denotes focal length and z_i denotes depth of joint i. Note that in Eqn 5.1 there is no weak perspective assumption about the relationship between 3D pose and 2D pose. Given 2D joint locations $[x_i, y_i]$ and focal length f we need the depth value for each joint z_i, global rotation R and translation T to recover all 3D joint locations. Since there are infinite combinations of transformation matrix $[R|T]$ and world coordinate $[X_i, Y_i, Z_i]$ which can produce the same $[x_i, y_i]$ and z_i with unknown camera position, therefore in this work we focus on predicting $z = [z_1, ..., z_N]$ to recover the 3D pose in the camera's coordinate system $\hat{W} = [R|T] \cdot [W|\mathbf{1}]^T$.

In order to predict the depth, we define the joint distribution of depth z, 2D pose S and image I:

$$P(z, S, I) = P(z|S, I) \cdot P(S|I) \cdot P(I) \tag{5.2}$$

where $P(S|I)$ represents the 2D pose estimation from single image I which can be handled by any off-the-shelf 2D pose estimation method. The separate estimation of depth and 2D pose allows $P(S|I)$ to be modeled by any complex method. Any improvement made in $P(S|I)$ can be immediately plugged into $P(z, S, I)$ and re-training of the whole system is not needed. $P(z|S, I)$ is modeled as a two-level hierarchy of LSTM which utilizes the 2D

pose S and image evidence I in the first level, and integrates two networks in the second level for final depth prediction. The details of our model are described below.

5.2.2 Components of our Model

To take advantage of the global skeleton feature and local image feature to predict depth, we use a deep structure of LSTMs with two levels. As shown in Figure 5.1, the first level consists of two recurrent networks: a skeleton-LSTM stacked with a 2D pose encoding network which takes the predicted 2D pose S as input and a patch-LSTM stacked with image patch encoding network which takes the local image patches $I(s_i), i \in [1, ..., N]$ around 2D joints as input. The hidden states of the two networks at each joint are max pooled and forwarded to the LSTM at the second level to predict the final real valued depth d_i for each joint.

Inspired by those graphical model based pose estimation methods [YR12, RPZ13, PAG13], we represent human pose as a tree structure based on the kinematic relation of skeleton. The articulated relation are better represented and the correlation of features at parent joint and child joint are better captured within tree structure than the flat or sequential structure. Similar to the framework of [**?**], we adapt the tree-structured LSTM for modeling human pose and integrating global and local features. The aggregated contextual information are propagated efficiently through the edges between joints. In experiments we evaluate different choices of model structure and demonstrate the empirical strength of tree-structure over the flat or sequential model.

The three tree-structured LSTMs in our model share the same equation and only differ in the input data. At joint j, the state of the LSTM unit is composed of the current input feature x_j, all hidden states h_k and memory states c_k from its parents, and the output is the hidden state h_j and memory state c_j which is forwarded to the child joint:

$$(h_j, c_j) = LSTM(x_j, \{h_k\}_{k \in N(j)}, \{c_k\}_{k \in N(j)}, W, U) \tag{5.3}$$

where W and U are weight matrices. To obtain a fixed dimension of the input feature, the hidden states from all parents of joint j are mean pooled:

$$\bar{h}_j = \Big(\sum\nolimits_{k \in N(j)} h_k\Big) \Big/ |N(j)| \tag{5.4}$$

$\bar{h}_j$ is used to compute LSTM gates of joint j as below:

$$\begin{aligned}
i_j &= \sigma(W^i x_j + U^i \bar{h}_j + b^i) \\
f_{jk} &= \sigma(W^f x_j + U^f h_k + b^f) \\
o_j &= \sigma(W^o x_j + U^o \bar{h}_j + b^o) \\
\tilde{c}_j &= tanh(W^c x_j + U^c \bar{h}_j + b^c) \\
c_j &= i_j \odot \tilde{c}_j + \sum\nolimits_{k \in N(j)} (f_{jk} \odot c_k) \\
h_j &= o_j \odot tanh(c_j)
\end{aligned} \tag{5.5}$$

where i_j is the input gate, f_{jk} is the forget gate of parent k, o_j is the output gate, σ denotes the sigmoid function and $\odot$ denotes the element-wise multiplication. Note that our LSTM has different forget gates for different parent joint and the multiplication of each f_{jk} and c_k indicates the influence of parent k on current joint j.

2D pose encoding. As shown on the left of Figure 5.1, the skeleton-LSTM utilize the global information from 2D skeleton S to predict the depth. In order to have a better representation of the 2D skeleton, we apply a multi-layer perceptron network shared by all joints to extract the global pose feature. The input feature of the skeleton-LSTM at joint j is $x_j^s = MP(\hat{S}_j)$ where $\hat{S}_j$ is the normalized 2D pose by subtracting each joint location by the current joint location $[x_j, y_j]$. The structure of the multi-layer perceptron is visualized in Figure 5.2. It is trained together with the skeleton-LSTM.

image patch encoding. As shown on the right of Figure 5.1, the patch-LSTM utilizes the local information from image patches of body parts for depth prediction. The input of LSTM unit at joint j is the encoded feature of the corresponding image patch around that joint. We use $x_j^p = CNN(I(x_j, y_j))$ to denote the input feature which is the last layer of a small ConvNet shared by all joints. The structure of the ConvNet is visualized in Figure 5.3.

