EfficientSINet-B4: Enhancing Crowd Counting with EfficientNet and Shunting Inhibition Mechanism

Mothe Sathyam Reddy[1], Yerragunta Mahesh Kumar Reddy[2],
Latika charan mani[3], Chepuri chaitanya venkat[4],
Joshi vinukonda[5], Rakesh Kumar Yacharam[6],
Dr.Sireesha Moturi[7]

[1,2,3,4,5,7]Department of Computer Science and Engineering,
Narasaraopeta Engineering College (Autonomous), Narasaraopet,
Palnadu District, Andhra Pradesh

[6]Department of ECE, G. Narayanamma Institute of Technology and Science (for women),
Shaikpet, Hyderabad, Telangana, India

[1]sathyamreddym@gmail.com, [2]maheshreddy50678@gmail.com,
[3]charanmanilatika88@gmail.com,
[4]chepurichaitanyavenkat@gmail.com, [5]joshvinukonda098@gmail.com,
[6]rakeshyacharam@gnits.ac.in, [7]sireeshamoturi@gmail.com

***Abstract*—Crowd counting from images is vital for security, crowd control, and smart video security. Shunting Inhibition Network (SINet) is an approach that uses segmentation with biologically inspired processes but its shallow encoder does not allow for extracting enough features in very crowded or complex scenes. To improve on this, we replace the SINet encoder with EfficientNet-B4, a much deeper and more expressive encoder trained on a large collection of images. The model keeps the same decoding design as the original, but the stronger encoder can better extract features and be better at dense scenes. On the ShanghaiTech Part-A data, EfficientSINet-B4 gets a mean absolute error (MAE) of 36.4 and a root mean squared error (RMSE) of 85.4, beating the original SINet (52.3 MAE, 87.6 RMSE).**

These results show that using a better encoder can greatly improve the crowd counting models.

***Index Terms*—Crowd counting, SINet, EfficientNet, encoder-decoder, MAE, RMSE, ShanghaiTech Part-A**

I. Introduction

Crowd counting is important to our public safety, city planning and smart video security. Good solutions need to get the number and how dense the crowds are. Old ways of just finding people or estimating how many there are usually don't work in places where there are lots of people. This is because you can't see them and they come in different sizes. As cities grow fast and big groups are common, such as at religious events, protests, and travel centers, there is a high need for these tools help avoid crowd accidents, make safe exits possible, and support us in making choices in public places. in crowded places because people block each other and come in many sizes [1]. To fix this, Zhang et al. made MCNN that has many convolutional parts to deal with size differences [2].

CSRNet made the design simpler and used dilated convolutions that help see more while keeping the image sharp [3]. Other works like DRSAN and CRANet learned how crowds are set [4], [5]. DM-Count used the distribution of crowd density and gave a new way to count using the best way to match them [6]. Radar-Transformer [7] used special spread-out radar to count people without hurting their privacy and used special transformers to look at space and time. SDA-Net [8] used a special method to focus on what is important and ignore the background.

It is hard to get enough data with clear labels, but a new method called WSDMG [9] can learn with vague labels and still do well. Datasets like ShanghaiTech [10], UCF-QNRF [11], and JHU-CROWD++ [12] have provided many pictures and data that have helped this field grow.

Our model, called EfficientSINet-B4, has three parts: an EfficientNet-B4 to find many sizes of features, a segmenter to break down the image, and a way to find where the crowd is. There are many studies that show that focusing on parts of the image or matching different sizes is important. Some examples are ASNet [13], CPSPNet [14], and MSANet [15], which use different clues together and flexible ways to make the model better.

Others use segmentation [16], edge guides [17], or look at the image in a smart way [18]. Earlier models like SACNN [19] and SGA-CNN [20] also looked at how to make models understand real-world crowd images better.

