Requirements

>= Python 3.6
>= Pytorch1.3.1
Download the pretrained model from Link(Password: 585s)

Method Introduction

Our method of areal crowd counting combines the model AS-DA-Net by Jiang et al. [1] with a heuristic subarea partition approach.

1. Crowd density map estimation

Jiang et al. [1] proposed two complementary CNNs(attention scaling network, ASNet; density attention network, DANet) to predict a crowd density map given an image. We choose this model for its superiority in tackling scene with crowds of diverse density located in different areas. The overall structure of the model is demonstrated in Figure 1.

ASNet consists of two branches. The first branch outputs intermediate density maps of different density levels. The second branch outputs scaling facors which are then used to scale the corresponding density maps.

DANet is responsible for getting the attention masks for areas of different density levels. It performs a pixel-wise classification task, where each class label represents a density level. Multiplying the attention masks, intermediate density maps and scaling factors gives several adjusted density maps, sum of which is the predicted density map.

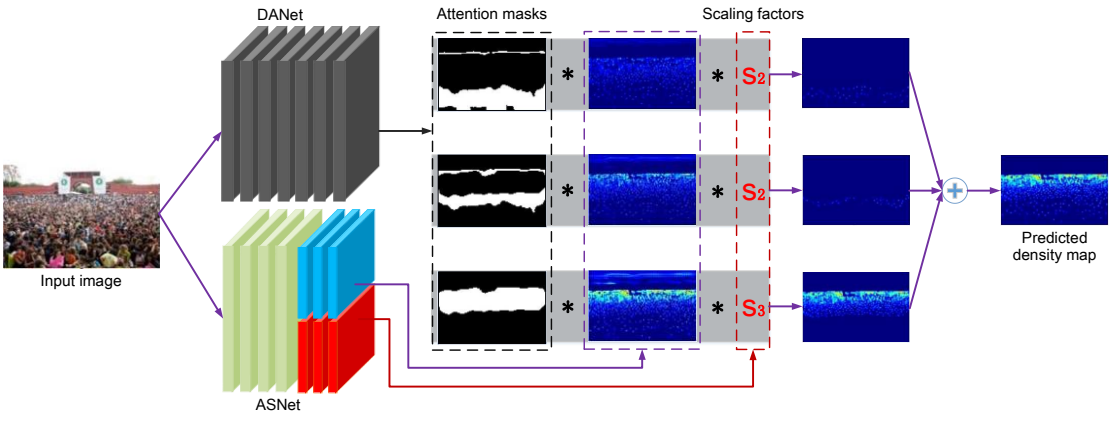

Figure 1. The architecture of AS-DA-Net proposed by Jiang et al. [1]. The number of density levels is a hyperparameter and is set as 3 here.

2. Areal crowd counting

We propose a heuristic subarea partition approach to get areal count. As is illustrated in Figure 2, we multiply the predicted density map with subarea masks for heads located in different areas, which gives subarea density maps of target areas. And the sum of each areal density map is the final count of corresponding area. To get the

subarea masks, we demarcate boundaries between areas in a typical image by the camera and then label each pixel by checking its location relative to the boundaries.

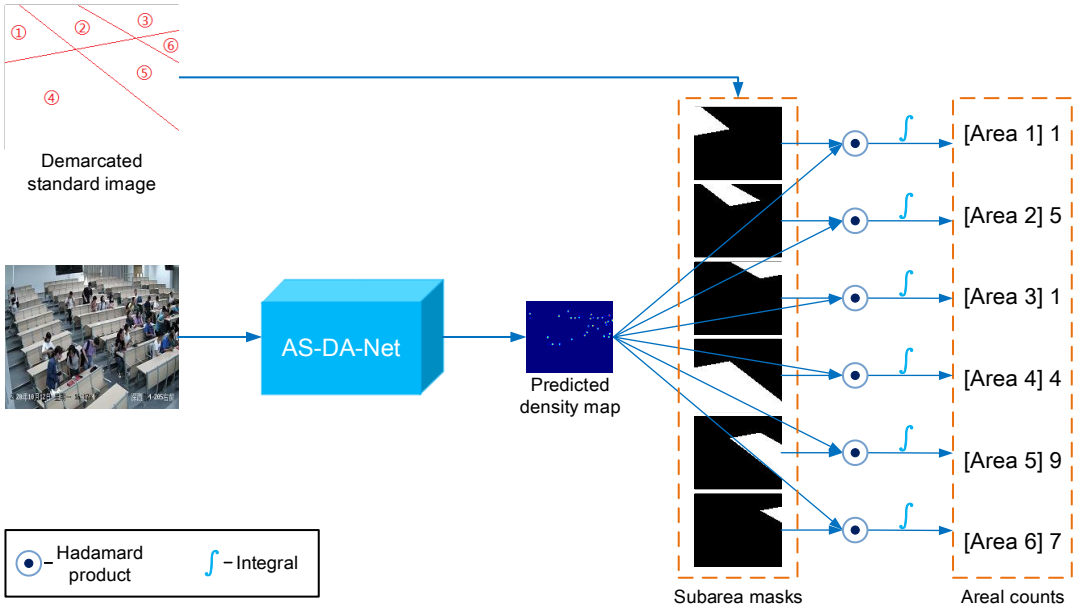

Figure 2. Overall pipeline of our areal crowd counting approach. For a fixed camera, we choose a standard image by it and demarcated boundries between areas manually. The subarea masks are then generated by by checking each pixel's location relative to the boundaries. Multiplying the estimated density map with subarea masks give subarea density maps. The integral of each subarea density map is the final count of the corresponding area.