For the final LSTM at the second layer, the input feature at joint j is the element-wise

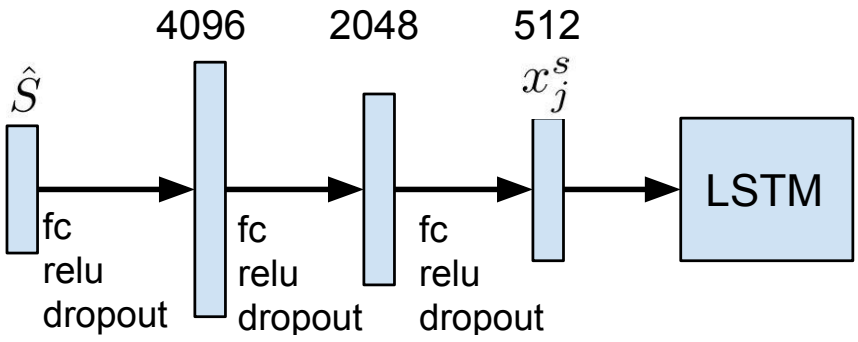

Figure 5.2: The multi-layer perceptron network for 2D pose encoding.

max pooling of hidden states from skeleton-LSTM and patch-LSTM: $x_j = max(h_j^s, h_j^p)$. The real-value depth in log space $log(z_j)$ at each joint is predicted by attaching another fully-connected layer on the hidden state h_j: $log(z_j) = \sigma(W^z h_j + b^z)$.

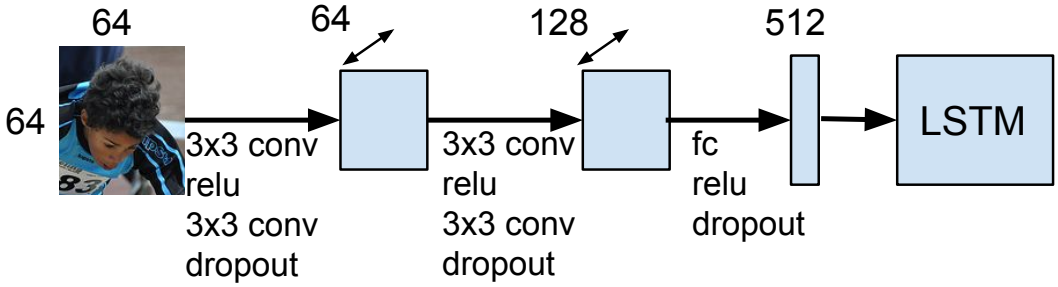

Figure 5.3: The convolutional network for image patch encoding.

5.3 Learning

The model weights that we need to learn include the weights of three LSTMs, and the weights of the 2D pose encoding network and image patch encoding network. The learning process consists of three phrases:

1) The skeleton-LSTM and skeleton encoding network are first pre-trained from Mocap data using the 2D-3D pose pairs with depth prediction loss. The RGB images are not needed and infinite 2D-3D pose pairs can be generated by projecting each 3D pose into 2D poses under different camera viewpoints.

2) The patch-LSTM and image encoding network are first pre-trained on RGB images from both MoCap dataset and in-the-wild 2D pose dataset with multi-task loss. Although the 2D pose dataset does not have depth data, they act as a regulariser with loss function of 2D pose regression.

3) The last step is to combine the two LSTMs in the second layer for end-to-end training.

5.3.1 Loss Function

The loss functions for depth regression at the above three phrases share the same formulation but use different input feature and hyper parameters. Inspired by [EPF14], we define the loss based on both relative error and absolute error:

$$\begin{aligned} d_i &= log(z_i) - log(z_i^*) \\ d_{ij} &= (log(z_i) - log(z_j)) - (log(z_i^*) - log(z_j^*)) \\ L(z, z^*) &= \lambda \frac{1}{n} \sum_{i=1}^{n} d_i^2 + \beta \frac{1}{|E|} \sum_{(i,j) \in E} d_{ij}^2 \end{aligned} \tag{5.6}$$

where z is the vector of all depth values on joints, n is the number of joints and E denotes the set of edges in the tree structure. The first term of $L(z, z^*)$ is the mean squared error which enforces the absolute depth at each joint to be correct and the second term penalizes the difference of relative depth between each parent-child pairs. Instead of considering all pairwise depth relations in [EPF14], we focus on the parent-child depth relations represented by edges in the tree structure of our model. The hyper parameters λ and β control the balance between absolute depth loss and relative depth loss. We set different λ and β for training skeleton-LSTM and patch-LSTM since they are good at minimizing different losses with different features.

5.3.2 Multi-task learning for patch-LSTM

As mentioned in Section 3.1, the patch-LSTM needs to be trained on image data with depth values of joints, and the images from Mocap data are captured from a highly constrained environment with small appearance variation which may lead to severe over-fitting. To

decrease over-fitting, we argument training data using in-the-wild images with annotations of 2D poses from public pose datasets. Although the 2D pose datasets do not have depth, we apply the multi-task learning [**?**] to combine it with Mocap dataset in the same network. Specifically, we add another fully-connect layer on top of the hidden state of LSTM to regress the 2D joint locations which are normalized following [TS14b]. The overall training loss is the sum of depth prediction loss which only operates on Mocap data and 2D pose regression loss which operates on both Mocap data and 2D pose data.

5.4 Experiment Results

Datasets. For empirical evaluation of our 3D pose estimation we use two datasets: Human3.6M dataset (H3.6M) [IPO14] and UCLA Human-Human-Object Interaction Dataset (HHOI) [SRZ16]. The Human3.6M dataset is a large-scale dataset which includes accurate 3.6 million 3D human poses captured by Mocap system. It also includes synchronized videos and projected 2D poses from 4 cameras so the 2D-3D pose pairs are available. There are total 11 actors performing 15 actions such as Sitting, Waiting and Walking. This dataset is captured in a controlled indoor environment. The HHOI dataset contains human interactions captured by MS Kinect v2 sensor. It includes 3 types of human-human interactions: shake hands, high five and pull up and 2 types of human-object-human interactions: throw and catch, hand over a cup. There are 8 actors performing 23.6 instances per interaction on average. The data is collected in a common office with clutter background. For in-the-wild 2D pose dataset, we use the MPII-LSP-extended dataset [PIT16] which is a combination of the extend LSP and the MPII dataset. After flipping each image horizontally, we get a total of 80000 images with 2D pose annotations.