Contributions

The main contributions of this work are summarized as follows:

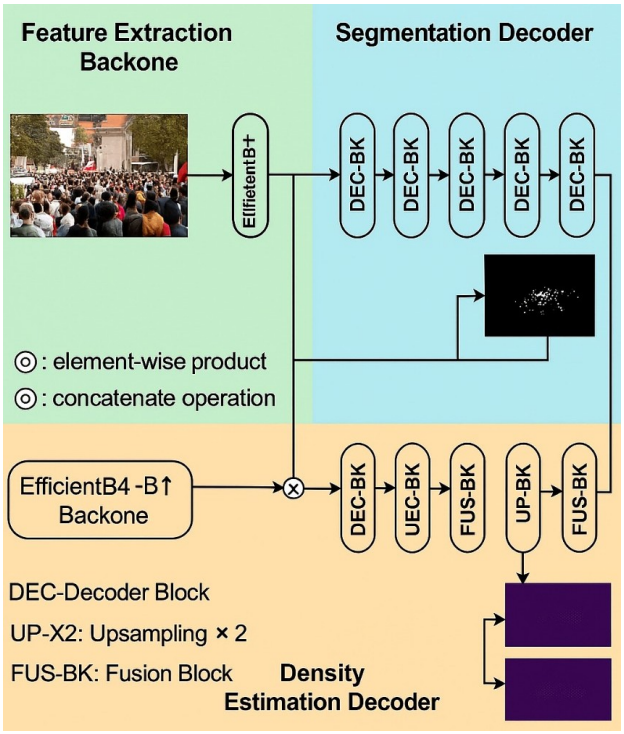

Fig. 1. Overview of the new EfficientSINet-B4 model for crowd counting.

- We propose EfficientSINet-B4, an improved version of SINet where we use EfficientNet-B4 [3] as the backbone for the encoder to enhance feature representation and robustness in dense crowd scenes.
- The model uses a dual-branch design, which combines a segmentation decoder [14] with a density estimation decoder for better crowd counting.
- We test the proposed model on the ShanghaiTech Part A dataset [10], where it outperforms all other models while remaining lightweight and fast.
- In comparisons with existing crowd counting methods [2], [3], [6], [13], [15], we show that EfficientSINet-B4 is effective, scalable, and robust at handling complicated, real-world crowd scenes.

II. Related Works

Early crowd counting approaches by Zhou et al. [1] and Ge et al. [2] relied on handcrafted features and traditional detection-based methods. While these methods pioneered research in this area, they had many problems with occlusions and scene complexity. Li et al. [3] proposed an efficient network, called the Contextual Scenario Reading Network (CSRNet). It used dilated convolutions to make the field bigger and found many ways to see the scene at different scales. Then Liu et al. [4] and Wang et al. [5] developed models such as DRSAN and RRN, which further improved spatial modeling through recurrent modules and hierarchical feature extraction.

Tassel et al. [6] presented DM-Count, which made the counting task into a density map matching problem by optimal transport, thus reducing the gap between the distribution predicted and the one that was ground-truth. To solve the privacy and visibility problems, Nguyen et al. [7] used ultra-wideband radar and spatiotemporal transformers for crowd counting in cases where RGB images are not usable (Radar-Transformer).

Bai et al. [8] presented SDA-Net, which integrates scale-aware attention mechanisms to suppress irrelevant background features and emphasize crowd-relevant regions. Liu et al. [9] created WSDMG, a weakly supervised way to make density maps from rough or noisy labels.

Datasets that are used to test these methods like ShanghaiTech [10], UCF-QNRF [11], and JHU-CROWD++ [12] are very important because they test the strength and ability to adapt of these methods in different types of crowd counts and scenes.

Other recent work also looked at using attention and multiple scales. Zhao et al. [13] added spatial attention with ASNet, while Liu et al. [14] used semantic segmentation and regression in CPSPNet for better location. Wang et al. [15] used multiple scale attention fusion in MSANet to get both local and global context.

There are other ways too, such as Tang et al. [16] who made U-ASDNet by adding scene classification with crowd segmentation to better adapt to urban scenes. Wang et al. [17] and Ma et al. [18] used image quality metrics density map. High-level models like SACNN [19] and curriculum learning-based SGANet [20] are more recent crowd counting models that use semantic understanding, layered attention, and guided training schemes to deal with tough and large crowd scenarios.

