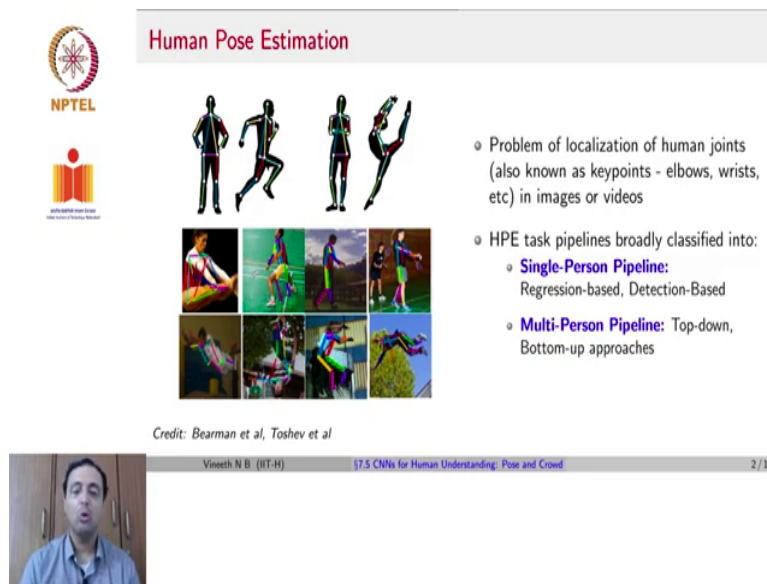


Deep Learning for Computer Vision
Professor Vineeth N Balasubramanian
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad
Lecture 50

CNNs for Human Understanding Human Pose and Crowd

Beyond face understanding, CNNs have been used for many other human understanding tasks; gesture recognition, emotion recognition, gait recognition, so on and so forth. But to give a flavor of different varieties of tasks, we will now look at two other tasks, human pose estimation and crowd counting, using deep learning and CNNs.

(Refer Slide Time: 00:52)



Human Pose Estimation

- Problem of localization of human joints (also known as keypoints - elbows, wrists, etc) in images or videos
- HPE task pipelines broadly classified into:
 - **Single-Person Pipeline:** Regression-based, Detection-Based
 - **Multi-Person Pipeline:** Top-down, Bottom-up approaches

Credit: Bearman et al, Toshev et al

Vineeth N B. (IIT-H) 57.5 CNNs for Human Understanding: Pose and Crowd 2 / 19



The task of human pose estimation is the problem of localization of human joints, such as elbows, wrists, so on and so forth in both images and videos, where would such a task be used, it could be for sports analytics, it could be for the Microsoft Xbox that detects your pose and accordingly asks your avatar on the screen to play a particular tennis shot or a golf swing.

Existing methods, especially deep learning based methods for human pose estimation, are broadly categorized into single person pipelines, where you are trying to get the pose of just a single person in the frame, or a multi person pipeline, where there could be multiple people in the frame. And you would like to know the pose of each of them.

What you see here are illustrations of how a pose estimation model works. What we are ideally looking for is the positions of each of these joints that you see on any of these images.

As you can see here, there are many challenges. Firstly, this seems different from the tasks that we have seen so far, different from image level classification, or detection or segmentation or face verification as another task. And we will see how this is done using CNNs in this lecture. Beyond that, you can also see that when occlusions come into play, especially self-occlusions where a part of the human body occludes another part, this task can get very challenging.

(Refer Slide Time: 02:52)


HPE: How to evaluate?

- **Percentage of Correct Parts (PCP):** A limb is considered detected if distance between detected joint and true joint $<$ half limb length (denoted as PCP@0.5)
- **Percentage of Detected Joints (PDJ):** A detected joint is correct if distance between predicted and true joint is within certain fraction of torso diameter; e.g. PDJ@0.2 \implies distance between predicted and true joint $<$ $0.2 \times$ torso diameter
- **Object Keypoint Similarity (OKS) based mAP:**

$$OKS = \frac{\sum_i \exp\left(\frac{-d_i^2}{2s^2k_i^2}\right)\delta(v_i > 0)}{\sum_i \delta(v_i > 0)}$$

where d_i is Euclidean distance between detected keypoint and corresponding ground truth, v_i is visibility flag of ground truth, s is object scale, and k is per-keypoint constant that controls falloff (OKS is IoU equivalent for keypoint evaluation)

Credit: A 2019 Guide to Human Pose Estimation with Deep Learning by Nanonets
 Vineth R.B. (IIT-H) 57.5 CNNs for Human Understanding: Pose and Crowd 3/19



Before we actually go ahead and discuss the methods, we will try to first ask the question, how would you know whether the model that you developed for human pose estimation is good or not? How do you evaluate it? There are well known metrics today. So some of the metrics are PCP, which stands for percentage of correct parts, which states that a limb is considered detected, if the distance between the detected joint and the true joint is less than half the limb's length.