Implementation details. We use the public deep learning library Keras to implement our method. The training and testing are conducted on a single NVIDIA Titan X (Pascal) GPU. To train the skeleton-LSTM, we use the 2D-3D pose pairs down-sampled at 5 fps from Human3.6M dataset. Each 3D pose is projected onto 2D poses under 4 camera viewpoints. The 2D pose encoding network is stacked with skeleton LSTM for joint training with param-

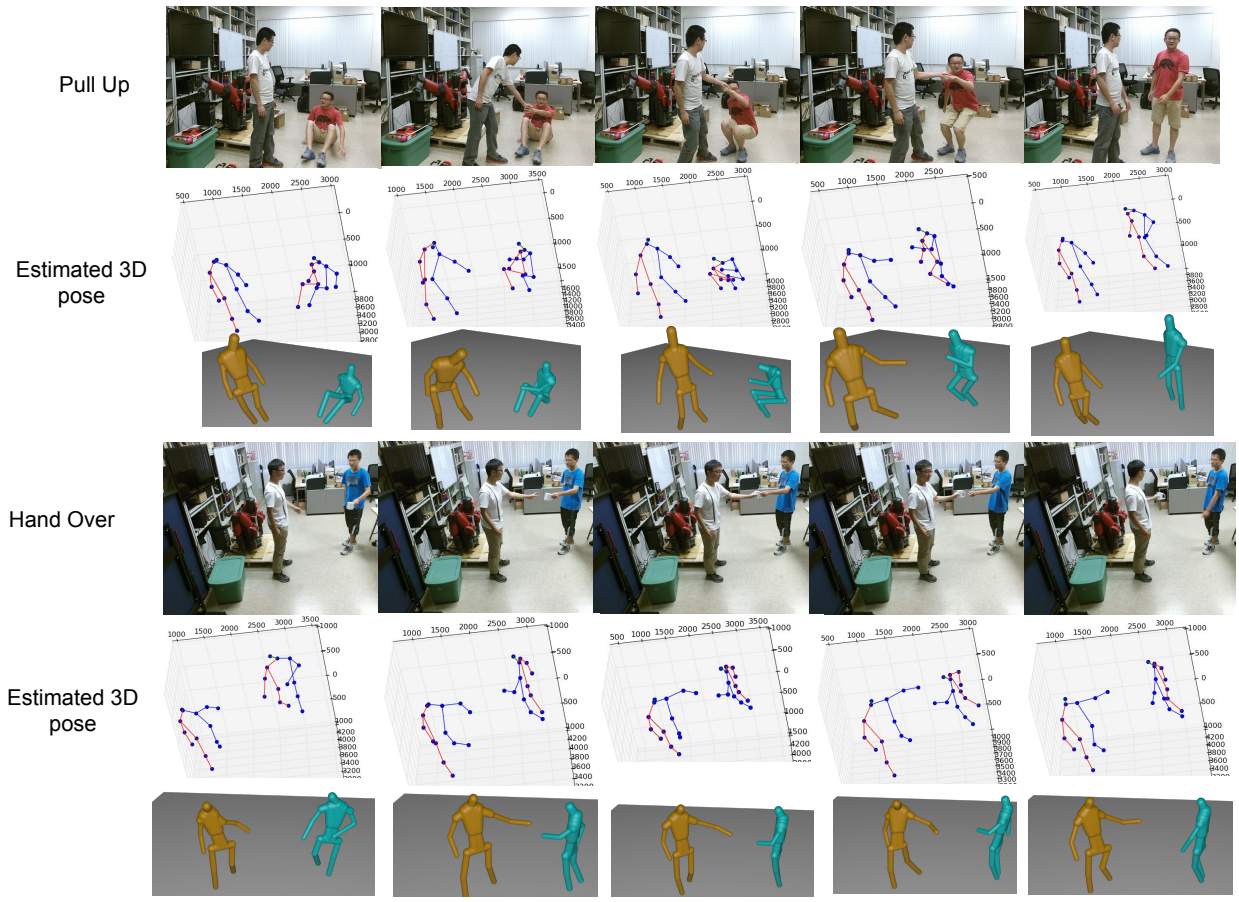

Figure 5.4: Qualitative results from HHOI dataset. We visualize ten frames and their estimated 3D poses from action 'Pull Up' and 'Hand Over'. Besides the original results, we show pose rendering results for better visualization.

eter $\lambda = 0.5$ and $\beta = 0.5$. To train the patch-LSTM, we use image frames down-sampled at 25 fps from Human3.6M and all images from MPII-LSP-extended dataset. The image patch encoding network is stacked with patch-LSTM for joint training with parameter $\lambda = 0.2$ and $\beta = 0.8$. After the separate training of the two networks, we finally combine them with the final LSTM for end to end training using $\lambda = \beta = 0.5$. RMSprop [TH] is used for mini-batch gradient descent and the learning rate is 0.00001 for all networks. The batch size is 128 for skeleton-LSTM and 64 for others.

Baseline. In addition to comparing our final system with state of the art methods, we also use two variations of our method as baselines : 1) To isolate the impact of image feature, we only keep the skeleton-LSTM and the 2D pose encoding network and train them jointly to predict the depth and then recover 3D pose. This baseline is denoted as "ours(s)"; 2) We only keep patch-LSTM and image patch encoding network and it is denoted as "ours(p)".