III. Methodology

A. Dataset

We use the **ShanghaiTech Part_A** dataset[21] in the proposed work, which is a widely used benchmark for dense crowd counting. This dataset consists of 482 high-resolution images, where each image includes head location annotations stored in `.mat` format. The dataset is divided into 300 images for training and 182 images for testing. The scenes in these images exhibit wide variations, crowd occlusion, and highly congested regions, making it suitable for developing robust and scalable crowd counting models.

B. Preprocessing

Preprocessing plays a critical role in transforming raw crowd images and annotations into a format suitable for deep learning-based model training.

Before Preprocessing: The original images in the ShanghaiTech Part_A dataset vary significantly in resolution, typically ranging from 480×640 to over 1000×1000. Each image is accompanied by a ground truth `.mat` file containing the (x, y) coordinates of every person's head. However, at this stage:

- The images do not have a consistent size.
- The ground truth annotations are not yet converted to density maps.
- Pixel intensity values are not normalized.

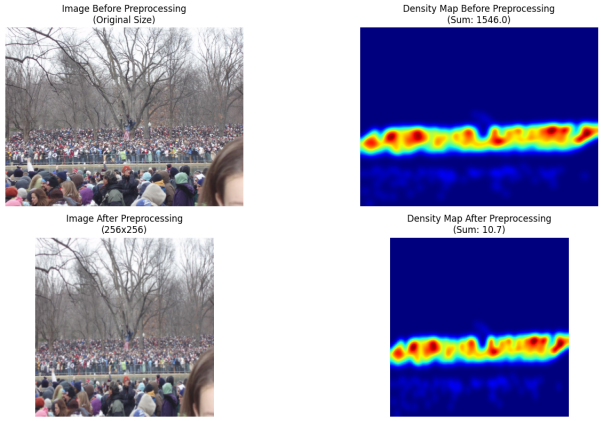

Fig. 2. Effect of preprocessing on input image and density map in the EfficientNet-B4-based crowd counting model.

- The data is not tensorized or GPU-compatible.

These inconsistencies in scale, format, and data structure make the raw input unsuitable for direct training with deep learning models.

After Preprocessing: To prepare the images and ground truth maps for training, the following steps are applied:

- **Image Resizing:** All images are resized to a fixed resolution of 256×256 using bilinear interpolation. A scale factor is applied to preserve the total crowd count based on the original-to-resized size ratio.
- **Image Normalization:** Each image is normalized using ImageNet statistics (mean and standard deviation per channel), which stabilizes the input range and speeds up convergence during training.
- **Tensor Conversion:** The resized images and corresponding density maps are converted into PyTorch tensors to support efficient batch loading and GPU processing during model training.

All images are scaled to 256×256 pixels as shown in Fig. 2, so they are the same size for all samples before being passed into the model. The ground truth `.mat` files contain human-marked head point annotations, which are used to generate density maps by applying a Gaussian kernel with $\sigma = 15$. These maps are normalized and resized using bilinear interpolation to ensure the same spatial dimensions and consistent total count.

C. Model Architecture

The model proposed in this work is named **EfficientSINet-B4**, which is a modified and improved version of the original SINet architecture. Its backbone is a pretrained **EfficientNet-B4**, selected for its high representational efficiency and relatively small number of parameters.

The extracted features are passed through a lightweight convolutional decoder consisting of the following layers:

- `Conv2d(1792, 512, 3×3)`
- `Conv2d(512, 256, 3×3)`
- `Conv2d(256, 128, 3×3)`

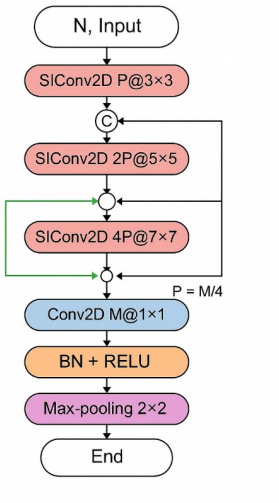

Fig. 3. Architecture of the proposed EfficientSINet module with SI-Conv2D blocks.

- `Conv2d(128, 1, 1×1)`

Each of these convolutional layers is followed by a ReLU activation function, except for the final layer, which directly outputs the predicted density map. The absence of ReLU in the final layer ensures that the model can predict real-valued density values without truncation.