So you could have a long limb or a short limb depending on which part of the body you are trying to model. And if the distance between the predicted joint position and the correct joint position is less than half the length of that limb, we consider that to be a fairly correct prediction. This is known as PCP at 0.5, if you considered quarter limb length, it would be PCP at 0.25.

A related metric is known as percentage of detected joints PDJ, which states that the detected joint is correct, if the distance between the predicted and the detected joint is within a certain fraction of the torso diameter. So you could look at the torso diameter as a central scale for

the person that you are modeling in that particular, when you are trying to predict the pose of that person. So, if you say PDJ at 0.2, you want to ensure that the distance between the predicted and the true joint is less than 0.2 times the torso diameter of the person under consideration.

There has also been a different metric known as object key point similarity based mAP which you can say is an equivalent of IOU for human pose estimation, which is given by



$$\sum_i \exp\left(\frac{-d_i^2}{2s^2k_i^2}\right) \times \text{an impulse function to check whether } v_i > 0 \text{ or not.}$$

Let us try to explain each of these quantities, denominator is going to be over all possible i's, d_i here is the Euclidean distance between a detected key point and the corresponding ground truth, which is the actual presence of the joint. So that is the d_i that you see in the numerator here. v_i is the visibility flag of the ground truth, as we said, there could be certain joints that are occluded by other joints, and trying to get a correct estimate of that joint would be an impossible task.

So by ensuring that you have a visibility flag for each joint, you know that if a joint was not visible, any error on the joint can be weighted to a lower extent. So v_i is a visibility flag of a ground truth, s is the object scale. So this distance between the detected point and ground truth has to be normalized by the scale of that particular object.

So if that object is very large in the image, this distance could be relatively larger than for another human in the same image, where the object scale is small. Finally, k is a per key point user defined constant just to control fall off that you could consider as a hyper parameter. So as you can see here, we now have a metric that is an inverse exponent of the distance which means it gives an extent of how good the prediction is, but it factors in scale, as well as visibility of the joint to evaluate your final performance.

(Refer Slide Time: 06:42)

Regression-based Methods: Iterative Error Feedback²

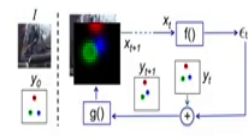
- Mean pose recursively updated to match ground truth
- Given image concatenated with output representation, f is trained to predict "correction" that brings mean poses closer to ground truth
- Mathematically:

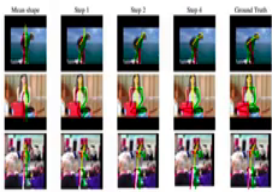
$$x_0 = I$$

$$\epsilon_t = f(x_t)$$

$$y_{t+1} = y_t + \epsilon_t$$

$$x_{t+1} = x_t \oplus g(y_{t+1})$$





²Carreira et al, Human Pose Estimation with Iterative Error Feedback, CVPR 2016

Vineeth N B. (IIT-H)
[7.5 CNNs for Human Understanding: Pose and Crowd]
5 / 19

With these metrics specified, let us now talk about methods for human pose estimation using deep neural network models. The precursor of all of such models was called DeepPose, which was proposed in 2014. This was perhaps the first work to kick off deep learning based human pose estimation. In this approach, an image was given to a model and the architecture at that time was of course, AlexNet inspired because in 2014, that was one of the most common models in practice, the only change from the original AlexNet architecture is in the output space.



Now, you are not predicting a class label for the entire image, but you are going to predict a set of joint positions, 2D joint positions, x_i, y_i for each joint on your input image. So, your output is given by y , which is a set of y_i 's, where each y_i contains x y coordinates for the corresponding joint. What would be the loss to use here? The loss would be a mean square error for the position of every joint or the L 2 norm for the position of every joint with respect to the ground truth of that joint.