5.4.1 Evaluation on Human3.6M Dataset

We compare our results with state of the art approaches in 3D pose estimation on Human3.6M dataset. We follow the evaluation protocol in [YIK16]. The image frames and poses from subject S1, S5, S6, S7, S8 and S9 are used for training and S11 for testing. The testing data from S11 is down-sampled at 64fps and some poses without synchronized images are removed so the total testing set has 3612 poses. The training set has around 1.8 million 2D/3D poses with synchronized images. The 3D pose error is measured by the mean per joint position error (MPJPE) [IPO14] at 13 joints up to a rigid transformation. We refer to this protocol by P1.

The quantitative results are presented in Table 5.1. The method "our(s)" and "our(p)" are two method variations and "our(s+p)" is our final system. In all model variations, we apply the pre-trained off-the-shelf 2D pose estimator from [HL16] to detect 2D poses without any re-training or fine-tuning because it is easy for the model to overfit the Human3.6M dataset which is captured in a highly constrained environment with limited appearance variation. Yasin et al. [YIK16] applies the pictorial structure model to represent 2D pose.

Methods	Direct	Discuss	Eat	Greet	Phone	Pose	Purchase	Sit	SitDown	Smoke	Photo	Wait	Walk	WalkDog	WalkTo	Mean
Yasin[YIK16]	88.4	72.5	108.5	110.2	97.1	81.6	107.2	119.0	170.8	108.2	142.5	86.9	92.1	165.7	102.0	110.2
Gall[KG15]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	115.7
Rogez[RS16]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	88.1
our(s)	70.8	71.0	81.0	83.2	87.6	73.3	80.7	103.4	121.7	95.1	91.2	80.8	**71.8**	89.3	**73.0**	84.9
our(p)	93.5	88.0	116.7	105.4	111.3	80.0	99.7	136.7	173.2	111.5	117.6	86.9	89.1	118.8	97.5	108.4
our(s+p)	**62.8**	**69.2**	**79.6**	**78.8**	**80.8**	**72.5**	**73.9**	**96.1**	**106.9**	**88.0**	**86.9**	**70.7**	71.9	**76.5**	73.2	**79.5**

Table 5.1: Quantitative comparison of mean per joint errors (mm) on Human3.6M dataset (Protocol 1).

Gall et al. [KG15] and Rogez et al. [RS16] predict the 3D poses without explicit 2D pose modeling.

Table 5.1 shows that our model variation "our(s)" outperforms other approaches which demonstrates the powerfulness of predicting depth from only 2D pose. The human 2d pose can be seen as a strong cue to estimate the corresponding 3D pose. Although there are some ambiguities in the perspective projection, with only 2D pose features our model already captures helpful information to predict the depth of joint. This result also indicates that predicting the joint depth is more robust than predicting the whole 3D pose.

Our method variation "our(p)" achieves similar results with [KG15] which also uses image patches to predict 3D joint locations. To train the patch-LSTM, we focus on the pairwise relative losses as shown in Eqn 5.6 because it is hard to predict the absolute depth with resized body part images. After integrating the skeleton-LSTM and patch-LSTM we further decrease the error to 79.5mm which outperforms the second best result by 9.8%.

We also report results for protocol 2 (P2) which is employed in [ZZL16, TRL16, RS16]. The 2D/3D poses and image frames of subject S1, S5, S6, S7 and S8 are used for training and S7, S9 are used for testing. The estimated 3D pose and ground truth pose are aligned with their root locations and MPJPE is calculated without rigid transformation. The results of our final system and state-of-the-art approaches are presented in Table 5.2. Our method clearly outperforms the second best result [ZZL16] by 13.72% even though they use temporal information to help 3D pose estimation.

Methods	Direct	Discuss	Eat	Greet	Phone	Pose	Purchase	Sit	SitDown	Smoke	Photo	Wait	Walk	WalkDog	WalkTo	Mean
Tekin[TRL16]	102.4	158.5	87.9	126.8	118.4	185.0	114.7	107.6	136.2	205.7	118.2	146.7	128.1	**65.9**	**77.2**	125.3
Zhou[ZZL16]	**87.4**	109.3	87.1	103.2	116.2	143.3	106.9	**99.8**	**124.5**	199.2	107.4	118.1	114.2	79.4	97.7	113.0
Rogez[RS16]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	121.2
our(s+p)	90.1	**88.2**	**85.7**	**95.6**	**103.9**	**92.4**	**90.4**	117.9	136.4	**98.5**	**103.0**	**94.4**	**86.0**	90.6	89.5	**97.5**

Table 5.2: Quantitative comparison of mean per joint errors (mm) on Human3.6M dataset (Protocol 2).

5.4.2 Evaluation on HHOI Dataset

To evaluate how our method can be generalized to data from a totally different environment, we train model on Human3.6M dataset and test it on HHOI dataset which is captured with Kinect sensor in a casual environment. We pick 13 joints defined by Kinect and also use mean per joint error as the evaluation metric. Each action instance is down-sampled at 10fps for efficient computation and both persons in each action are evaluated. We still use the focal length from Human3.6M to recover 3D poses and the poses are compared up to a rigid transformation and also scale transformation. The method of [HL16] is used to produce 2D poses. Some qualitative results are presented in Fig 5.4. For better visualization of 3D pose, we do pose rendering using the code released from [ZJZ16]. The two poses at each frame are recovered independently so their relative depth may not be correct. We regress the relative mean depth between two persons using the 2d distance on y axis between two persons' feet.

