DEEPVISION CROWD MONITOR

AI FOR DENSITY ESTIMATION AND OVERCROWDING DETECTION

Submitted by:
Arti Kumari

Table of Contents

1. Project Description

Crowd monitoring is a computer vision task that aims to estimate the number of people present in an image. Traditional detection-based methods struggle in highly crowded or occluded scenes, leading to the need for more robust, scale-invariant alternatives. Density map–based approaches address this challenge by generating a pixel-wise density representation of the crowd, enabling the model to learn spatial distribution rather than relying on individual detection.

Importance of Density Map–Based Approaches

- They handle severe occlusions and cluttered environments effectively.
- They remain robust across varying scales, perspectives, and crowd densities.
- They enable more precise spatial learning, allowing the model to capture local crowd variations.

Goal of the Project

The primary objective of this project is to develop a preprocessing and exploration framework for crowd monitoring using density maps. The workflow includes:

- Understanding the structure and characteristics of the ShanghaiTech dataset
- Processing raw images and annotations
- Preparing normalized image tensors and downsampled density maps for model training
- Visualizing the spatial distribution of crowd regions

This serves as the foundation for building a complete deep learning model for crowd estimation.

2. Dataset Used

Dataset Name:

ShanghaiTech Crowd Counting Dataset

Introduction

The ShanghaiTech dataset is one of the most widely used benchmarks for crowd counting. It provides high-quality annotated images featuring crowds with a wide range of densities, scales, and illuminations.

Dataset Partitions

The dataset is divided into two subsets:

Part A

- Contains highly congested scenes
- Captured from the Internet
- Higher crowd density and more variation in scale
- Labels provided as point-level annotations indicating head locations

Part B

- Captured in urban streets of Shanghai
- Relatively sparse crowds
- More consistent lighting and perspective
- Annotation format identical to Part A

Image Characteristics

- RGB images with varied resolutions
- Scene diversity across crowd density, lighting, and background
- Ground truth annotations include:
    - **Point annotations (dot positions)**
    - **Density map labels** (generated using Gaussian kernels)

3. Environment Setup

Development Tools

- **Python** — primary programming language
- **Jupyter Notebook** — environment for interactive experimentation
- **Visual Studio Code** — editor used for code organization and debugging

Operating System

- **Windows 11** (64-bit)

Key Libraries Used

- **PyTorch** — tensor operations and dataset handling
- **NumPy** — numerical operations
- **Matplotlib** — visualization of images and density maps
- **PIL (Python Imaging Library)** — image loading and manipulation

This software environment ensures efficient data processing and seamless integration with deep learning models.

4. Data Exploration

During this stage, the dataset was loaded, inspected, and validated to ensure correctness before model preprocessing. The following activities were performed:

Loading of Images and Annotations

- RGB images were loaded using standard image processing libraries.
- Corresponding ground-truth point annotation files were accessed for density generation.
- Image shapes and annotation counts were checked for consistency.

Visualization of Sample Data

To understand dataset diversity, representative samples were displayed:

- Raw input images
- Their density map counterparts
- Pixel distributions and crowd region intensities

Key Observations

- Significant variation exists in image dimensions across both parts.
- Part A contains extremely dense crowd scenes.
- Part B images are less congested and more uniform.
- Density maps highlight regions with high human concentration.

Challenges Faced

- Handling varying image shapes required resizing during preprocessing.
- Density maps needed careful downsampling to avoid count distortion.
- Some visualization attempts resulted in shape mismatch errors, requiring dimension checks.

5. Data Preprocessing

Data preprocessing is essential to convert raw dataset images and annotations into model-ready tensors. This stage ensures uniformity, stability, and compatibility with deep learning architectures.

Image Transformations

- All images were converted from BGR to RGB format.
- They were resized to a fixed resolution suitable for training pipelines.
- Pixel intensities were scaled to standard ranges for neural networks.

Normalization

- ImageNet-style normalization was applied for compatibility with pretrained backbone networks such as VGG-16.
- This ensures stable gradients and consistent performance across samples.

Density Map Generation

- Annotated points were converted into Gaussian-based density maps.
- Ground-truth density maps were **downsampled by a factor of 8**, matching the output stride of common CNN architectures.
- Density values were multiplied to preserve total count after downsampling.

Tensor Formatting

- Both images and density maps were converted into PyTorch tensors.
- Channel ordering was changed from **H × W × C** to **C × H × W**.
- Batch-ready dataset objects and dataloaders were created.

Handling Shape Mismatches

- Safe error-checking was added to verify matching shapes before visualization.
- Density maps were expanded using channel padding where needed for compatibility with matplotlib.

This preprocessing pipeline prepares the dataset for training, enabling reliable model behavior and consistent representation across all samples.

6. Conclusion

This project establishes a complete data handling pipeline for a crowd monitoring system using density maps. Through careful exploration, preprocessing, and validation, the dataset is transformed into a format suitable for high-quality deep learning models.

The workflow ensures:

- Standardized images
- Accurate and scale-preserved density maps
- Tensor-ready structures for batch processing
- Compatibility with popular architectures such as VGG-based models

These foundations prepare the project for subsequent stages, including model training, evaluation, and deployment.