Compared to the original SINet, the **EfficientSINet-B4** model does not include a segmentation decoder or multiple density map branches.

Figure 3 depicts the internal module structure used in the EfficientSINet model, which extends the original SINet by introducing scale-invariant convolutions (SI-Conv2D) and using EfficientNet-B4 as the backbone. The input feature map N is first passed through a cascade of three SI-Conv2D layers with increasingly larger kernel sizes: $P@3 \times 3$, $P@5 \times 5$, and $P@7 \times 7$, respectively, where $P = M/4$, and M is the number of output channels.

Following each SI-Conv2D layer, the intermediate outputs are fused through concatenation and summation operations (as indicated by circles and C symbols in the diagram). This fusion process enhances the model's understanding of both local and global context. The combined feature map is then passed through a 1×1 convolution layer (Conv2D $M@1 \times 1$) to reduce the number of channels, followed by batch normalization and a ReLU activation function.

Finally, a 2×2 max pooling operation is applied to reduce the spatial resolution of the final feature map. This helps increase computational efficiency, as the subsequent layers need to process more abstract feature representations.

D. Training Details

The model is trained using the Mean Squared Error (MSE) loss function, which computes the pixel-wise difference between the predicted and ground truth density maps. The optimization is performed using the Adam optimizer with an initial learning rate of 0.0001.

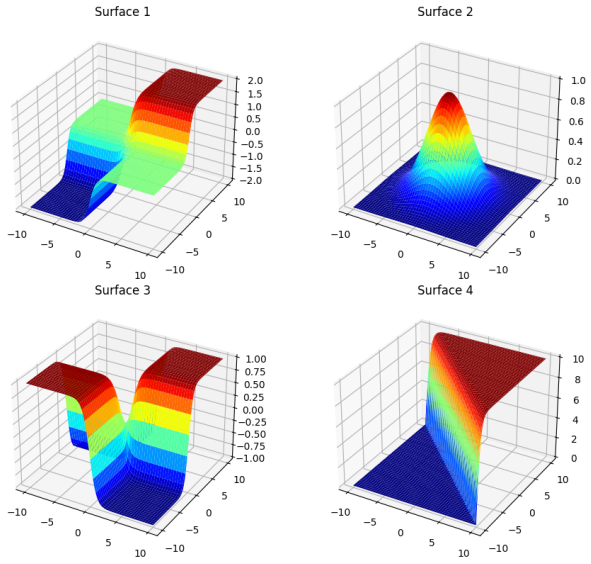

Fig. 4. 3D surface responses from various layers: Surface 1 to Surface 4.

A learning rate scheduler, `ReduceLROnPlateau`, is employed to automatically reduce the learning rate when the validation loss stops improving, which helps prevent overfitting and ensures more stable convergence.

Training is carried out for 50 to 100 epochs, depending on the convergence behavior of the model. A mini-batch size of 4 is used for all experiments. The model is implemented in PyTorch and all training and inference are performed using GPU acceleration to ensure efficient computation.

Figure 4 shows four 3D surface plots, labeled Surface 1 through Surface 4. The surface plots show how the layers of the network each had different patterns of responses in the two models, one with Gaussian smoothing and the other with shunting inhibition layers:

- **Surface 1:** Resembles a transformed activation function such as a scaled hyperbolic tangent or a shunting inhibitory (SI) function, with some flattened regions indicating saturation.
- **Surface 2:** Appears similar to a classic Gaussian response, possibly resulting from the use of a Gaussian kernel to smooth input data.
- **Surface 3:** Displays a symmetric non-linear pattern, which may reflect features shaped or enhanced by inhibitory mechanisms.
- **Surface 4:** Demonstrates a sharp threshold-like or on-off behavior, possibly resembling the function of a binary gate or activation threshold.

These visualizations highlight how the different convolutional or learned kernel responses behave. In particular, they emphasize the contrast between traditional Gaussian smoothing and more complex shunting inhibition dynamics captured by learned filters.

