DeepPose & Convolutional Pose Machines
Main Concepts

1. CNN with regressor head.
2. Object Localization.
3. Bigger to smaller or Smaller to bigger view.
4. Deep Supervised learning to prevent Vanishing gradient problem in deep nets
CNN with Regressor head

slides: andrej et al

Image → and Pooling → Final conv feature map → Fully-connected layers → Box coordinates

“Regression head”
CNN with Regressor Head eg code in keras

conv = Convolution2D(64,10,10,)

flat = Flatten()(conv)

left_eye = Dense(1, activation='linear', name='A')(flat)

right_eye = Dense(1, activation='linear', name='B')(flat)

model = Model(input=frame_in, output=[left_eye,right_eye])

model.compile(loss='mse')  #mse= L2 loss ,mean square error,
Object localization

**Simple Recipe for Classification + Localization**

**Step 2:** Attach new fully-connected “regression head” to the network

- **Image**
- **Convolution and Pooling**
- **Final conv feature map**
- **“Classification head”** (Fully-connected layers, Class scores)
- **“Regression head”** (Fully-connected layers, Box coordinates)
Object localization

**Simple Recipe for Classification + Localization**

**Step 3:** Train the regression head only with SGD and L2 loss
Human Pose as Object Localization but our object is non-rigid

Aside: Human Pose Estimation

Represent a person by K joints

Regress \((x, y)\) for each joint from last fully-connected layer of AlexNet

(Details: Normalized coordinates, iterative refinement)

Bigger to small or Small to Big(View)

1. First paper takes First Coarse view of full image then goes for magnified view of parts.
2. Second Paper first takes small Receptive field and as depth increases image view increases.
Deep Supervision

THAT'S NOT ENOUGH

WE HAVE TO GO DEEPER
What’s Deep Supervision?
Why it’s needed? Vanishing Gradient

Generally $|w_j| < 1$,

So, gradient will decrease, As depth increases.
Are there other solutions?

1. Yes, Proper weight initialization.

2. What LSTMS did for RNN, ResNet’s did for FeedForward NN.
Paper 1: DeepPose
Why is it Hard?

large variance  occlusion  L/R ambiguity
Background

1. Pictorial structures: parts and tree based relations between them based on some priors.

2. Non tree models: The first layer acts as a discriminative, independent body part classifier. The second layer takes the estimated class distributions of the first one into account and is thereby able to predict joint locations by modeling the interdependence and co-occurrence of the parts.
Problem Specification

Encode the locations of all “k” body joints in pose vector defined as $y = (...,y_i,...), i \in \{1,...,k\}$, where $y_i$ contains the x and y coordinates of the ith joint.
Bounding Box Trick

Lets define a bounding box, \( b = (bc, bw, bh) \) in absolute image coordinates

Where \( bc \) = center of box

\( bh \) = height of box

\( bw \) = width of box
Transporting point to BOX coordinate from Absolute image coordinate

Any point $Y_i$ from absolute to BOX coordinate system $b$. 

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

Yi=(4,4) in absolute frame

BOX b=(bc=(2,2),bw=4,bh=4)

Yi in BOX coordinate=(½,1/2)

Similarly Y=(0,0) becomes(-½,-1/2)
Why Box Coordinate?

For our network it becomes very easy to work in this normalized environment, Variance in Images. Large, Small etc.

Can always take inverse to get back Absolute Coordinates wrt to image.
Some Assumptions

From Now On We will work in Normalized box coordinate system.

Our network will give us predictions in Normalized BOX coordinate system.

Our loss function(L2) will also work in BOX frame.

We will take inverse to get back absolute coordinates.
Algo: we will work in stages to improve

Basicly AlexNet with K regressor heads
Stage: 1

Where $N(x,b)$ in transformation in box b coordinate system,

$B_0$ is full image or by person detector

$\Psi$ is neural Net with parameters $\Theta_1$

Stage 1:  $y^1 \leftarrow N^{-1}(\psi(N(x; b^0); \theta_1); b^0)$
Stage 2 onwards:

Note: For next stages we are only predicting displacement of improvement.

\[
\text{Stage } s: \quad y^s_i \leftarrow y^{(s-1)}_i + N^{-1}(\psi_i(N(x; b); \theta_s); b)
\]