There is no public code for recent methods compared in Human3.6M dataset so we implement another baseline 'Nearest' which match the predicted 2D pose with 2D poses from Human3.6M and select the depth from the 3D pose paired with the nearest 2D pose as the predicted depth. Note that the Kinect may produce unreasonable 3D poses because of occlusions and the evaluation with those poses cannot reflect true performance of compared methods, thus we looked at each action video and carefully select some of them for quantitative comparison. Specifically we keep all videos from 'PullUp' and 'HandOver', and a few videos from 'HighFive' and 'ShakeHands'. We select the smaller error calculated among the predicted pose and its flipped one due to the left-right confusion of Kinect. The quantitative

results are summarized in Table 5.3. The action 'Pull Up' gets the biggest error among all actions due to the large pose variation. Our final model outperforms other baselines in three actions.

Method	PullUp	HandOver	HighFive	ShakeHands
Nearest	161.2	126.2	117.3	129.6
our(s)	139.8	105.2	98.4	**113.1**
our(p)	132.4	102.5	103.0	129.0
our(s+p)	**124.8**	**101.9**	**96.1**	118.6

Table 5.3: Quantitative comparison of mean per joint errors (mm) on HHOI dataset.

5.4.3 Diagnostic Experiments

To better justify the contribution of each component of our method, we do several diagnostic comparisons in the following. The Human3.6M and protocol 1 is used for all comparisons.

Effect of 2D poses. We first evaluate our method on 3D pose estimation when ground truth 2D poses are given and compare it with [YIK16]. The results are presented in Table 5.4 (a) and indicate the potential of improvement when a more accurate 2D pose estimator is available.

(a)

Method	Error
Yasin et al. [YIK16]	70.3
our(s)	46.3
our(p)	79.3
our(s+p)	**42.9**

(b)

Method	No scale	scale
our(s)	84.9	80.6
our(p)	108.4	105.4
our(s+p)	79.5	74.0

Table 5.4: Quantitative comparison of mean per joint errors (mm) on Human3.6M (a) given ground truth 2D poses; (b) with and without scale transformation.

We also consider the effect of performance of 2D pose estimation. To generate 2D poses

with different errors, we add disturbance to locations of different number of joints. Specifically, for each testing 2D pose, we randomly choose certain number of joints and add a uniform random noise in the range $[0, e], e = 0.1 \cdot max(h, w)$, where h and w are the height and width of the pose respectively. The absolute depth error and 3D pose error are calculated at each number of disturbed joints. The results are visualized in Fig 5.5. Although the absolute depth error increases quickly with the error of 2D pose estimation, the 3D pose error increases slowly which indicates that the relative depth relations are not effected too much by the disturbed 2D pose.

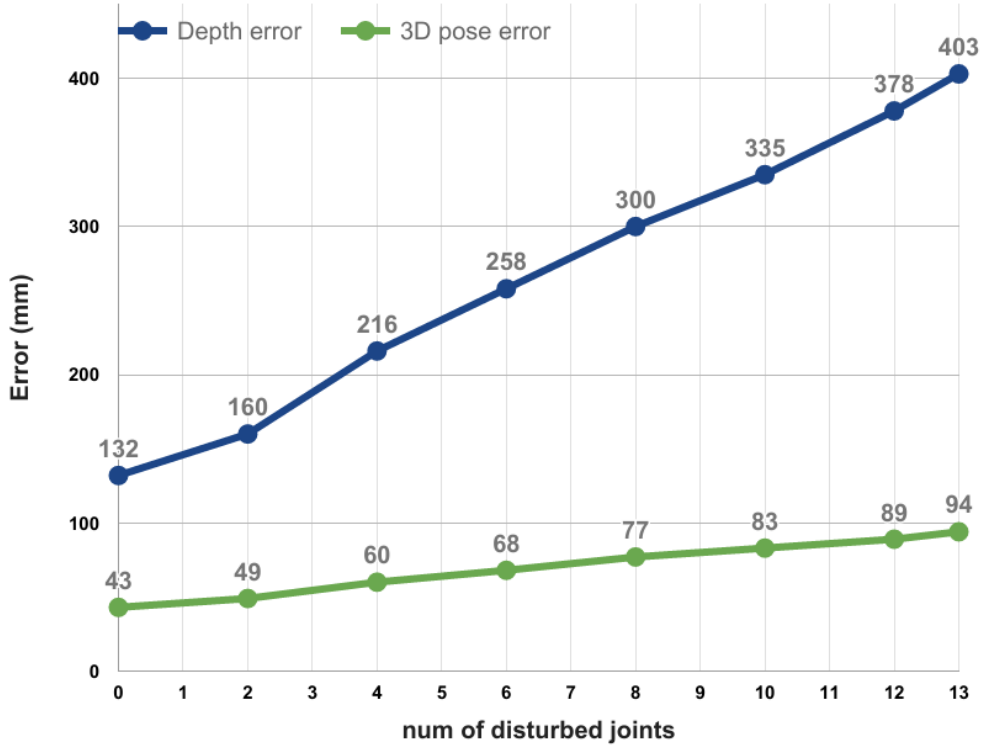

Figure 5.5: Depth and 3D pose error with different number of disturbed joints.

Scale transformation. In general, it is impossible to estimate the global scale of the person from monocular image so we evaluate different variations of our model with a scale transformation. The results are presented in Table 5.4 (b) which show that there is a consistent improvement on all model variations when the scale transformation is allowed.

Different model structures. We consider the effect of model structure on the 3D pose estimation performance when only 2D skeleton features are used. We compare the following structures with the loss function defined in Eqn. 5.6:

-**ske-econding**. We remove the tree-structure LSTM network and only keep the 2D pose encoding network. In this setting, the effect of explicit modeling of relations between joints are removed.

-**ske-seq**. We change the tree structure of the skeleton LSTM to a sequential chain structure with a fixed order of joints. It is impossible to evaluate all permutations of joints so we choose the order which is more similar to the tree structure: head-left limb-right limb.

-**ske-tree**. The skeleton-LSTM used in our final system.