This method did not stop here, to further improve the performance, it also used what are known as cascaded regressors to improve the precision of the location of each joint. What did this mean? Once you identify a joint position in your initial stage, you crop out a region around the joint position and that region is scaled up to the input of the entire CNN.

And this again predicts a refined value of the joint position inside that patch, which helps you fine tune your joint position over multiple stages. So these cropped images are fed into the

network in next stages and this ensures that across the final image scales, you get precise Key Point locations as output.

(Refer Slide Time: 8:58)

Regression-based Methods: Iterative Error Feedback²

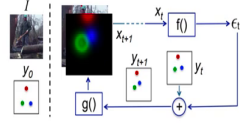
- Mean pose recursively updated to match ground truth
- Given image concatenated with output representation, f is trained to predict "correction" that brings mean poses closer to ground truth
- Mathematically:

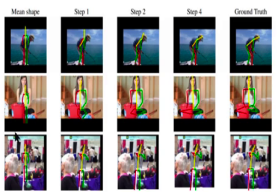
$$x_0 = I$$

$$\epsilon_t = f(x_t)$$

$$y_{t+1} = y_t + \epsilon_t$$

$$x_{t+1} = x_t \oplus g(y_{t+1})$$





²Carreira et al, Human Pose Estimation with Iterative Error Feedback, CVPR 2016

Vineeth N B. (IIT-H) 57.5 CNNs for Human Understanding: Pose and Crowd 5 / 19

Another approach which took this in a different direction, which was also an iterative error feedback based approach, but approached this differently, was a method proposed in 2016 where the entire prediction started with a mean pose skeleton, which is then updated iteratively over many steps. So you have an input image, and you have an average human pose skeleton here. And the job of the neural network is only to predict the deviation from the mean pose skeleton. So given an image concatenated with its output representation, the neural network now is trained to predict the correction that brings the mean pose closer to the ground truth.

Let us see this in a bit more detail. In the first stage, x_0 would be the image itself that is the image that is given here. And this image is given to the neural network, or the neural network predicts an ϵ_t for each joint, which would be the deviation from the mean pose that is the ϵ_t here. This ϵ_t is added to y_t which was the pose in your previous step, and you get a new pose y_{t+1} .

And now this y_{t+1} is overlaid on the image and the patch around it is now used to again refine the error and the deviation to get a new ϵ_t and this is now iterated over and over again to get a final estimate of the pose. Visually, here are a couple of illustrations. So you see here

that on this image, the mean pose is overlaid on this image, you can see the mean pose is often a person standing upright.

And in every step, the mean pose is adjusted towards getting the pose of this particular person. And you can see that after four steps, the predicted pose becomes close to the ground truth, which is shown in the last column. You see a couple of more examples here, where this can be challenging, you can see once again here, the standing pose, and over a few steps, you get the pose of the person squatting on a particular location.

(Refer Slide Time: 11:31)

Detection-based Methods³

NPTEL

Overview of Cascaded Architecture

Crop module functionality for a single joint

The fine heat-map network for a single joint

Recover spatial accuracy lost due to pooling of model by using additional ConvNet to refine localization result of coarse heat-map

³Tompson et al, Efficient Object Localization using Convolutional Networks, CVPR 2015

Vineeth N B (IIT-H) [7.5 CNNs for Human Understanding: Pose and Crowd] 6 / 19

While regression based methods that we saw on the earlier slides try to predict the joint location, another family of methods for human pose estimation are detection based, where at single shot, try to get the regions of each of these joints as a heat map. Methods that have used this kind of an approach, one of which was proposed in 2015, try to also employ a course to find multi scale strategy to help refine these heat maps to get better joint localizations.

Let us look at this approach here. So in this approach, an input image is given to a coarse heat map model, which gives you the heat map of joint locations, this could be a standard CNN backbone with minor modifications. And for each joint that you have predicted here around that could be located at the center of a particular heat map, you crop out a patch around it, and then have a fine heat map model, which improves the localization of the joint in that particular location.



Let us now see this fine heat map model in a bit more detail. So for each joint location in the coarse heat map, which could be the center of a particular region of the heat map, you have a multi scale pyramid, if the original image was 64×64 , you have 128×128 and 256×256 versions of it, you correspondingly crop out a 9×9 and 18×18 and a 36×36 region.

