Abstract

The pose Estimation is formulated as a DNN_based regressioin problem towards body joints, it has the advantage of reasoning about pose in a holistic fashion

Introduction

The problem of human pose estimation defined as the problem of localization of human joints.
challenges: strong articulations(粗大的铰接范围）, small and barely visible joints,occlusions and the need to capture context.
the **main stream** of work is motivated by the need to search in the large space of all possible articulated poses. and traditional efficient inference is achieved at the cose of limited expressiveness(the use of local detectors only modeling a small subset of all interactions between body parts).
We formulate the pose estimation as a joint regressoin problem , each body joint is regressed to using as an input the full image and a 7-layered generic convolutional DNN, there are 2 advantages:

1. DNN is capable of capturing the full context of each body jonit
2. simpler to formulate than methods based on graphical models, no need to explicitly design feature representations, no need to explicitly design a model topology and interactions between(no like the orientation definition in mixture tree-structured model).
   Further, we propose as cascade of DNN_based pose predictors.

Related Work

idea: representing articulated objects in general and human pose in particular.
the tractability is achieved by the rich part detectors, and recently use mixture model of parts or hierarchical model, but approaches which resoning about pose in a holistic manner have shown limited practicality,也有其他的论文用到卷积神经网络，但没用到级联。

Deep Learning Models for Pose Estimation

to express we encode the locations of all k body joints in pose vector, defined as
$y = (\dots, y_i^T, \dots)^T, i \in \{1, \dots, k\}$ where y_i contains the x and y coordinates of the i th joint.
we normalized by a box b with center b_c, width b_w and height b_h:

$$N(y_i; b) = \begin{pmatrix} 1/b_w & 0 \\ 0 & 1/b_h \end{pmatrix} (y_i - b_c) \quad (1)$$

3.1 Pose Estimation as DNN-based Regression

use the $\psi(x;\theta) \in R^{2k}$ as the regression functions, and considering using the normalization from (1) both for x and y，the pose prediction y^* in absolute image coordinates reads:

$$y^* = N^{-1}(\psi(N(x);\theta)) \quad (2)$$

the key is to select the ψ function: the network consists of 7 layers,Denote by C a convolutional layer, by LRN(local response normalization layer),P a pooling layers and F a fully connected layer, if we write the size of each layer i parentheses, the network can be described concisely as **C(55*5*96) - LRN - P - C(27*27*256) - LRN - P - C(13*13*384) - C(13*13*384) - C(13*13*256) - P - F(4096) - F(4096),the filter for the first two C layers is 11x11 and 5x5 and for the remainig three is 3x3**.网络结构如下图:

此外，这个模型不是面向特定领域的，并且没有显式的学习关节之间的关系。

3.3.1 Training

The difference to the refer model(Imagenet classification with DPCNN) is loss ,we train a linear regression on top of last network layer to predict a pose vector by minimizing L2 distance.
First, We normalize out training set D using the normalization:

$$D_N = \{(N(x), N(y))|(x, y) \in D\} \quad (3)$$

then the L2 loss:

$$arg\min_{\theta} \sum_{(x,y)\in D_N} \sum_{i=1}^{k} \|y_i - \psi_i(x;\theta)\|_2^2$$

params:size of mini-batch:128, learning rate: 0.0005, dropout of F layers: 0.6

3.2 Cascade of Pose Regressors:

for the capacity to look detail(the previous input size is 220x220, which is limited) and not increase the large number of parameters), it porpose to train a cascade of pose regressors.

- first the cascade starts off by estimating an initial pose as outlined ni the previous
- then additional DNN regressors are trained to predict a displacement of the joint locations from previous stage to the true location.
- each subsequent stage uses the predicted joint locations to focus on the relevant parts of the image.

即每个阶段每个子图象关注图片的不同位置，在上一阶段待预测的关节位置，该子图象被裁剪，然后上一阶段该位置的姿势位移回归量也被用于该子图象。在这个过程中，我们使用相同的网络结构但学习不同的参数。
我们使用关节限定盒b_i获取y_i附件的子图象： $b_i(y;\delta) = (y_i, \delta diam(y), \delta diam(y))$, 其中diameter(y) 表示姿态中对应关节的距离。
基于上文的描述，在stage1我们使用一个包含整张图片的界限盒b_0:

$$\text{Stage } 1: \quad \mathbf{y}^1 \leftarrow N^{-1}(\psi(N(x; b^0); \theta_1); b^0) \tag{5}$$

At each subsequent stage $s \geq 2$, for all joints $i \in \{1, \ldots, k\}$ we regress first towards a refinement displacement $\mathbf{y}_i^s - \mathbf{y}_i^{(s-1)}$ by applying a regressor on the sub image defined by $b_i^{(s-1)}$ from previous stage $(s-1)$. Then, we estimate new joint boxes b_i^s:

$$\text{Stage } s: \quad \mathbf{y}_i^s \leftarrow \mathbf{y}_i^{(s-1)} + N^{-1}(\psi_i(N(x; b); \theta_s); b) \tag{6}$$
$$\text{for } b = b_i^{(s-1)}$$
$$b_i^s \leftarrow (\mathbf{y}_i^s, \sigma \text{diam}(\mathbf{y}^s), \sigma \text{diam}(\mathbf{y}^s)) \tag{7}$$

3.2.1 Training

为了扩大样本的丰富性，我们生成仿真数据，通过从符合全局数据均值、方差的高斯分布中随机生成位移量施加在原始图片中，然后将该生成的训练数据与原始数据均匀混合。