The intuition behind this method is that heads of crowds generally locate around a plane parallel to the floor. And the projection of the plane to an image does not change the topology of the plane.

Though the head of a person located in one area may by chance appear in a nearby area in the image for reasons as abnormal stature or abnormal posture, it does not affect the expectation of areal counts statistically.

2.1. Areal boundaries demarcation

To get the boundaries between areas, we select a typical image of the scene shooting by the fixed camera. A typical image is recommended to have people of medium height standing near every border between every two areas. Such image doesn't necessarily exist, in which case we may take many images into consideration. For example, we can select one image for each boundary.

We then draw the boundaries between areas manually, as showed in Figure 3. The boundaries consist of two sets of dividing lines, one for horizontal division, another for vertical division. The coordinates of endpoints of horizontally-dividing line i H_i can be easily obtained with the help of image processing softwares like Microsoft's mspaint, from which we can calculate the equation of H_i

$$y = a_i x + b_i$$

, where a_i is the slope of H_i and b_i is the intercept of H_i. Note that the a_i here

may not exist, for a horizontally-dividing line may be parallel to y-axis. For convenience, we denote the case that a_i does not exist as $a_i = \infty$ and $x = b_i$, where b_i is no longer the intercept but x coordinate of every point on H_i.

Similarly, for vertically-dividing line j V_j, we have

$$y = a'_j x + b'_j.$$

We assume no vertically-dividing line is parallel to y-axis, so a'_j always exists.

Figure 3. An example of demarcated standard image.

2.2. Getting subarea masks

Given the two sets of dividing lines, we can calculate subarea mask for each area using a simple algorithm illustrated in Algorithm 1. Specifically, a subarea mask is a boolean matrix with the size of standard image. In a subarea mask, elements belong to corresponding area are assigned as 1, otherwise, 0.

All elements of all subarea masks are initialized to be 0. Then, for every pixel in the standard image, we check its location relative to the dividing lines to decide its area, and assign 1 to element sharing same x, y coordinates on the corresponding subarea mask.

Require: w, the width of standard image
Require: h, the height of standard image
Require: n, the number of horizontal splits
Require: m, the number of vertical splits
Require: $a_i,\ i \in \{1, \dots, n-1\}$, the array of slopes of horizontally-dividing lines
Require: $b_i,\ i \in \{1, \dots, n-1\}$, the array of intercepts of horizontally-dividing lines
Require: $a'_j,\ j \in \{1, \dots, m-1\}$, the array of slopes of vertically-dividing lines
Require: $b'_j,\ j \in \{1, \dots, m-1\}$, the array of intercepts of vertically-dividing lines
Initialize: $M_i = \mathbf{0}_{w \times h},\ i \in \{1, \dots, n \times m\}$, subarea masks

```
for x = 0, 1, ..., w − 1 do
    for y = 0, 1, ..., h − 1 do
        col = n
        row = m

        for i = 1, ..., n − 1 do
            if (a_i = ∞ and x ≤ b_i) or (a_i ≠ ∞ and x + (b_i − y)/a_i ≤ 0)
                col = i
                break

        for j = 1, ..., m − 1 do
            if y − a'_j x + b'_j ≤ 0
                row = j
                break

        area = (row − 1) × m + col
        M_area[x][y] = 1
```

Algorithm 1.

2.3. Getting areal counts

To get final areal counts, we first caculate subarea density map for each area, which is the Hadamard product of the predicted density map and corresponding subarea mask. For area i, we have

$$D_i = D_{whole} \odot M_i$$

, where D_i is subarea density map i and D_{whole} is the predicted density map output by AS-DA-Net.

Integrating subarea density map i gives us the final count of area i

$$Count_i = \sum_x \sum_y D_i[x][y].$$

Reference

[1] Xiaoheng Jiang, Li Zhang, Mingliang Xu, Tianzhu Zhang, Pei Lv, Bing Zhou, Xin Yang, Yanwei Pang. Attention scaling for crowd counting. In Proceedings of the IEEE International Conference on Computer Vision, pages 4706–4715, 2020.

Pretrained Model

The Pretrained model is borrowed from https://github.com/laridzhang/ASNet**.** Link, Password: 585s

License

The majority of the code is borrowed from https://github.com/laridzhang/ASNet. Any commercial use of the code should get approval from the original author first. Feel free to use it for study, but do cite the original author's paper.