And now each of these three regions go through separate convolutional pipelines. You can see here that the 9×9 goes through this pipeline, the 18×18 goes through two different pipelines with different convolutional filters, and the 36×36 goes through different

pipeline, different convolutional plus relu layers. And now all of these are upsampled to 36×36 , obviously the top one does not need upsampling.

The remaining ones are up sampled to 36×36 . All these feature maps are concatenated to make the final prediction, which is more precise for that particular joint. This refinement process helps the final performance.

(Refer Slide Time: 14:00)

Multi-Person Pose Estimation: Top-Down Pipeline

Image

→

Human detection

→

Keypoints estimation for detected box

→

Post processing

→

Result

- Detect all persons from given image
- Single-person approaches performed in each detected bounding box
- Context information from whole image can be used to improve performance

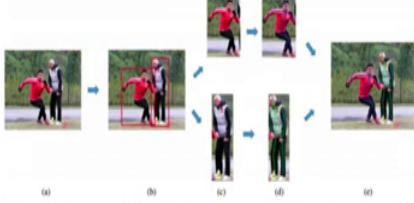


Fig. 8 An illustration of top-down pipeline. (a) Input image, (b) Two persons detected by human detector, (c) cropped single person image, (d) single person pose detection result, and (e) multi-person pose detection result.

Credit: Dang et al, Deep Learning based 2D Human Pose Estimation: A Survey, 2019

Vineth N B (IIT-H)
5.5 CNNs for Human Understanding: Pose and Crowd
7 / 19

Another approach, as we mentioned, is another category of methods, as we mentioned, is when there are multiple people whose poses you would like to estimate. Here, there are broadly two kinds of pipelines, the methods are very similar to what we saw earlier, but two pipelines which are different, one of them is known as the top down pipeline. In this particular case, we like to detect all persons, the poses of all persons in the given image. So we first of all start by detecting all people in the given image.

Then for each bounding box of the people detected, you run a single pose estimation approach that we just saw on the previous slides. You could help refine the pose estimates using some global context information if you like. So here is an illustration. You have an input image, two people detected using a human detector, using any other detection approach. Now you crop out these two people and run a single human pose estimator using a regression or a detection based approach. And you can get the skeleton, and similarly for the other person, and you then overlay both of these skeletons on the input image.

(Refer Slide Time: 15:22)

Multi-Person Pose Estimation: Bottom-Up Pipeline

- Procedure reversed from top-down
- All body parts (keypoints) are detected in first stage, then associated to human instances in second stage
- Inference stage likely to be faster - since no need to detect pose for each person separately




Fig. 10 Framework of bottom-up pipeline.





Fig. 11 An illustration of bottom-up pipeline. (a) Input image, (b) keypoints of all the persons, and (c) all detected keypoints are connected to form human instances.

Credit: Dang et al, Deep Learning based 2D Human Pose Estimation: A Survey, 2019

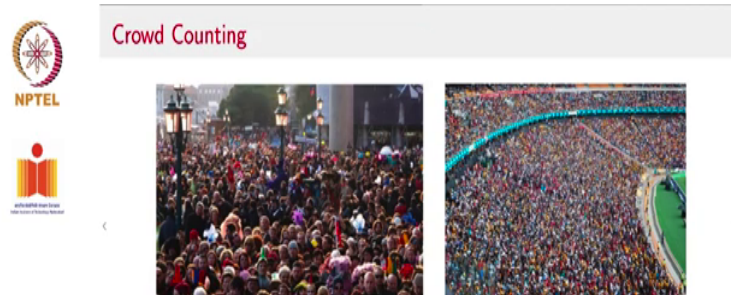
Vineeth N B (IIT-H) [7.5 CNNs for Human Understanding: Pose and Crowd] 8 / 19



On the other hand, you can also have a bottom up pipeline, where you reverse the process, where initially, you detect all the key points in the image, irrespective of whom it belongs to, so you just detect all the key points, so you could use a detection based approach, where you get a heat map for the full image. And the centers of all of these heat map regions could be different key points, you do not know which key point belongs to which person.

Once these key points are detected, then you associate these key points to human instances using different methods. And for more details of these methods, you can see this survey called Deep Learning based 2D human pose estimation. Evidently, you can see that in this approach, inference is likely to be much faster, because you are processing all people's information at the same time, rather than run each people's bounding box through different pipelines.