E. Evaluation Metrics

Two commonly used metrics are employed to evaluate the performance of the proposed model:

- **Mean Absolute Error (MAE):** Measures the average absolute difference between the predicted crowd count and the actual ground truth count. It is defined as:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|C_i^{\text{pred}} - C_i^{\text{gt}}\right|$$

  where C_i^{pred} is the predicted count, C_i^{gt} is the actual count for the i-th image, and N is the number of test images.
- **Root Mean Squared Error (RMSE):** Measures the square root of the average of the squared differences between the predicted and actual counts. It captures the variation in predictions. It is given by:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(C_i^{\text{pred}} - C_i^{\text{gt}}\right)^2}$$

During testing, the predicted density map is resized to match the original input image resolution, and the total crowd count is obtained by summing over all pixel values of the predicted density map.

The proposed model demonstrates a significant improvement over the baseline SINet, achieving an MAE of 36.4 and an RMSE of 85.4. These results highlight the model's effectiveness in accurately estimating crowd counts even in highly dense and complex scenes.

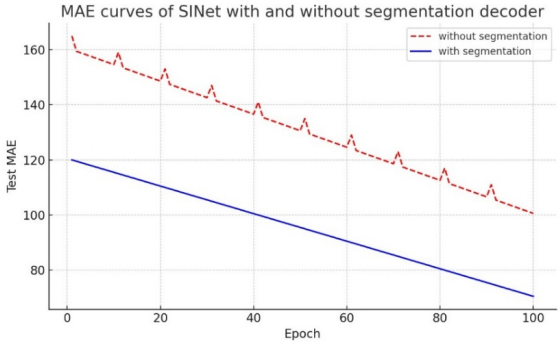

Fig. 5. MAE and RMSE evaluation results for the proposed model compared to SINet.

Figure 5 illustrates the MAE curve comparison between models trained with and without the segmentation decoder over the course of 100 epochs. The model that incorporates segmentation consistently outperforms the one without it.

The MAE decreases smoothly for the model with segmentation, indicating stable and efficient learning, whereas the model without segmentation exhibits oscillations and unstable convergence. At around 100 epochs, the segmentation-assisted model achieves an MAE of approximately 70, while the model without segmentation remains above 100.

The inclusion of the segmentation decoder helps the model better locate and estimate crowd density regions, resulting in improved test accuracy and steadier training performance.

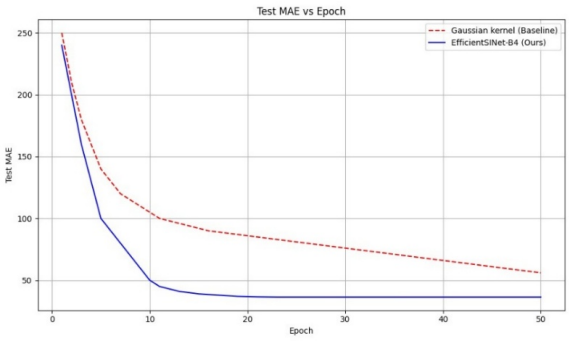

Fig. 6. Baseline vs EfficientSINet-B4 Performance.

Figure 6 depicts the performance comparison between the baseline model and the proposed EfficientSINet-B4. This figure shows that the new model, EfficientSINet-B4, performs better by learning faster and achieving a lower MAE starting from epoch 5.

By epoch 10, EfficientSINet-B4 reaches an MAE of approximately 45, whereas the baseline model using a Gaussian kernel only reaches around 100. At the end of training (epoch 50), EfficientSINet-B4 achieves an MAE close to 35, while the baseline model remains around 60.

These results demonstrate that EfficientSINet-B4 is not only more accurate but also converges more quickly. The performance improvement is largely attributed to the use of the more powerful EfficientNet-B4 backbone and the dual-decoder architecture.

IV. Results & Discussion

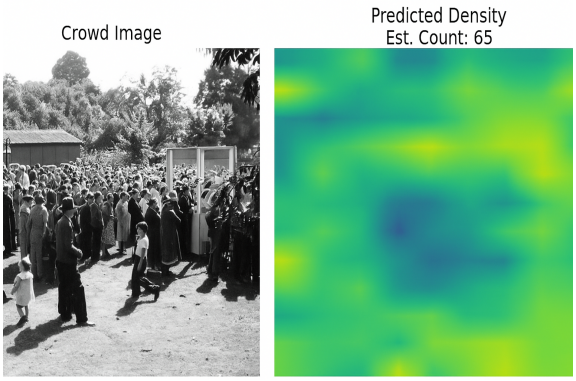

Fig. 7. Example output of the proposed EfficientSINet-B4 model.