Note: bounding box evaluation, from prev state

\[
b^s_i \leftarrow (y^s_i, \sigma \text{diam}(y^s), \sigma \text{diam}(y^s))
\]

Where \(\sigma\) is hyperparameter, \(\text{diam}(y)\) is distance btw opposite joints in human torso. Left shoulder right hip,
LOSS function:

Everything happening in BOX frame

\[ D_N = \{(N(x), N(y))|(x, y) \in D\} \]  \hspace{1cm} (3)

Then the $L_2$ loss for obtaining optimal network parameters reads:

\[ \arg \min_{\theta} \sum_{(x, y) \in D_N} \sum_{i=1}^{k} \| y_i - \psi_i(x; \theta) \|_2^2 \]  \hspace{1cm} (4)
Benchmark Datasets

<table>
<thead>
<tr>
<th></th>
<th>FLIC</th>
<th>LSP</th>
<th>MPII</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>3987 training</td>
<td>11000 training</td>
<td>29116 training</td>
</tr>
<tr>
<td></td>
<td>1016 testing</td>
<td>1000 testing</td>
<td>11823 testing</td>
</tr>
<tr>
<td>type</td>
<td>movie scenes</td>
<td>sports</td>
<td>diverse</td>
</tr>
<tr>
<td>annotation</td>
<td>upper body</td>
<td>full body</td>
<td>full body w/ truncation</td>
</tr>
</tbody>
</table>
Dataset setup

FLIC: Hollywood

Used Body detector to get initial bounding box. Face based.

LSP: Sports

Directly Used. Since Humans are tightly cropped in image.
Dataset Validation

FLIC: Hollywood
Scalar sigma=1.0

LSP: Sports
Scalar sigma=2.0

Number of stages:
S=3 after algo stopped improvement on validation dataset.
Metrics: Percent of Detected Joint (PDJ)

A joint is considered to be detected if the distance between the predicted and the true joint is within a certain fraction of the torso diameter.

By varying this fraction detection rates are obtained for varying degrees of localization.
Metrics Percent Correct Parts

A candidate body part is labeled as correct if its segment endpoints lie within 50% of the length of the ground-truth annotated endpoints.

Penalizes shorter limbs a lot which are harder to detect
Some Results

Figure 3. Percentage of detected joints (PDJ) on FLIC for two joints: elbow and wrist. We compare DeepPose, after two cascade stages, with four other approaches.
Effects of iterative refinement

Figure 5. Percent of detected joints (PDJ) on FLIC or the first three stages of the DNN cascade. We present results over larger spectrum of normalized distances between prediction and ground truth.
Effects of iterative refinement

Figure 6. Predicted poses in red and ground truth poses in green for the first three stages of a cascade for three examples.
Take Away

Use DNN regression for accurate object localization.

Very Easier to implement

Iterative refinement works for non rigid objects as well.
Where $g_t$ is multiclass(k) predictor making belief maps.

**Convolutional Pose Machines (CPMs)**
Convolutional Pose Machines
Capturing Local Appearance by FCNN
Convoluotional Pose Machines
Learning Image-dependent Spatial Model

Stage 1

Stage 2

Stage T

Stage 1

Stage 2

Loss
Stage 1

\[
\begin{align*}
\text{conv1\_stage1} & = \text{layers.conv2d}(\text{image}, 64, 9(\text{kernel}), 1(\text{stride})) \\
\text{pool1\_stage1} & = \text{layers.max\_pool2d}(\text{conv1\_stage1}, 2(\text{kernel}), 2(\text{stride})) \\
\text{conv2\_stage1} & = \text{layers.conv2d}(\text{pool1\_stage1}, 64, 9(\text{kernel}), 1(\text{stride})) \\
\text{pool2\_stage1} & = \text{layers.max\_pool2d}(\text{conv2\_stage1}, 2(\text{kernel}), 2(\text{stride})) \\
\text{conv3\_stage1} & = \text{layers.conv2d}(\text{pool2\_stage1}, 64, 9(\text{kernel}), 1(\text{stride})) \\
\text{pool3\_stage1} & = \text{layers.max\_pool2d}(\text{conv3\_stage1}, 2(\text{kernel}), 2(\text{stride})) \\
\text{conv4\_stage1} & = \text{layers.conv2d}(\text{pool3\_stage1}, 64, 5(\text{kernel}), 1(\text{stride})) \\
\text{conv5\_stage1} & = \text{layers.conv2d}(\text{conv4\_stage1}, 64, 5(\text{kernel}), 1(\text{stride})) \\
\text{conv6\_stage1} & = \text{layers.conv2d}(\text{conv5\_stage1}, 64, 1(\text{kernel}), 1(\text{stride})) \\
\text{conv7\_stage1} & = \text{layers.conv2d}(\text{conv6\_stage1}, 10(\text{p}+1), 1(\text{kernel}), 1(\text{stride}))
\end{align*}
\]
Stage 2