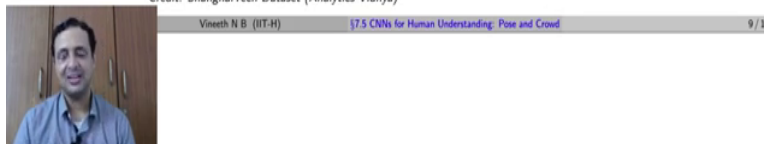
(Refer Slide Time: 16:21)



Crowd Counting

Estimating crowd density in images a crucial task for urban planning, public safety and security

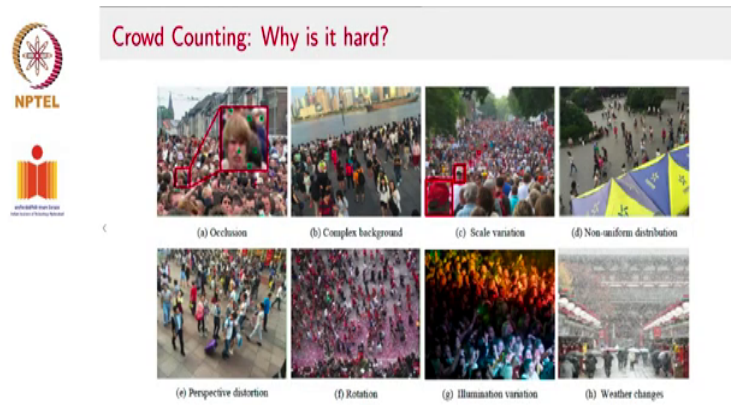
Credit: ShanghaiTech Dataset (Analytics Vidhya)



Vineth N B (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 9 / 19

Another task that we mentioned that we will look at is the task of crowd counting. Crowd counting is an extremely important task for urban planning, public safety, security, governance, so on and so forth. However, this can be a very challenging task in practice.

(Refer Slide Time: 16:43)

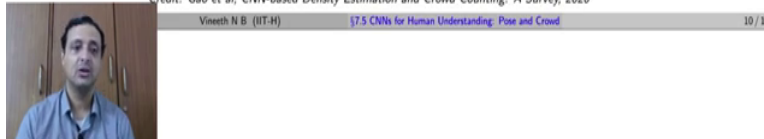


Crowd Counting: Why is it hard?

(a) Occlusion (b) Complex background (c) Scale variation (d) Non-uniform distribution

(e) Perspective distortion (f) Rotation (g) Illumination variation (h) Weather changes

Credit: Gao et al, CNN-based Density Estimation and Crowd Counting: A Survey, 2020



Vineth N B (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 10 / 19

You could face several challenges such as occlusion, as you can see here, in a single patch in this image, there are so many different faces denoted by green dots, each of which are occluded heavily with respect to the other, you could have a very complex background, you could have scale variations, depending on what perspective the camera took the picture from, you could have a non-uniform distribution of people across the image, you could have perspective distortions, you could have rotation issues, you could have illumination variance

variations, such as a show where you may want to count the number of people or you could be faced with weather changes. So all of these make this problem of crowd counting extremely hard.

(Refer Slide Time: 17:40)

CNNs for Crowd Counting

- **Basic CNN:** Basic CNN layers with no additional feature information
- **Multi-column:** Usually adopt different columns to capture multi-scale information corresponding to different receptive fields
- **Single-column:** Usually deploy single and deeper CNNs; premise to not increase complexity of network

Credit: Gao et al. CNN-based Density Estimation and Crowd Counting: A Survey, 2020



Vineth N B (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 11 / 19

Existing methods in using CNNs for crowd counting can be categorized into three different kinds of methods, one that use a basic CNN architecture to achieve the purpose, as we will see soon, where you have an input, you have a simple CNN, and you get a density map as the output of the model itself.

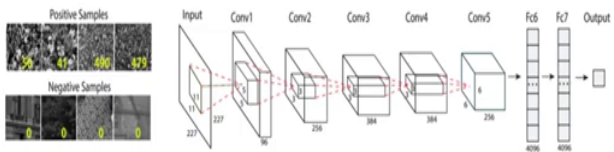
And the peaks in the heat map or the density map can give you an estimate of the count of the people that are in a given picture. Another approach is a multi-column approach, which hypothesizes that to be able to count crowds, when you have people in different scales, people faces in different scales, a big face, a small face, so on and so forth. You need to counter this with a multi column approach where each column looks at faces in a different scale.