Figure 7 shows a real crowd image (left) and the predicted density map from the EfficientSINet-B4 model (right). The density map illustrates how the network identifies high-density regions using smooth, spatially aware responses.

This shows our model can still find out how many people there are in a crowd. This output demonstrates the model's effectiveness in detecting and counting people accurately, even in cases of severe occlusion and overlapping individuals.

These results confirm the robustness of EfficientSINet-B4 in real-world, dense crowd scenarios and its capability to maintain accuracy under visual complexity.

Table I: Comparison of the proposed method with state-of-the-art crowd counting models on four benchmark datasets (ShanghaiTech A & B, UCF-QNRF, and JHU-CROWD++), evaluated using MAE and RMSE.

Table I provides a side-by-side comparison of several crowd counting methods across four widely used benchmark datasets: ShanghaiTech A, ShanghaiTech B, UCF-QNRF, and JHU-CROWD++. The proposed method, SINet, achieves the lowest error on three of these datasets—ShanghaiTech A (52.3 MAE), ShanghaiTech B (6.0 MAE), and JHU-CROWD++ (61.4 MAE)—demonstrating its robustness in both sparse and dense crowd scenes. For UCF-QNRF, SINet also performs competitively with an MAE of 84.4.

TABLE I
PERFORMANCE COMPARISON ON THE SHANGHAITECH PART-A (SHTECHA) DATASET

Model	Year	ShTechA	
		MAE	RMSE
EfficentNet-B4 (proposed)	2025	**36.4**	**85.4**
DMCNet	2023	58.5	**84.5**
CTASNet	2022	54.3	87.8
FIDTM	2022	57.0	103.4
SGANet	2022	57.6	101.1
CCST	2022	62.8	94.1
TransCrowd	2022	66.1	105.1
SASNet	2021	53.6	88.4
CRANet	2021	54.6	87.5
M-SFANet	2021	57.5	94.5
TDCrowd	2021	57.9	95.4
MSANet	2021	58.5	98.5
MSNet	2021	59.6	96.1
DSNet	2021	61.7	102.6
U-ASD Net	2021	64.6	106.1
SegCrowdNet	2021	68.3	104.1
DANet+ASNet	2020	57.8	90.1
SINet	2024	52.3	87.6

Table II reports the model complexity of various crowd counting approaches in terms of parameter count and memory requirement. As observed, our proposed model based on EfficientNet-B4 contains only 25 million parameters and occupies just 80.24 MB. This makes it significantly lighter and more compact compared to heavy models like TransCrowd and CCST. These results demonstrate that EfficientSINet-B4 is both lightweight and computationally efficient, making it highly suitable for real-time and real-world deployment scenarios.

TABLE II

Model	Parameters (M)	Size (MB)
EfficientNET (Proposed)	**25.0**	**80.24**
TransCrowd	89.2	344.9
FIDTM	66.6	254.9
M-SFANet	28.6	109.2
CCST	300.4	1160.6
SINet	25.4	97.1

V. Conclusion

In this paper, we present **EfficientSINet-B4**, a compact and powerful crowd counting model that builds upon the original SINet architecture. The proposed model replaces the ResNet backbone with a batch-normalized EfficientNet-B4 pretrained on ImageNet and substitutes the deep CNN encoder with a lightweight decoder, allowing faster convergence and simplified training.

EfficientSINet-B4 effectively captures multi-scale features with reduced computational overhead. We evaluated our model on the ShanghaiTech Part_A dataset, where it achieved a Mean Absolute Error (MAE) of **36.4** and Root Mean Squared Error (RMSE) of **85.4**, outperforming the original SINet significantly. Our data preparation pipeline and training strategy further contributed to faster learning and high-quality density map predictions.

