BA865

Crowd Counting

Keshuo Liu, Qianru Ai, Zheming Xu

Table of Content

Background

Smart City

To improve the efficiency of city governance, regarding public transportation, utility services, parking and vehicles, public art, etc.

Go Boston 2030

To expand access, improve safety, and ensure liability.

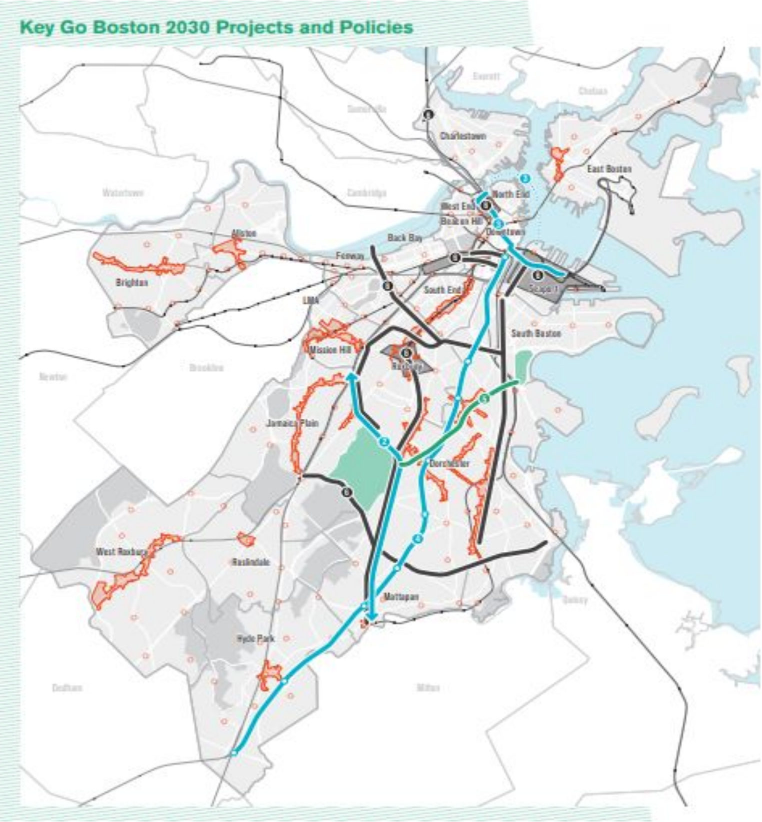
Top Projects

- Walking and Bicycle Friendly Main Street Districts
- Mattapan to LMA Rapid Bus
- North Station to South Boston Waterfront Rapid Bus and Ferry
- Fairmount Indigo Line Service Improvements and Urban Rail
- Columbia Road Greenway
- Smart Signal Corridors and Districts
- Neighborhood Mobility microHUBS

Top Policies

- State of Good Repair–Particularly Bridges
- Restructure All Bus Routes
- Autonomous Vehicles
- Vision Zero Safety Initiatives (Corridors, Crossings, Slow Streets)

Action Plan Highlights



Motivation

- To build deep learning algorithms about crowd counting
    - a fundamental algorithm for broad application

- To count people in different images.
    - automated public monitoring such as surveillance and traffic control
    - Under the current pandemic situation, crowd counting is a more heating topic because it could be used to monitor the risk of disease transmission
    - more complex since each person has different features

Dataset

Train + Validation:

Original Source: http://personal.ie.cuhk.edu.hk/~ccloy/downloads_mall_dataset.html

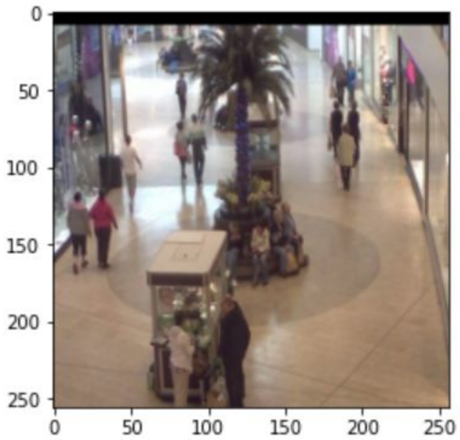
- 2000 RGB images of frames in a video (as inputs) & the number of pedestrians in the image (as target variable)
- 640x480 pixels at 3 channels
- The label of this dataset is the 'count' of people in each frame which is an existing feature in the dataset.