So you can see that here, you are given an input, you now convert it to a multi-scale pyramid kind of an approach, where in each of these individual pipelines you are trying to detect faces of a certain scale. A third approach is a single column approach again, but this approach tries to observe performance of multi column approaches, and simplify them to an extent to ensure that the network architecture is not too complex. Let us see each of these approaches in more detail.

(Refer Slide Time: 19:09)




Basic CNN Approach⁴



- One of first efforts to use CNNs for direct regression; based on AlexNet architecture for dense crowd counting
- Expanded set of negative samples, whose ground truth counts are zeros, used to reduce interference
- Sensitive to density, distribution of crowd and scale of people

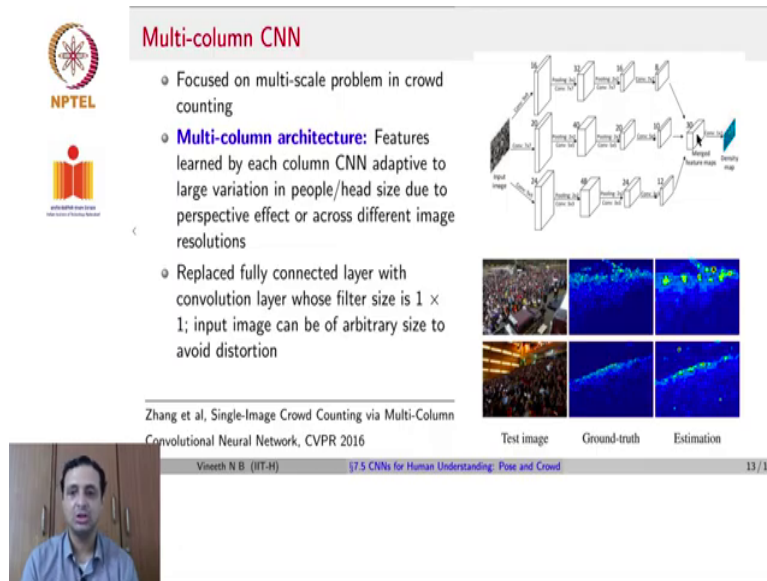
⁴Wang et al, Deep People Counting in Extremely Dense Crowds, ACM MM 2015
Vineeth N B. (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 12 / 19



So the basic CNN approach looks at crowd counting problem as a regression problem, where given an image, you have to predict a number as the output. So here are some training data points where given a set of crowd positive examples, and the counts in each of these images. Similarly, negative examples of other scenes where the crowd count is 0, each of these training samples are fed through a CNN architecture and the output is a count of the people in that image, which can directly be solved through regression and say L2 loss or a smooth L1 loss.

So you could here have an expanded set of negative samples because it is perhaps easier to get images without people whose ground truth counts as zeros, and this helps reduce interference and get better performance across the positive samples. However, you can make out that this is a crude approach, and it can be sensitive to density, distribution of crowd, scale of people, so on and so forth.

(Refer Slide Time: 20:26)



Multi-column CNN

- Focused on multi-scale problem in crowd counting
- **Multi-column architecture:** Features learned by each column CNN adaptive to large variation in people/head size due to perspective effect or across different image resolutions
- Replaced fully connected layer with convolution layer whose filter size is 1×1 ; input image can be of arbitrary size to avoid distortion

Zhang et al, Single-Image Crowd Counting via Multi-Column Convolutional Neural Network, CVPR 2016

Test image Ground-truth Estimation

Vineeth N.B. (IIT-H) 57.5 CNNs for Human Understanding: Pose and Crowd 13/19

The slide features the NPTEL logo, a diagram of the multi-column CNN architecture showing parallel processing at different scales, and a 2x3 grid of images comparing test images, ground truth density maps, and estimated density maps.

That leads us to multi column based approaches, which try to address the fact that people's faces could be at different scales in an image. So, in such a multi column CNN, each individual pipeline looks at an input image at different scales. You can see here, a conv 9×9 filter, a conv 7×7 filter and a conv 5×5 filter which looks at multiple scales in the input and this helps get a better performance towards the end, where after the last layers of each of these pipelines, the feature maps are merged and then you have a conv 1×1 layer to get a density map on the same resolution as input.