We also evaluate the effect of the model structure when only body part image features are used. We remove 2D pose features and compare the three model variations:

-**whole-vgg**. We apply the VGG model [SZ14b] to predict the depth from the cropped image of the whole person instead of body part.

-**patch-seq**. It has the same sequential structure as ske-seq.

-**patch-tree**. The patch-LSTM used in our final system.

The results are shown in Table 5.5. The method with LSTM network boost performance a lot on both skeleton features (84.9 vs 113.0) and image patch features (108.4 vs 169.8) which demonstrates that the modeling of relationships between joints are essential for predicting depth. The comparison between sequential chain structure and tree structure demonstrates that the latter is more appropriate for modeling human skeleton than the former on both features.

5.5 Summary

In this Chapter, we propose a framework to predict the depth of human joints and recover the 3D pose. Our approach is built on a two level hierarchy of LSTM by utilizing two cues: the 2D skeleton feature which is captured by skeleton-LSTM and image feature of

Method	Error
ske-encoding	113.0
ske-seq	89.0
ske-tree	**84.9**
whole-vgg	169.8
patch-seq	118.6
patch-tree	**108.4**

Table 5.5: Comparison between different model structures.

body part which is captured by patch-LSTM. The whole framework can be trained end to end and it allows any off-the-shelf 2D pose estimator to be plugged in. The experiments demonstrate our better performance qualitatively and quantitatively. In the future work, we plan to extend this work to video-domain and combine it with 3D scene reconstruction by considering temporal constraints and person-object relations.

CHAPTER 6

Conclusion

In computer vision, human pose estimation and action recognition are two essential tasks not only for other research areas but also for building real-world applications. In this dissertation, we propose several novel approaches for the two tasks and the extensive experiment results on public datasets demonstrate the superiority and effectiveness of our methods. Specifically we study several ways to joint the training and inference of the two tasks so they can improve the performance of each other.

Firstly we propose animated pose template (APT) to represent actions and key poses in a two-level And-Or Tree structure. In this hierarchical model, each action consists of a sequence of moving pose templates (MPT) each of which corresponds to the key pose in the action. The MPTs can transit to each other with learned transition probabilities. Each MPT contains two templates: 1) a shape template represented by an And-node capturing appearance from one static frame. 2) a motion template represented by an Or-node capturing motion information within 2-5 frames. The transitions between MPT are modeled by a Hidden Markov Model. During inference, the key poses candidates are extracted first and then the sequences of key poses for each action are computed effectively by dynamic programming.

Secondly, in order to recognize actions with unseen viewpoints, we propose a multi-view Spatial-Temporal And-Or Graph (MST-AOG) model to represent the cross-view action. Within the multiple layers in MST-AOG, the geometry, appearance and motion variations are explicitly modeled. We use the 3D key poses to represent action from arbitrary views and each 3D key pose is projected into different quantized views. The appearance features at unseen views are interpolated from features of known views and the 2D geometry features can be projected from 3D geometry. To train the model, we take advantages of 3D human

skeleton data which are obtained from Kinect cameras. We only use RGB videos for testing and the 3D skeletons are not needed. A new Multi-view action3D dataset is collected by us and released to public.

Thirdly, we study the joint modeling of action and pose in a deeper way so that the fine-level parts of pose can also be localized. We employ a deeper Spatial-Temporal And-Or Graph (ST-AOG) to model the action and pose. The ST-AOG captures information at three scales: 1) the coarse level captures the rough motion and appearance of action. With coarse level feature we don't need to infer pose for classifying actions. 2) the middle level captures the action specific motions by using the spatial-temporal parts (ST-parts). 3) the fine level captures the appearance of fine level parts so the all parts can be localized in this level. We train classifiers for the features at each level and they are integrated by mixture of experts algorithm. We also demonstrate the ability of our model to incorporate features from deep neural networks. The performance is improved a lot with the deep learned features.

The last but not the least, we propose a novel framework based on deep networks for full-body 3D human pose estimation which is quite essential for several high-level tasks such as human attention prediction in 3D scene, robot-based human action prediction and human-robot interaction. Our method is built on a two-level Long Short-Term Memory (LSTM) network to utilize two cues: the global features from 2D human skeleton and local features from image patches of body parts. With the tree-structure of our network defined on human skeleton, the information between different joints are broadcasted in top-down manner. The experiments demonstrate that our method can be generalized to in-the-wild dataset even the model is trained by only 3D dataset captured in a constrained environment.

REFERENCES

[BD01] A. Bobick and J. Davis. "The Recognition of Human Movement Using Temporal Templates." *PAMI*, **23**(3):257–267, 2001.

[BM09] L. Bourdev and J. Malik. "Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations." In *ICCV*, 2009.

[CLH10] L. Cao, Z. Liu, and T. Huang. "Cross-dataset action detection." In *CVPR*, 2010.

[CMA14] A. Cherian, J. Mairal, K. Alahari, and C. Schmid. "Mixing Body-Part Sequences for Human Pose Estimation." In *CVPR*, 2014.

[CSW17] Z. Cao, T. Simon, S. Wei, and Y. Sheikh. "Realtime Multi-person 2D Pose Estimation using Part Affinity Fields." In *CVPR*, 2017.

[CY14] X. Chen and A. Yuille. "Articulated pose estimation by a graphical model with image dependent pairwise relations." In *NIPS*, 2014.

[DHG15] J. Donahue, L. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description." In *CVPR*, 2015.

[DR12] C. Desai and D. Ramanan. "Detecting Actions, Poses, and Objects with Relational Phraselets." In *ECCV*, 2012.

[DRC05] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. "Behavior Recognition via Sparse Spatio-Temporal Features." In *ICCV VS-PETS*, 2005.

[DT05] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection." In *CVPR*, 2005.