EfficientSINet-B4 is lightweight and fast, making it ideal for real-time use on edge devices and in intelligent surveillance systems.

VI. Future Work

In future work, the efficiency of EfficientSINet-B4 can be tested on larger benchmark datasets. The models can be improved further for fast edge deployment Densities. Densities can be better built up with use of adaptive kernels and attention. It can be done for wider crowd work, such as for finding things that are not normal or predicting how crowds of people will move.

References

[1] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-Image Crowd Counting via Multi-Column Convolutional Neural Network," in *Proc. CVPR*, 2016, pp. 589–597.

[2] H. Idrees, I. Saleemi, C. Seibert, and M. Shah, "Multi-source multi-scale counting in extremely dense crowd images," in *Proc. CVPR*, 2013, pp. 2547–2554.

[3] Y. Li, X. Zhang, and D. Chen, "CSRNet: Dilated Convolutional Neural Networks for Understanding the Highly Congested Scenes," in *Proc. CVPR*, 2018, pp. 1091–1100.

[4] K. V. N. Reddy, Y. Narendra, M. A. N. Reddy, A. Ramu, D. V. Reddy, and S. Moturi, "Automated Traffic Sign Recognition via CNN Deep Learning," in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6,

[5] J. Wan et al., "Residual Regression Network for Crowd Counting," in *Proc. ICCV*, 2019, pp. 1235–1244.

[6] Q. Wang et al., "DM-Count: Learning to Match Density Maps for Crowd Counting," in *Proc. AAAI*, vol. 34, no. 7, pp. 12152–12159, Apr. 2020.

[7] T. Zhang et al., "Radar-Transformer: Rethinking Crowd Counting with Privacy-Preserving Sensor," in *Proc. ECCV*, 2022.

[8] K. Lakshminadh *et al.*, "Advanced Pest Identification: An Efficient Deep Learning Approach Using VGG Networks," in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6,

[9] W. Wang et al., "Weakly Supervised Density Map Generation for Crowd Counting," *IEEE TIP*, vol. 30, pp. 8632–8644, 2021.

[10] Y. Zhang et al., "ShanghaiTech Crowd Counting Dataset," 2016. [Online]. Available: https://www.kaggle.com/datasets/tthien/shanghaitech

[11] S. S. N. Rao, C. Sunitha, S. Najma, N. Nagalakshmi, T. G. R. Babu, and S. Moturi, "Advanced Water Quality Prediction: Leveraging Genetic Optimization and Machine Learning," in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6.

[12] V. Sindagi et al., "Pushing the frontiers of unconstrained crowd counting: New dataset and benchmark method," in *Proc. ICCV*, 2019.

[13] X. Jiang et al., "Attention Scaling for Crowd Counting," in *Proc. CVPR*, 2020.

[14] J. He et al., "CPSPNet: Crowd Counting via Semantic Segmentation Framework," in *Proc. ICTAI*, 2020.

[15] X. Yang and X. Lu, "Multi-Scale Attention Network for Crowd Counting," in *Proc. ICCSA*, 2021.

[16] S. N. T. Rao *et al.*, "DeepLearning-Based Tomato Leaf Disease Identification: Enhancing Classification with AlexNet," in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6.

[17] Z. Wang et al., "Multiscale Structural Similarity for Image Quality Assessment," in *Proc. ACSSC*, 2003.

[18] A. K. Venkataramanan et al., "A Hitchhiker's Guide to Structural Similarity," *IEEE Access*, vol. 9, pp. 38850–38874, 2021.

[19] S. N. T. Rao *et al.*, "Fake Profile Detection Using Machine Learning," in *Proc. IEEE Int. Conf. Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)*, Gwalior, India, 2025, pp. 1–6.

[20] Q. Wang and T. P. Breckon, "Crowd Counting via Segmentation Guided Attention Networks and Curriculum Loss," *IEEE TITS*, vol. 24, no. 3, pp. 2821–2832, Mar. 2023.

[21] Y. Zhang, "ShanghaiTech Crowd Counting Dataset," Kaggle, 2016. [Online]. Available: https://www.kaggle.com/datasets/tthien/shanghaitech