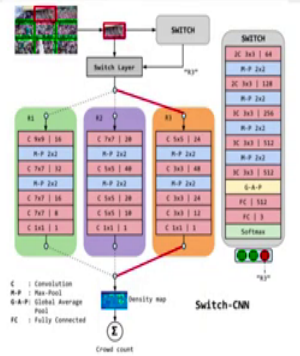
So, here are a few illustrations, given test image and the corresponding ground truth, you can see the estimated heat map here whose peaks you can consider to be able to get an estimate of the count if required. In certain applications, a density map by itself may actually serve the purpose but if a count is required, one could infer the count using these heat maps.

(Refer Slide Time: 21:45)



Multi-column CNN⁵

- Trains several independent CNN crowd density regressors on image patches (each regressor same as previous method of Zhang et al)
- Switch classifier** trained alternatively on regressions to select best one for density estimation \Rightarrow offers ability to model large-scale variations and leverage local variations in density in crowd scene
- Weighted averaging used to fuse features is global in nature



⁵Sam et al, Switching Convolutional Neural Network for Crowd Counting, CVPR 2017



Vineeth N.B. (IIT-H)

7.5 CNNs for Human Understanding: Pose and Crowd

14 / 19

A further approach of a multi column CNN expanded on the approach on the previous slide, and introduced the concept of a switch classifier, this work was done in 2017, where each patch of an input image was given to a switch layer, which decided which of these resolutions was the right scale for this particular patch.

So, the switch classifier, which you can see in the rightmost region here, took the patch of the image and then gave an output to see among these three scales which you had as individual columns in your CNN in the multi column CNN, it now says that for this particular patch R 3 is the right scale to detect faces in and now that patch is given to this CNN and the density map at that scale is used to get the final outcome. Why does this make sense?

This comes from the fact that within a local region of an image, it is likely that scales of faces are going to be maintained within the same range, whereas in another patch of the image there could be a scale that is very different. So, this uses that locality of scale in crowds to switch the corresponding CNN pipeline for each region and thereby achieve a good performance towards the end. And at the end, it uses weighted averaging to fuse the features, which can be used globally.

(Refer Slide Time: 23:21)

Single-column CNN⁶

- Based on observation that using a single column from multi-column networks retained 70% of accuracy on some datasets; hence, used a single-column CNN with single filter size as backbone
- By combining feature maps of multiple layers, could adapt network to variations in pedestrian (head) scale and perspective
- Used deconv layer to adapt network output instead of upsampling/elementwise summation

⁶Zhang et al, Crowd Counting via Scale-Adaptive Convolutional Neural Network, WACV 2018
Vineeth N B. (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 15 / 19



Single column CNNs are derived from multi column CNNs by making some observations from their performance, one of the first efforts here in 2018 observed that a single column in a multi column CNN for crowd counting retained about 70 percent of the accuracy on certain data sets. So, why make the architecture complex. So this single column CNN uses a standard set of convolutional layers initially, and then on passes feature maps from earlier layers to later layers, which are obtained after deconvolution.

So you can see here that after conv 6, the feature map from the previous layer is concatenated and then you do deconvolution, which is the equivalent of up sampling here to get a higher resolution image, then a feature map from an earlier layer is added to this up sampled image to get a new feature map and these are passed through certain set of convolutions and finally, a 1×1 convolution to get the final density map.

In this particular case, deconvolution was used instead of up sampling or element wise summation. Also, this work used both density map based loss as well as a count based loss to train the neural network. So all these approaches assume that the ground truth density map is given as well as the headcount is given.

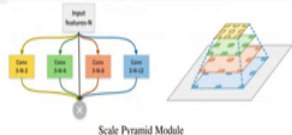
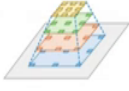
And so you can use the loss corresponding to both of these to back propagate and train the rest of the neural network. So what loss do you use? An L 2 loss on the density map. And if it is a count, you can just use an L 1 loss on the count, which is the absolute value or the difference between your predicted count and the correct count.

