Context-Aware Crowd Counting

Tushaar Ranganathan

October 2024

https://github.com/Tushaar-R/Context-Aware-Crowd-Counting

1 Model Architecture For Feature Extraction

1.1 VGG-19

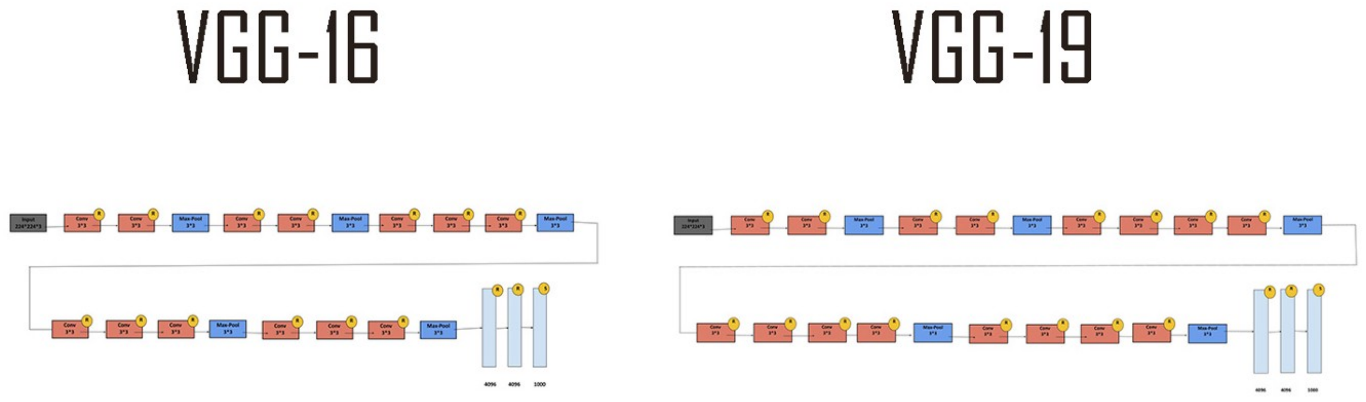

Figure 1: VGG-16 vs VGG-19

The current model in the research paper uses the first 10 layers of VGG-16 for feature extraction. We can choose to use the first 12 layers of VGG-19 for feature extraction instead. The added layers in VGG-19 allow the model to learn more complex hierarchical features. The convolution operation can be expressed mathematically as:

$$(f * g)(x, y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m, n)g(x - m, y - n)$$

Here, f represents the input image, and g represents the filter/kernel. More convolutional layers mean more feature extraction capabilities, allowing VGG-19 to capture more intricate patterns. Better features lead to improved predictions and lower loss.
On practical implementation(for 10 epochs) VGG-19 takes more time per epoch however it converges much faster than VGG-16. Since we use pre-trained weights of both models, with VGG-19 giving a test loss of 159 and VGG-16 giving a test loss of 273 after 10 epochs

1.2 EfficientNet

- EfficientNet was first released in 2019 and outperforms VGG-16 architecture in feature extraction.
- It uses compound scaling method that uniformly scales the network's depth, width, and input resolution, which will better fit with the objective of multi scaled feature fusion.
- The scaling factors for EfficientNet can be represented as:

$$\text{Depth} = \alpha^d, \quad \text{Width} = \beta^w, \quad \text{Resolution} = \phi^r$$

  where:

  - α, β, and ϕ are constants.
  - d, w, and r represent the depth, width, and resolution scaling factors, respectively.

- EfficientNet employs **depthwise separable convolutions**, which factor the standard convolution into two layers:
  - **Depthwise Convolution:** Applies a single filter per input channel:

$$(f * g_d)(x, y) = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} f(m, n)\, g_d(x - m, y - n)$$

  - **Pointwise Convolution:** Combines the output of the depthwise convolution using 1×1 convolutions:

$$(h * g_p)(x, y) = \sum_{c=0}^{C-1} h(x, y, c)\, g_p(x, y, c)$$

  Here:

  - f represents the input,
  - g_d represents the depthwise filter,
  - h represents the output from the depthwise convolution,
  - g_p represents the pointwise filter,
  - C is the number of input channels, and k is the kernel size.

- It contains Squeeze-and-Excitation layers which perform the following operations:
  - **Squeeze:** Global average pooling is used to generate channel descriptors:

$$z_c = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_{i,j,c}$$

  - **Excitation:** Using a learned function to produce channel-wise weights:

$$s = \sigma(W_2 \cdot \delta(W_1 \cdot z))$$

    where:

    * σ is the sigmoid activation function,
    * W_1 and W_2 are weights of the fully connected layers,
    * δ represents a non-linear activation function (often ReLU or similar).

- It gives a higher accuracy compared to VGG-16 and also uses lower number of parameters.

2 Divide and Conquer Strategy

2.1 Partition+Density Correction

This idea stems from the fact that ground truth density maps are continuous.

Let us assume that through the workings of another ML model, we get an indicator through which we can recognize whether the count of people in some specified portion of our estimated density maps overshoot or undershoots the actual count of people present in that portion.

During testing, the test images will be partitioned into smaller, equally sized sections. If an image has dimensions $l \times b$, it will be divided into 4 sub-images, each with dimensions $\frac{l}{2} \times \frac{b}{2}$. . For each of the 4 sub-images, we will create a new image for which the rest of the portion will be blank. We will only concern ourselves with the results or ground truth densities obtained in the estimated ground truth map in the region of the respective partition.

By performing analysis on the training data, if we strongly feel that the estimated count of people in the current partition will be greater than the actual count of people in the same partition, then we will keep the surrounding image blank.

If we strongly feel that the estimated count of people in the current partition will be less than the actual count of people in the same partition, then we will manually add faces for the model to detect along the borders of the respective partition.

2.2 Mathematical Proof

Let us assume that estimated count of people in partition¿actual count of people in the partition. Due to the property that the ground truth density map is continuous, the final estimated ground truth density map for the partition region will be impacted by the lack of presence of crowd in the surrounding region.This will create the illusion of sparsity along the borders of the estimated density map for the model and will bring down the estimated count of people in the given partition.

Now let us assume that the estimated count of people in partition¡actual count of people in the partition. Due to the property that the ground truth density map is continuous, the final estimated ground truth density map for the partition region will be impacted by the additional presence of crowd(which was not previously present). This will create the illusion of a dense crowd along the borders of the estimated density map for the model and will bring up the estimated count of people in the given partition.

Thus in both cases, we will reduce the total absolute error between actual count of people in a partition and total count of people in a partition, which will result in better estimation and reduction in MAE for the entire test set. Breaking test images into further partition may yield better results but at the cost of higher computation time.