[EPF14] D. Eigen, C. Puhrsch, and R. Fergus. "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network." In *NIPS*, 2014.

[FGM10a] P. Felzenszwalb, R. Girshick, and D. McAllester. "Cascade object detection with deformable part models." In *CVPR*, 2010.

[FGM10b] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. "Object Detection with Discriminatively Trained Part Based Models." *PAMI*, **32**(9), 2010.

[FMR08] P. Felzenszwalb, D. McAllester, and D. Ramanan. "A Discriminatively Trained, Multiscale, Deformable Part Model." In *CVPR*, 2008.

[GBS05] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. "Actions as Space-Time Shapes." In *ICCV*, 2005.

[GBS07] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. "Actions as space-time shapes." *PAMI*, pp. 2247–2253, 2007.

[GDD14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." In *CVPR*, 2014.

[Gir15] R. Girshick. "Fast R-CNN." In *ICCV*, 2015.

[GTJ16] G. Gkioxari, A. Toshev, and N. Jaitlv. "Chained Predictions Using Convolutional Neural Networks." In *ECCV*, 2016.

[HG12] M. Hofmann and D. M. Gavrila. "Multi-view 3D Human Pose Estimation in Complex Environment." *IJCV*, **96**(1):103–124, 2012.

[HL16] Shuqin Xie Haoshu Fang and Cewu Lu. "RMPE: Regional Multi-person Pose Estimation." *arXiv preprint arXiv:1612.00137*, 2016.

[IPO14] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments." *PAMI*, **36**(7):1325–1339, 2014.

[JDL08] I. N. Junejo, E. Dexter, I. Laptev, and P. Patrick. "Cross-View Action Recognition from Temporal Self-Similarities." In *ECCV*, 2008.

[JGZ13] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. "Towards understanding action recognition." In *ICCV*, 2013.

[JJ93] M. I. Jordan and R. A. Jacobs. "Hierarchical mixtures of experts and the EM algorithm." In *IJCNN*, 1993.

[KG10] A. Kovashka and K. Grauman. "Learning a hierarchy of discriminative space-time neighborhood features for human action recognition." In *CVPR*, 2010.

[KG15] I. Kostrikov and J. Gall. "Depth sweep regression forests for estimating 3D human pose from images." In *BMVC*, 2015.

[KL15] A. Karpathy and F. Li. "Deep Visual-Semantic Alignments for Generating Image Descriptions." In *CVPR*, 2015.

[KPK10] M. P. Kumar, B. Packer, and D. Koller. "Curriculum Learning for Latent Structural SVM." In *NIPS*, 2010.

[KSH12] A. Krizhevsky, I. Sutskever, and G. Hinton. "Imagenet classification with deep convolutional neural networks." In *NIPS*, 2012.

[KTS14] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. Li. "Large-scale Video Classification with Convolutional Neural Networks." In *CVPR*, 2014.

[LC05] I. Laptev and R. Cedex. "On Space-Time Interest Points." *IJCV*, **64**(2-3):107–123, 2005.

[LC14] S. Li and A. Chan. "3D Human Pose Estimation from Monocular Images with Deep Convolutional Neural Network." In *ACCV*, 2014.

[LCS12] B. Li, O. I. Camps, and M. Sznaier. "Cross-view activity recognition using hankelets." In *CVPR*, 2012.

[LHW13] B. Li, W. Hu, T. Wu, and S. Zhu. "Modeling Occlusion by Discriminative AND-OR Structures." In *ICCV*, 2013.

[LK81] B. Lucas and T. Kanade. "An iterative image registration technique with an application to stereo vision." In *Proc. of the 7th intl joint conf. on Artificial intelligence*, 2081.

[LLS09] J. Liu, J. Luo, and M. Shah. "Recognizing realistic actions from videos." In *CVPR*, 2009.

[LMS08] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. "Learning realistic human actions from movies." In *CVPR*, 2008.

[LP07] I. Laptev and P. Perez. "Retrieving actions in movies." In *ICCV*, 2007.

[LSK11] J. Liu, M. Shah, B. Kuipers, and S. Savarese. "Cross-view action recognition via view knowledge transfer." In *CVPR*, 2011.

[LSL15] F. Liu, C. Shen, and G. Lin. "Deep Convolutional Neural Fields for Depth Estimation from a Single Image." In *CVPR*, 2015.

[LWY10] T. Lan, Y. Wang, W. Yang, and G. Mori. "Beyond actions: Discriminative models for contextual group activities." In *NIPS*, 2010.

[LWZ14] B. Li, T. Wu, and S. C. Zhu. "Integrating context and occlusion for car detection by hierarchical and-or model." In *ECCV*, 2014.

[LZ12] R. Li and T. Zickler. "Discriminative virtual views for cross-view action recognition." In *CVPR*, 2012.

[LZC15] S. Li, W. Zhang, and A. Chan. "Maximum-Margin Structured Learning With Deep Networks for 3D Human Pose Estimation." In *ICCV*, 2015.

[MBM11] S. Maji, L. Bourdev, and J. Malik. "Action recognition from a distributed representation of pose and appearance." In *CVPR*, 2011.

[MXY15] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. "Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN)." In *ICLR*, 2015.

[NWF08] J. Niebles, H. Wang, and L. Fei-fei. "Unsupervised learning of human action categories using spatial-temporal words." *IJCV*, 2008.

[NYD16] A. Newell, K. Yang, and J. Deng. "Stacked hourglass networks for human pose estimation." In *ECCV*, 2016.

[PAG13] L. Pishchulin, M. Andriluka, P. Gehler, and B. Schiele. "Strong Appearance and Expressive Spatial Models for Human Pose Estimation." In *ICCV*, 2013.

[PIT16] 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.