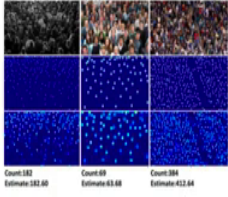
(Refer Slide Time: 25:21)


Another Single-column CNN⁷

- Observed that low-level features from same depth of different columns in multi-column CNNs are similar
- Employ a single-column structure as shared backbone and extract multi-scale features from high-level features in high layers
- Use dilated convolutions which can obtain different receptive fields at different rates; this Scale Pyramid Module placed between Conv4.3 and Conv5.1 of VGG16





⁷Chen et al, Scale Pyramid Network for Crowd Counting, WACV 2019
Vineeth N.B. (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 16 / 19



Another single column CNN, a more recent one in 2019 observed that low level features from multi column CNNs had very similar functions, very similar features in the crowd counting context. So what they propose is to retain the same pipeline for initial layers of the CNN. And when you go to the later layers, have a multi scale pyramid, which is known as a scale pyramid module, which combines features at different scales to get the final output of the density map.

So in this particular case, the scale pyramid module was implemented using dilated convolution at different scales to be able to analyze faces at different scales. And this was placed between conv 4_3 and conv 5_1 of a VGG 16 architecture. So you can see here a few examples, where given an image, the estimated count and the correct count. And you can see that in most of these cases, it is fairly close to a certain error tolerance to the ground truth estimate.

(Refer Slide Time: 26:40)




Homework

Readings

- [A detailed blog post on human pose estimation by Nanonets](#)
- [Gao et al, CNN-based Density Estimation and Crowd Counting: A Survey, 2020](#)
- (Optional) [Dang et al, Deep Learning Based 2D Human Pose Estimation: A Survey, 2019](#)

Exercise

CNNs for human understanding can especially suffer from biases in datasets (towards a particular race, ethnic background or gender); how do you find if a model is biased?






Vineeth N B (IIT-H) | 5.7.5 CNNs for Human Understanding: Pose and Crowd | 17 / 19

So, the homework for this lecture is a very nice blog post on human pose estimation by Nanonets, as well as the survey on density estimation and crowd counting that was recently released in 2020. If you are interested, you can also read this survey on 2D human pose estimation.

The exercise for this lecture, or a thought experiment for this lecture is CNNs, when used especially for human understanding, can suffer from biases in datasets. So depending on which race dominates the kind of people in a particular data set for human pose estimation or face recognition, the decisions could be biased by those statistics in a data set. Before we talk about addressing those biases, how do you first of all find if a model that you trained for a human based task is biased? Think about it.




(Refer Slide Time: 27:39)



References I

- [1] Alexander Toshev and Christian Szegedy. "DeepPose: Human Pose Estimation via Deep Neural Networks". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 1653–1660.
- [2] Amy L. Bearman, Stanford, and Catherine Dong. "Human Pose Estimation and Activity Classification Using Convolutional Neural Networks". In: 2015.
- [3] Jonathan Tompson et al. "Efficient object localization using Convolutional Networks". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 648–656.
- [4] Chuan Wang et al. "Deep People Counting in Extremely Dense Crowds". In: Oct. 2015, pp. 1299–1302.
- [5] João Carreira et al. "Human Pose Estimation with Iterative Error Feedback". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 4733–4742.
- [6] Yingying Zhang et al. "Single-Image Crowd Counting via Multi-Column Convolutional Neural Network". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 589–597.
- [7] Deepak Sam, Shiv Surya, and R. Babu. "Switching Convolutional Neural Network for Crowd Counting". In: July 2017, pp. 4031–4039.

Vineeth N.B. (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 18 / 19



References II

- [8] Lu Zhang, Miaojing Shi, and Qiaobo Chen. "Crowd Counting via Scale-Adaptive Convolutional Neural Network". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 1113–1121.
- [9] Xinya Chen et al. "Scale Pyramid Network for Crowd Counting". In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2019), pp. 1941–1950.
- [10] Q. Dang et al. "Deep learning based 2D human pose estimation: A survey". In: *Tsinghua Science and Technology* 24.6 (2019), pp. 663–676.
- [11] Guangshuai Gao et al. "CNN-based Density Estimation and Crowd Counting: A Survey". In: *ArXiv abs/2003.12783* (2020).

Vineeth N.B. (IIT-H) | 7.5 CNNs for Human Understanding: Pose and Crowd | 19 / 19

Here are some references to conclude.