Associative Embedding: End-to-End Learning forJoint Detection and Grouping
(2017)

Alejandro Newell et al.
Resume

April 26, 2019

1 Introduction

This paper introduces associative embeddings for joint detection and grouping, it is used to do multiple joint detection and instance segmentation in an end to end manner instead of multistate pipelines, in this case, the networks outputs at the same time the detections, probabilities of each pixel beloning to a given joint, but also tags per pixel per joint (here tags are 1D), and the joints beloning to the same person need to have similar tag, and by ground the pixels with the same tags, we can differentiate between different people instances.

The training of the netwok to output similar tags to similar persons is done in two parts, first by pulling all the tags of the same person together, and secondly by pushing the tags (their mean) of different people away.

2 The method

2.1 Stacked Hourglass Network

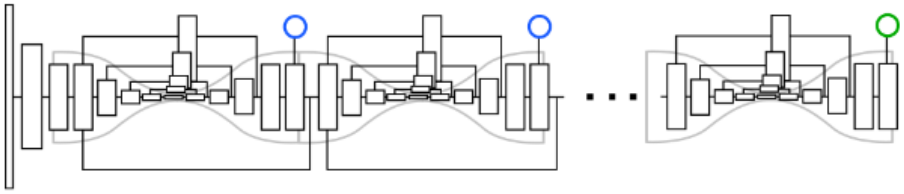

Figure 1: We use the stacked hourglass architecture from Newell et al. [31]. The network performs repeated bottom-up, top-down inference producing a series of intermediate predictions (marked in blue) until the last "hourglass" produces a final result (marked in green). Each box represents a 3x3 convolutional layer. Features are combined across scales by upsampling and performing elementwise addition. The same ground truth is enforced across all predictions made by the network.

In this work, they use stacked hour glass net as their network for keypoint detection, outputing the tags in addition to the detection probabilities, each hourglass has a standard set of convolutions and pooling layers that process features down to low resolution capturing the full context, and then upsampling these features until reaching the final resolution, and then we stack multipple hourglasses to enable a repeated bottom up and top town inference to produce more accurate predicitons, they make some sligh changes, like increasing the number of features when reducing the spatial dimensions and unsing conv 3x3 instead of residual blocks.

2.2 multi person pose estimation

The networks output a set of probabilities for each joint at each pixel in the image, and a tag for each joint at each pixel in the image, so for m joints, the number of channels will be $2m$, m for the detection of each joint and m for the tags, each is 1D,

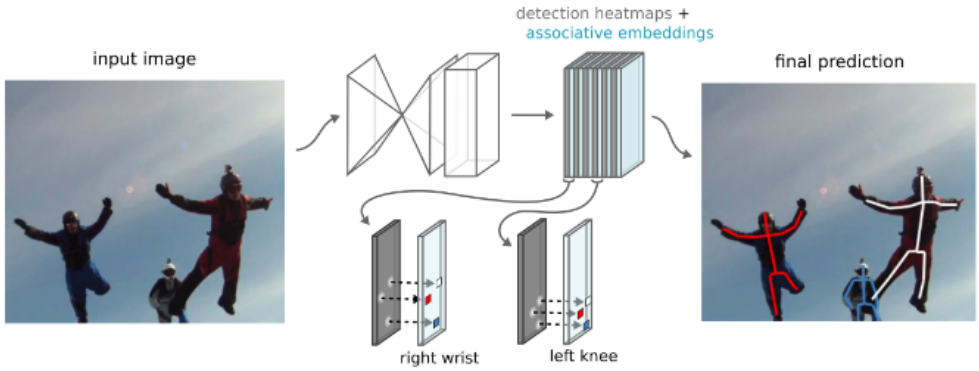

Figure 2: An overview of our approach for producing multi-person pose estimates. For each joint of the body, the network simultaneously produces detection heatmaps and predicts associative embedding tags. We take the top detections for each joint and match them to other detections that share the same embedding tag to produce a final set of individual pose predictions.

for the detection the loss is a simple MSE between each prediction heatmap and the ground truth, for the tags, then the objective is to join the tags of the same person together and push the others away, this is done by imposing a grouping loss on the outputs, the loss assesses how well the predicted tags agree with the ground thruth grouping indirectely, the loss pulls the tags of the joints belonging to the same person together, and pushing the tags of the joints of different people away, and the objective is to have:

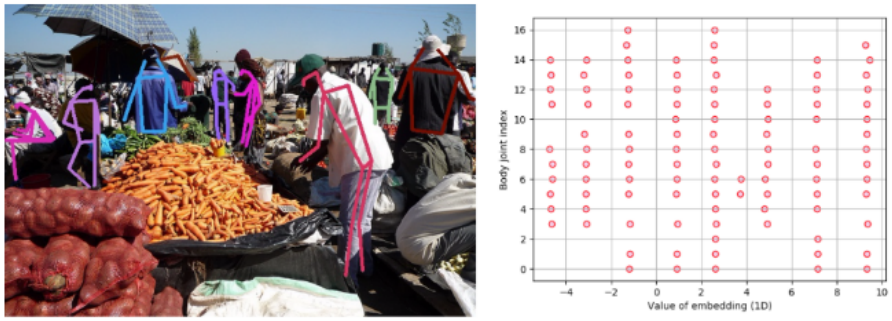

Figure 3: Tags produced by our network on a held-out validation image from the MS-COCO training set. The tag values are already well separated and decoding the groups is straightforward.

