DEEPVISION CROWD MONITOR

AI FOR DENSITY ESTIMATION AND OVERCROWDING DETECTION

Theory document-

Model training and evaluation for both parts A and B

Submitted by:

Arti Kumari

Table of Contents

1. Introduction

Crowd counting is a critical task in computer vision with applications in public safety, smart city monitoring, event management, and emergency response. Accurate estimation of people in a scene helps prevent overcrowding, improves resource allocation, and supports real-time decision-making. The task, however, is challenging due to high crowd density, heavy occlusion, perspective distortion, scale variation, and diverse environmental conditions.

Traditional detection-based approaches often fail in dense scenes because they attempt to locate each individual explicitly. Instead, modern methods use *density map regression*, where the model predicts a continuous density distribution whose integral corresponds to the crowd count. This approach is far more robust in complex, densely populated environments.

In this project, we implement and train **CSRNet (Congested Scene Recognition Network)**, a state-of-the-art crowd counting architecture. CSRNet combines a VGG-16 frontend for feature extraction with a dilated convolutional backend that increases the receptive field while preserving resolution, enabling the model to capture both fine details and large contextual regions.

The model is evaluated on the **ShanghaiTech dataset**, which contains two subsets:

- **Part A** – extremely dense web-sourced images with high occlusion and large-scale variation
- **Part B** – moderately crowded street scenes with fewer people and clearer visibility

The notebook includes preprocessing, density map generation, model training, inference, visualization, and quantitative evaluation using MAE and RMSE metrics. Although training was conducted on CPU with a small number of epochs for demonstration, the workflow is fully scalable for GPU training and real-world deployment.

2. Dataset Description

The **ShanghaiTech Crowd Counting Dataset** is a benchmark dataset widely used for evaluating density-estimation models in computer vision. It is divided into two subsets—Part A and Part B—each representing different crowd densities and environments.

Subset	Train Images	Test Images	Density Level
Part A	300	182	Very High (hundreds–thousands)
Part B	400	316	Low–Medium (tens–hundreds)

Part A — High Density Scenes

Part A images are collected from the Internet and include highly congested public gatherings with extreme occlusion and scale variations. Many images contain more than 1,000 individuals, making this subset ideal for testing robustness under complex visual conditions.

Part B — Medium Density Scenes

Part B contains street-level images captured in Shanghai under normal pedestrian flow. These scenes are less congested, with clearer visibility and moderate crowd sizes. This subset evaluates performance in everyday, real-world scenarios.

Ground-Truth Annotations and Density Maps

Each image has a corresponding **.mat file** providing manually labeled **(x, y)** head positions. These coordinates are converted into **density maps** by applying Gaussian kernels, ensuring that the **sum of pixel values equals the true crowd count**. Density maps allow the model to learn spatial distribution rather than explicit person detection, which is well suited for congested scenes.

3. Data Preprocessing Methodology

To prepare data for CSRNet, the notebook performs several steps:

3.1 Image Loading and Normalization

- Images are loaded in **RGB format**, scaled to `[0,1]` float32.
- If an image is too large, it is resized such that its **maximum side ≤ 512 px**.
- All images are normalized using **ImageNet mean and std**, since CSRNet uses VGG-16 weights pretrained on ImageNet.

3.2 Ground Truth Loading

Ground-truth `.mat` files contain head locations. These coordinates are scaled exactly according to the resized image.

3.3 Density Map Generation

Each point is converted into a Gaussian kernel with a fixed **sigma = 4**, creating a density representation.
Why Gaussian?

- Spreads each annotation over neighboring pixels
- Prevents the label from being too sparse
- Encourages smooth spatial learning

3.4 Padding

CSRNet requires dimensions divisible by **8**.
Images and density maps are padded to:

- Height = **768**
- Width = **1024**

Padding ensures:

- Efficient batch processing
- No shape mismatch in convolutions
- Consistent output resolution

3.5 Downsampling

Because CSRNet outputs density maps at **1/8 the resolution**, the ground-truth density map is downsampled by 8× using:

- Average pooling
- Sum correction to preserve total count

This ensures predictions and ground truth are at compatible scales.