\[
conv4\_stage2 = \text{layers.conv2d}(pool3\_stage2, 64, 5(\text{kernel}), 1(\text{stride}))
\]

\[
concat\_stage2 = \text{tf.concat}([\text{conv7}\_\text{stage1}, \text{conv4}\_\text{stage2}])
\]

\[
Mconv1\_\text{stage2} = \text{layers.conv2d}(concat\_stage2, 64, 11(\text{kernel}), 1(\text{stride}))
\]

\[
\text{conv6}\_\text{stage3} = \text{layers.conv2d}(Mconv1\_\text{stage2}, 10(p+1), 1(\text{kernel}), 1(\text{stride}))
\]

\[
\text{model} = \text{Model}(\text{input=\text{image}}, \text{output=}[[\text{conv6}\_\text{stage3}, \text{conv7}\_\text{stage1}])}
\]

\[
\text{model.compile}(<\text{loss='mse'}>) \ #\text{mse= L2 loss ,mean square error,}
\]
Large Receptive Field

Stage 1

Stage 2

Stage T

Stage 1

Stage 2

Stage T
Convolutional Pose Machines
Overall Architecture

Stage 1

Stage 2

Stage T

Note: feature sharing from stage 2 onwards
Iteratively Refined Confidence Maps

right elbow

right wrist

Input Image  1st stage  2nd stage  3rd stage
Recover from False Negative

1st stage

R. Elbow

2nd stage

R. Elbow

3rd stage

R. Elbow
Training CPMs
Ideal Confidence Maps for Intermediate Supervisions

\[ f_t = \| \quad - \quad \| F \]

overall loss \[ \mathcal{F} = \sum_{t=1}^{T} f_t \]
Training styles
Training CPMs
Intermediate Supervisions Resolves Gradient Vanishing
Histograms of Gradient Magnitude During Training

- Input
  - Layer 1
  - Layer 3
  - Layer 6
- Stage 1: Supervision
  - Layer 7
  - Layer 9
  - Layer 12
- Stage 2: Supervision
  - Layer 13
  - Layer 15
  - Layer 18
- Output

<table>
<thead>
<tr>
<th>Epoch 1</th>
<th>Epoch 2</th>
<th>Epoch 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Graphs for Epoch 1" /></td>
<td><img src="image2.png" alt="Graphs for Epoch 2" /></td>
<td><img src="image3.png" alt="Graphs for Epoch 3" /></td>
</tr>
</tbody>
</table>

- Gradient ($\times 10^{-3}$)
- Black: With Intermediate Supervision
- Red: Without Intermediate Supervision
Dataset Cropping

Train:

Roughly resize images to have same scale.

Crop or Pad image according to center positions of human to make it 368*368.

Rough scale estimates are provided if not estimate them using joints.

Testing:

They do resizing and cropping(padding)
MPII dataset
MPII dataset with different viewpoints

Figure 11: Comparing PCKh-0.5 across various viewpoints in the MPII dataset. Our method is significantly better in all the viewpoints.
FLIC

Beats the prev paper, see black is way above blue.

Figure 12: Quantitative results on the FLIC dataset for the elbow and wrist joints with a 4-stage CPM. We outperform all competing methods.
Failure Cases

L/R confusion  rare viewpoint  rare pose  severe occlusion

right wrist
Summary:

Iterative refining is the key.

Vanishing Gradients, Use Deep Supervision

This architecture can be extended to multiple persons.

I don’t know why authors didn’t used ResNet.

Try to model human 3d model from single image.