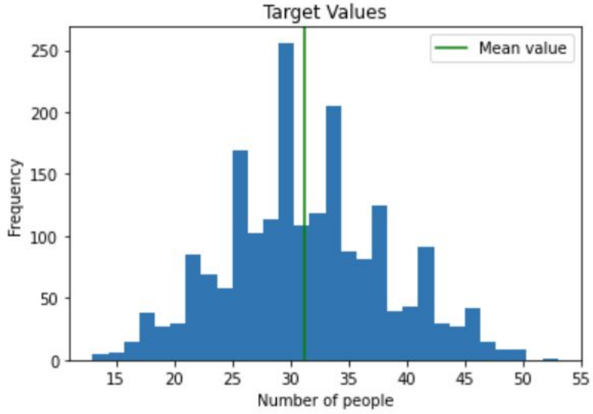
Test (34): Public Transportation

We took 34 pictures of the crowd waiting for the green line or traffic lights for testing. The label of this dataset is also 'count'.

Pre-processing

- Reshape images to 256x256 pixels
- Augmentation
- Target values (people count) vary between 13 and 53 with a mean of 31.16. Values are normally distributed with the median value close to the mean.





Self-trained Model

- Residual Block
  (Conv+ Relu Activation)
- Dense
  (128 → 8)
- Loss function: MAE
- Optimizer: Adam

<table>
<tr><td colspan="2">Input 3@256×256</td></tr>
<tr><td colspan="2">Data Augmentation 3@256×256</td></tr>
<tr><td colspan="2">Conv 3@256×256</td></tr>
<tr><td>Conv 32@254×254</td><td rowspan="3">Residual (Conv 32@254×254)</td></tr>
<tr><td>Conv 32@254×254</td></tr>
<tr><td>MaxPooling 32@254×254</td></tr>
<tr><td colspan="2">Add 32@85×85</td></tr>
<tr><td>Conv 32@85×85</td><td rowspan="3">Residual (Conv 32@85×85)</td></tr>
<tr><td>Conv 64@85×85</td></tr>
<tr><td>MaxPooling 64@85×85</td></tr>
<tr><td colspan="2">Add 64@29×29</td></tr>
<tr><td>Conv 64@29×29</td><td rowspan="2">Residual (Conv 64@29×29)</td></tr>
<tr><td>Conv 128@29×29</td></tr>
<tr><td colspan="2">Add 128@29×29</td></tr>
<tr><td colspan="2">GlobalMaxPooling 128@29×29</td></tr>
<tr><td colspan="2">Dropout 128</td></tr>
<tr><td colspan="2">Dense 128→8</td></tr>
<tr><td colspan="2">Output 1</td></tr>
</table>

Self-trained Model:
Hyper-parameters

kernal_size	[3,5]
strides	[2,3]
units	[16,32]
batch_size	[10,20]
epochs	[20,30]

GridSearchCV

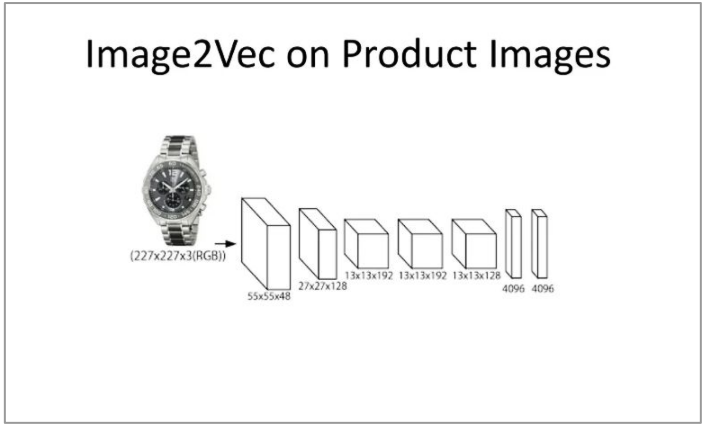
```
scoring='neg_mean_absolute_error'

cv=5


Best Score  : -5.6352673482894895
```

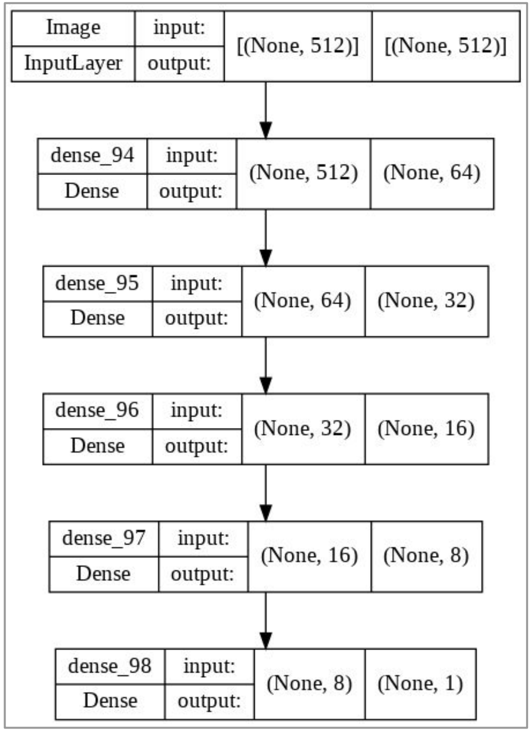
Pre-trained Models:

Img2Vec: Image Embedding



Default Resnet-18 → vector length: 512

+



Pre-trained Models:

MCNN: Multi-Column CNN

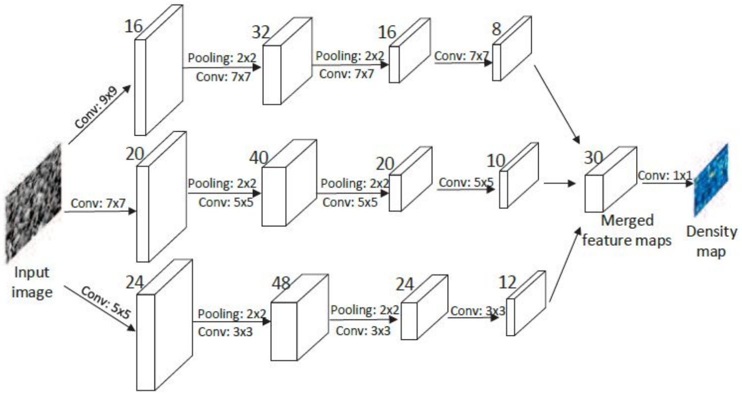
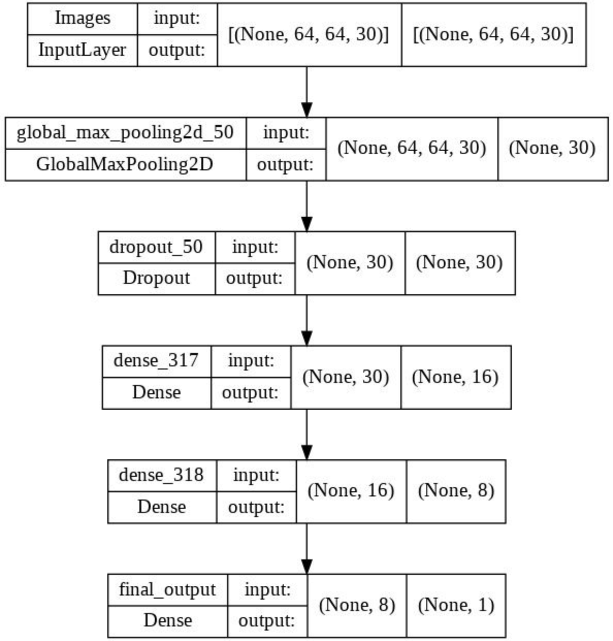


Figure 3: The structure of the proposed multi-column convolutional neural network for crowd density map estimation.

+

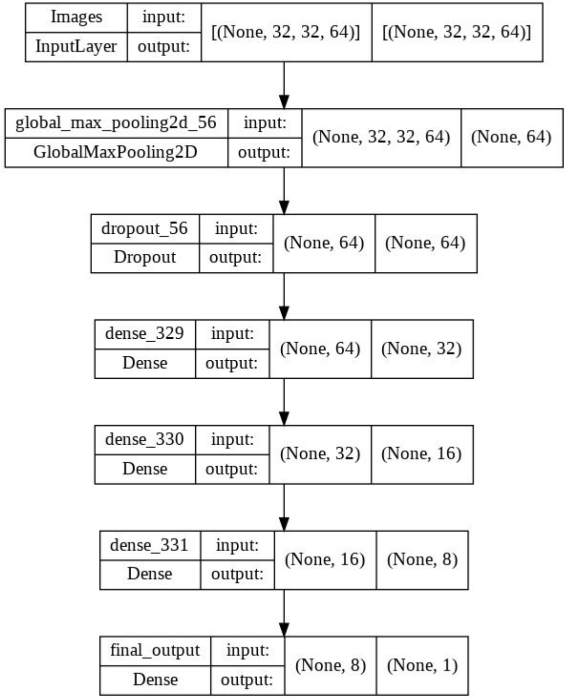


Pre-trained Models:

CSRNet: CNN2D frond-end + Dilated CNN back-end

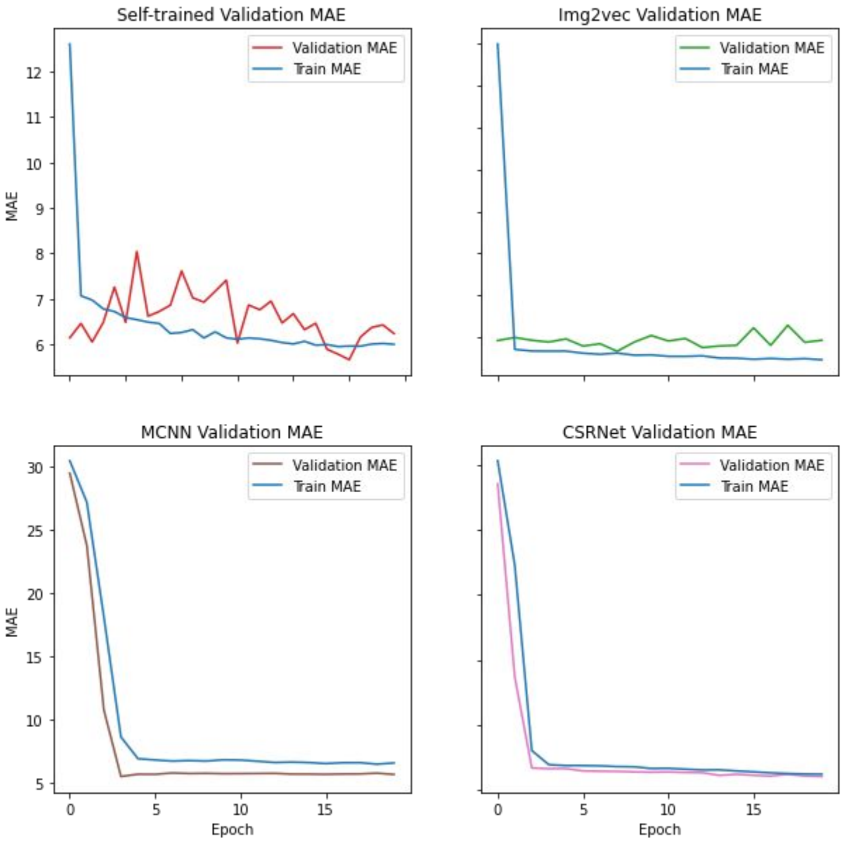
Configurations of CSRNet			
A	B	C	D
input(unfixed-resolution color image)			
front-end (fine-tuned from VGG-16)			
conv3-64-1 conv3-64-1			
max-pooling			
conv3-128-1 conv3-128-1			
max-pooling			
conv3-256-1 conv3-256-1 conv3-256-1			
max-pooling			
conv3-512-1 conv3-512-1 conv3-512-1			
back-end (four different configurations)			
conv3-512-1 conv3-512-1 conv3-512-1 conv3-256-1 conv3-128-1 conv3-64-1	conv3-512-2 conv3-512-2 conv3-512-2 conv3-256-2 conv3-128-2 conv3-64-2	conv3-512-2 conv3-512-2 conv3-512-2 conv3-256-4 conv3-128-4 conv3-64-4	conv3-512-4 conv3-512-4 conv3-512-4 conv3-256-4 conv3-128-4 conv3-64-4
conv1-1-1			