[PJZ11] M. Pei, Y. Jia, and S.-C. Zhu. "Parsing video events with goal inference and intent prediction." In *ICCV*, 2011.

[PR11] D. Park and D. Ramanan. "N-Best Maximal Decoders for Part Models." In *ICCV*, 2011.

[RHG15] S. Ren, K. He, R. Girshick, and J. Sun. "Fast R-CNN." In *NIPS*, 2015.

[RPZ13] B. Rothrock, S. Park, and S.C. Zhu. "Integrating Grammar and Segmentation for Human Pose Estimation." In *CVPR*, 2013.

[RS16] G. Rogez and C. Schmid. "Mocap-guided data augmentation for 3d pose estimation in the wild." In *NIPS*, 2016.

[SC12] S. Sadanand and J. Corso. "Action bank: A high-level representation of activity in video." In *CVPR*, 2012.

[SFC11] J. Shotton, A. W. Fitzgibbo, 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.

[SLC04] C. Schuldt, I. Laptev, and B. Caputo. "Recognizing human actions: A local SVM approach." In *ICPR*, 2004.

[SRZ16] T. Shu, M. S. Ryoo, and S. Zhu. "Learning Social Affordance for Human-Robot Interaction." In *IJCAI*, 2016.

[SWJ13] X. Song, T. Wu, Y. Jia, and S. C. Zhu. "Integrating context and occlusion for car detection by hierarchical and-or model." In *CVPR*, 2013.

[SYM14] H. Shen, S. Yu, D. Meng, and A. Hauptmann. "Unsupervised video adaptation for parsing human motion." In *ECCV*, 2014.

[SZ13] Z. Si and S.-C. Zhu. "Learning AND-OR Templates for Object Recognition and Detection." *PAMI*, 2013.

[SZ14a] K. Simonyan and A. Zisserman. "Two-stream convolutional networks for action recognition in videos." In *NIPS*, 2014.

[SZ14b] K. Simonyan and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *CoRR*, **abs/1409.1556**, 2014.

[TH] T. Tieleman and G. Hinton. "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude." In *Coursera: Neural networks for machine learning.*

[TJH05] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. "Large Margin Methods for Structured and Interdependent Output Variables." *JMLR*, pp. 1453–1484, 2005.

[TJL14] 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.

[TRL16] B. Tekin, A. Rozantsev, V. Lepetit, and P. Fua. "Direct prediction of 3d body poses from motion compensated sequences." In *CVPR*, 2016.

[TS14a] A. Toshev and C. Szegedy. "DeepPose: Human Pose Estimation via Deep Neural Networks." In *CVPR*, 2014.

[TS14b] A. Toshev and C. Szegedy. "DeepPose: Human pose estimation via deep neural networks." In *CVPR*, 2014.

[WBT09] G. Willems, J. Becker, T. Tuytelaars, and L. Van Gool. "Exemplar-based action recognition in video." In *BMVC*, 2009.

[WKS13] H. Wang, A. Klaser, C. Schmid, and C. Liu. "Dense trajectories and motion boundary descriptors for action recognition." *IJCV*, **103**(1):60–79, 2013.

[WLW12] J. Wang, Z. Liu, Y. Wu, and J. Yuan. "Mining Actionlet Ensemble for Action Recognition with Depth Cameras." In *CVPR*, 2012.

[WNX14] J. Wang, B. X. Nie, Y. Xia, Y. Wu, and S. C. Zhu. "Cross-view Action Modeling, Learning and Recognition." In *CVPR*, 2014.

[WQT15] L. Wang, Y. Qiao, and X. Tang. "Action recognition with trajectory-pooled deep-convolutional descriptors." In *CVPR*, 2015.

[WRK16] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. "Convolutional Pose Machines." In *ECCV*, 2016.

[WSL15] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A.L. Yuille. "Towards Unified Depth and Semantic Prediction from a Single Image." In *CVPR*, 2015.

[YF10] B. Yao and L. Fei-Fei. "Modeling mutual context of object and human pose in human-object interaction activities." In *CVPR*, 2010.

[YGG12] A. Yao, J. Gall, and L. Gool. "Coupled action recognition and pose estimation from multiple views." In *IJCV*, 2012.

[YIK16] H. Yasin, U. Iqbal, B. Kruger, A. Weber, and J. Gall. "A dualsource approach for 3d pose estimation from a single image." In *CVPR*, 2016.

[YL12] B. Yao and F. Li. "Action Recognition with Exemplar Based 2.5D Graph Matching." In *ECCV*, 2012.

[YR12] Y. Yang and D. Ramanan. "Articulated Human Detection with Flexible Mixtures of Parts." *PAMI*, **35**(12):2878–2890, 2012.

[YWY11] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. "Accurate 3D Pose Estimation From a Single Depth Image." In *ICCV*, 2011.

[YWZ14] L. Yang, T. Wu, and S. Zhu. "Online Object Tracking, Learning, and Parsing with And-Or Graphs." In *CVPR*, 2014.

[YYM10] W. Yang, Y.Wang, and G. Mori. "Recognizing human actions from still images with latent poses." In *CVPR*, 2010.

[ZJZ16] Yixin Zhu, Chenfanfu Jiang, Yibiao Zhao, Demetri Terzopoulos, and Song-Chun Zhu. "Inferring Forces and Learning Human Utilities From Videos." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[ZZD13] W. Zhang, M. Zhu, and K. Derpanis. "From Actemes to Action: A Strongly-supervised Representation for Detailed Action Understanding." In *ICCV*, 2013.

[ZZL16] X. Zhou, M. Zhu, S. Leonardos, K. G. Derpanis, and K. Daniilidis. "Sparseness meets deepness: 3d human pose estimation from monocular video." In *CVPR*, 2016.