Formally, let $h_k \in R^{W \times H}$ be the predicted tagging heatmap for the k-th body joint, where $h(x)$ is a tag value at pixel location x. Given N people, let the ground truth body joint locations be $T = \{(x_{nk})\}, n = 1, \ldots, N, k = 1 \ldots, K$ where xnk is the ground truth pixel location of the k-th body joint of the n-th person. Assuming all K joints are annotated, the reference embedding for thenth person would be:

$$\overline{h}_n = \frac{1}{K} \sum_k h_k \left(x_{nk}\right)$$

The grouping loss L_g is then defined as

$$L_g(h, T) = \frac{1}{NK} \sum_n \sum_k \left(\overline{h}_n - h_k \left(x_{nk}\right)\right)^2 + \frac{1}{N^2} \sum_n \sum_{n'} \exp\left\{-\frac{1}{2\sigma^2} \left(\overline{h}_n - \overline{h}_{n'}\right)^2\right\}$$

The first half of the loss pulls together all of the embeddings belonging to an individual, and the second half pushes apart embeddings across people. We use avalue of 1 in our training

Once the network has been trained, decoding is straightforward. We perform non-maximum suppres-sion on the detection heatmaps and threshold to get a set of detections for each body joint. Then, we go through the detection one joint (starting by the neck, for example) at the time, first we create new groups for all the detections / joints, and then in the second iteration we compare the tags of these joints to the tags of each group, and assign the tag to the group bellow a certain threshold, and if can't find any group to assign it to, we can simply create a new group, given that we can have some instances with a limited number of joints.

For multi scale testing, they pass the image through the network at different scales and combine the detection, and the concatenate the tags (now the distances are measured between vectors and not scalers).

2.3 Instace segmentation

For instance segmentation the problem is similar, the network output the probability detection of each pixel (backgroud of foreground), and tag heatmap, and this time instead of having a loss between all the pixel, it is easier to sample an set of N pixels from each instance using the groundthruth, and then calculate the loss using the tags of these N samples, more formely let $h \in R^{W \times H}$ be a predicted WH tagging heat map. Let $x$denote a pixel location and $h(x)$ the tag at the location, and let $S_n = x_{kn}, k = 1, \ldots, K$ be a set of locations randomly sampled within the n-th object instance. The grouping loss L_g is defined as

$$L_g(h,T) = \sum_n \sum_{x \in S_n} \sum_{x' \in S_n} \left(h(x) - h\left(x'\right)\right)^2 + \sum_n \sum_{n'} \sum_{x \in S_n} \sum_{x' \in S_{n'}} \exp\left\{-\frac{1}{2\sigma^2}\left(h(x) - h\left(x'\right)\right)^2\right\}$$

To decode the output of the network, we first thresholdon the detection channel heatmap to produce a binary mask, Then, we look at the distribution of tags within this mask. Wecalculate a histogram of the tags and perform non-maximumsuppression to determine a set of values to use as identifiersfor each object instance. Each pixel from the detection maskis then assigned to the object with the closest tag value.

3 Experiments

	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
Iqbal&Gall, ECCV16 [21]	58.4	53.9	44.5	35.0	42.2	36.7	31.1	43.1
Insafutdinov et al., ECCV16 [20]	78.4	72.5	60.2	51.0	57.2	52.0	45.4	59.5
Insafutdinov et al., arXiv16a [35]	89.4	84.5	70.4	59.3	68.9	62.7	54.6	70.0
Levinkov et al., CVPR17 [25]	89.8	85.2	71.8	59.6	71.1	63.0	53.5	70.6
Insafutdinov et al., CVPR17 [19]	88.8	87.0	75.9	64.9	74.2	68.8	60.5	74.3
Cao et al., CVPR17 [6]	91.2	87.6	77.7	66.8	75.4	68.9	61.7	75.6
Fang et al., ICCV17 [10]	88.4	86.5	78.6	**70.4**	74.4	**73.0**	**65.8**	76.7
Our method	**92.1**	**89.3**	**78.9**	69.8	**76.2**	71.6	64.7	**77.5**

Table 1: Results (AP) on MPII Multi-Person.

	AP	AP^{50}	AP^{75}	AP^M	AP^L	AR	AR^{50}	AR^{75}	AR^M	AR^L
CMU-Pose [6]	0.611	0.844	0.667	0.558	0.684	0.665	0.872	0.718	0.602	0.749
G-RMI [33]	0.643	0.846	0.704	**0.614**	0.696	0.698	0.885	0.755	0.644	0.771
Our method	**0.663**	**0.865**	**0.727**	0.613	**0.732**	**0.715**	**0.897**	**0.772**	**0.662**	**0.787**

Table 2: Results on MS-COCO **test-std**, excluding systems trained with external data.

	AP	AP^{50}	AP^{75}	AP^M	AP^L	AR	AR^{50}	AR^{75}	AR^M	AR^L
CMU-Pose [6]	0.618	0.849	0.675	0.571	0.682	0.665	0.872	0.718	0.606	0.746
Mask-RCNN [17]	0.627	**0.870**	0.684	0.574	0.711	–	–	–	–	–
G-RMI [33]	0.649	0.855	0.713	**0.623**	0.700	0.697	0.887	0.755	0.644	0.771
Our method	**0.655**	0.868	**0.723**	0.606	**0.726**	**0.702**	**0.895**	**0.760**	**0.646**	**0.781**

Table 3: Results on MS-COCO **test-dev**, excluding systems trained with external data.