+

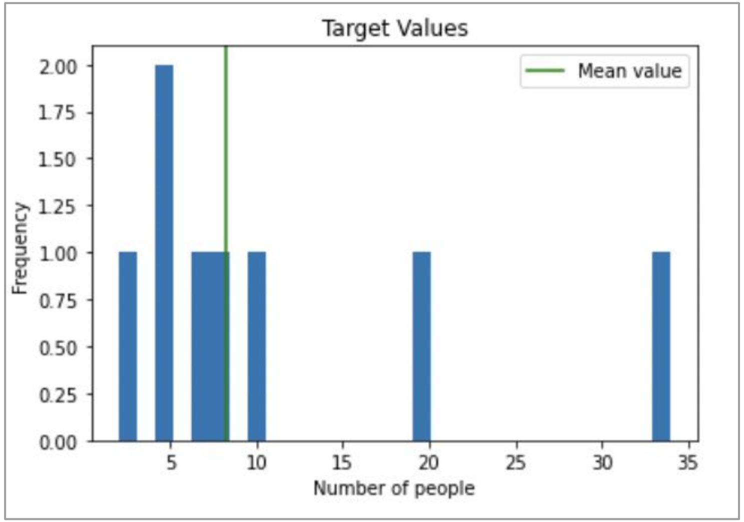


Comparison: 5-CV MAE

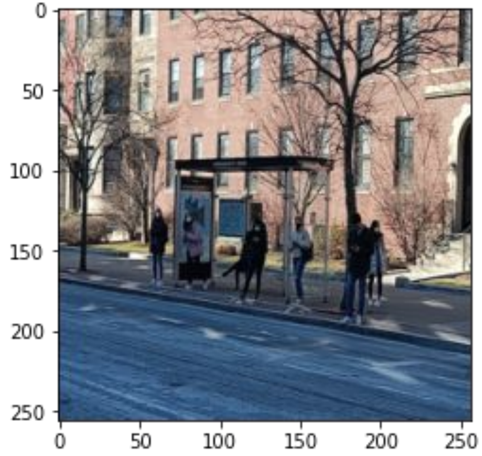
Model	Avg Validation MAE
Self_trained	6.240
Pre_trained_Img2vec	5.936
Pre_trianed_MCNN	5.654
Pre_trained_CSRNet	6.052

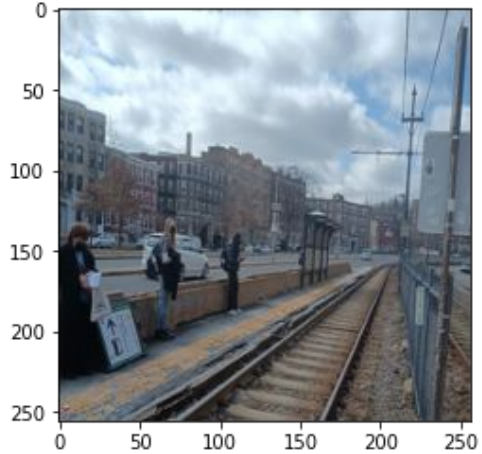


Test Dataset



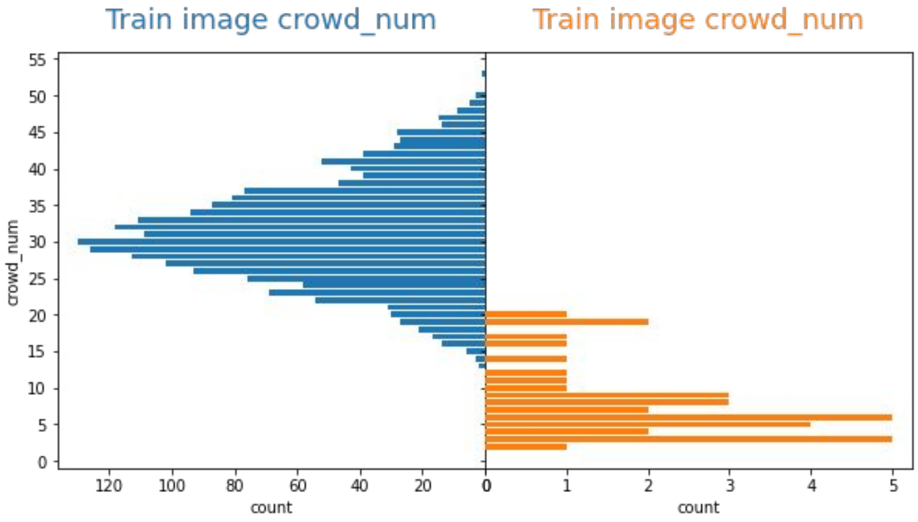
	count	mean	std	min	25%	50%	75%	max
crowd_num	34.0	8.176471	5.07203	2.0	5.0	6.5	9.75	20.0





Test Dataset: Performance

- 34 images around Boston University
- Reshape images to 256x256 pixels
- Sparse distribution
- Target values (people count) vary between 2 and 20 with a mean of 8.18
- MAE = 23.97



Conclusion

- The MCNN pretrained model performed the best (lowest Validation MAE: 5.654)
- Model performance on the test dataset is bad
  - Different angles, backgrounds…
  - People in some of the images were so close to other people and objects

Next Steps

- Training models using images of different scenarios
- Try thermograms
  - Protect privacy
  - Not limited by different backgrounds

Reference

Mall Dataset: http://personal.ie.cuhk.edu.hk/~ccloy/downloads_mall_dataset.html

Img2Vec: https://github.com/christiansafka/img2vec

MCNN: http://people.eecs.berkeley.edu/~yima/psfile/Single-Image-Crowd-Counting.pdf

https://github.com/svishwa/crowdcount-mcnn

CSRNet: https://arxiv.org/pdf/1802.10062.pdf

https://github.com/leeyeehoo/CSRNet-pytorch