4. Model Architecture: CSRNet

CSRNet consists of two major blocks:

4.1 Frontend — VGG-16 Convolutional Layers

The first 10 convolutional layers of VGG-16 extract low-level and mid-level features such as:

- edges
- textures
- object outlines
- body shapes

The max pooling layers gradually reduce spatial resolution.

4.2 Backend — Dilated Convolutional Layers

The backend replaces further pooling with **dilated (atrous) convolutions**, enabling:

- **Larger receptive fields**
- Ability to understand **large crowd regions**
- Preservation of spatial resolution

Architecturally:

- Six dilated convolution layers
- Followed by a final **1×1 convolution** producing a 1-channel density map

4.3 Parameter Count

The output:

CSRNet trainable parameters: 16263489

Meaning:

- The model has **16.26 million** trainable parameters
- Majority come from VGG-16 frontend
- This is relatively light compared to modern large models
- Allows training on CPU (although slow)

4.4 Tensor Size Check

Dummy input shape: (1, 3, 256, 256)

Dummy output shape: (1, 1, 32, 32)

This means:

- Input image is 256×256 RGB
- Output is a density map with **1/8 resolution**
- (256 / 8 = 32)

Thus, if a pixel in the output has value = 0.5, summing all predicted pixels gives the estimated crowd count.

5. Training Pipeline and Hyperparameters

5.1 Loss Function

The training uses **Mean Squared Error (MSE)**:

$$\mathcal{L} = \frac{1}{N}\sum(y_{\text{pred}} - y_{\text{gt}})^2$$

Why not MAE?

- MSE is smoother and penalizes large errors heavily.
- Suitable for dense regression tasks like density maps.

5.2 Optimizer

```
optimizer = Adam(lr=1e-5, weight_decay=5e-4)
```

- Adam handles noisy gradients well
- Low lr helps stability when fine-tuning pretrained VGG layers

5.3 Epoch Loop

Each epoch performs:

1. Forward Pass → predictions
2. Compute Loss → MSE
3. Backward Pass → gradient computation
4. Weight Update → model learning
5. Loss Logging → batch + epoch curves

The loss gradually decreases, confirming training stability.

6. Evaluation Methodology

6.1 Forward Inference

During evaluation:

- The model is set to `eval()` mode (turns off gradient computation)
- Each test image is passed through CSRNet
- Predicted density map is summed to obtain **predicted count**

6.2 Metrics Used

Two key metrics:

Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum | p_i - g_i |$$

Measures accuracy of counting.

Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\frac{1}{N}\sum(p_i - g_i)^2}$$

Measures stability and penalizes large deviations.

6.3 Evaluation Results

```
--- Evaluating Part A ---
MAE: 150.682 | RMSE: 214.796

--- Evaluating Part B ---
MAE: 33.509 | RMSE: 41.523
```

Meaning:

- On **Part A**, the model is less accurate due to extreme crowd density
- On **Part B**, the model performs significantly better

7. Results Interpretation

Dataset	MAE	RMSE	Interpretation
Part A	150.68	214.79	High error due to extreme density, occlusion
Part B	33.51	41.52	Good performance, manageable crowd sizes

8. Why Part A Performs Worse

Part A contains:

- Images with **hundreds to thousands** of people
- Severe occlusion
- Large perspective variations
- Complex dense clusters

CSRNet requires:

- More training epochs (≥ 300)
- GPU support
- Adaptive Gaussian kernels

The CPU-only training you performed (10 epochs) is insufficient to reach research-level accuracy.

9. Conclusion

This project successfully:

- Implemented a complete CSRNet-based crowd counting pipeline
- Preprocessed the ShanghaiTech dataset
- Trained CSRNet on Part A (demo-scale CPU training)
- Evaluated performance on both Part A & Part B
- Achieved reasonable results given limited training resources

Key insights:

- Part A requires much deeper training and GPU acceleration
- Preprocessing (especially density map generation and downsampling) is crucial
- CSRNet is highly effective for density estimation tasks

This document provides a thorough explanation of the methodology, training, evaluation, results, and insights.

10. References

- Zhang et al., **CSRNet: Dilated Convolutional Neural Networks for Understanding Highly Congested Scenes**, CVPR 2018
- ShanghaiTech Dataset
- PyTorch Documentation
- TorchVision Models
- NumPy, OpenCV, SciPy libraries