remote sensing

MDPI

Article

Enhanced YOLOv8-Based Model with Context Enrichment Module for Crowd Counting in Complex Drone Imagery

Abdullah N. Alhawsawi [1], Sultan Daud Khan [2,*] and Faizan Ur Rehman [3]

[1] Department of Information and Scientific Services, Custodian of the Two Holy Mosques Institute for Hajj and Umrah Research, Umm Al-Qura University, Makkah 24236, Saudi Arabia; anhawsawi@uqu.edu.sa
[2] Department of Computer Science, National University of Technology, Islamabad 44000, Pakistan
[3] Saudi Data and Artificial Intelligence Authority, Riyadh 11525, Saudi Arabia; faizanurrehman@gmail.com or furrehman@nic.gov.sa
* Correspondence: sultandaud@gmail.com or sultandaud@nutech.edu.pk

Abstract: Crowd counting in aerial images presents unique challenges due to varying altitudes, angles, and cluttered backgrounds. Additionally, the small size of targets, often occupying only a few pixels in high-resolution images, further complicates the problem. Current crowd counting models struggle in these complex scenarios, leading to inaccurate counts, which are crucial for crowd management. Moreover, these regression-based models only provide the total count without indicating the location or distribution of people within the environment, limiting their practical utility. While YOLOv8 has achieved significant success in detecting small targets within aerial imagery, it faces challenges when directly applied to crowd counting tasks in such contexts. To overcome these challenges, we propose an improved framework based on YOLOv8, incorporating a context enrichment module (CEM) to capture multiscale contextual information. This enhancement improves the model's ability to detect and localize tiny targets in complex aerial images. We assess the effectiveness of the proposed framework on the challenging VisDrone-CC2021 dataset, and our experimental results demonstrate the effectiveness of this approach.

Keywords: crowd counting; tiny target detection; aerial images; deep learning; YOLOv8

Citation: Alhawsawi, A.N.; Khan, S.D.; Rehman, F.U. Enhanced YOLOv8-Based Model with Context Enrichment Module for Crowd Counting in Complex Drone Imagery. *Remote Sens.* **2024**, *16*, 4175. https://doi.org/10.3390/rs16224175

Academic Editor: Riccardo Roncella

Received: 14 August 2024
Revised: 2 October 2024
Accepted: 15 October 2024
Published: 8 November 2024

1. Introduction

Analyzing crowded scenes is crucial for efficient crowd management [1]. Poor crowd management in public places, such as marathons, large religious gatherings, and political events, can lead to stampedes and other crowd-related disasters [2]. To ensure effective crowd management, it is essential to develop computational models that can analyze and understand crowded scenes through surveillance cameras. However, crowded scenes present challenges due to the complex interactions between individuals in unconstrained areas [3]. Various crowd-related problems, such as crowd tracking [4], counting [5], and congestion detection [6], have been extensively studied in the literature. Among these issues, crowd counting is particularly important as it serves as a prerequisite for comprehensive crowd analysis.

Crowd counting entails estimating the number of individuals in a scene. This information helps crowd managers identify congested areas and implement safety measures to prevent potential crowd disasters. Additionally, accurate crowd counting can aid in the efficient utilization of resources, such as deploying the appropriate number of security personnel and effectively managing logistics and infrastructure for public gatherings. Crowd counting is a widely studied topic, with numerous researchers proposing various sophisticated models to achieve accurate results. For instance, Wang et al. [7] recently introduced a model, SDANet, which incorporates a scale awareness module to utilize scale information in images and videos for crowd counting. Guo et al. [8] introduced a dual convolution neural network (Dual-CNN) for crowd counting. The first network estimates

the density map from the input image and the second network then re-constructs the crowd image from the estimated density map. Several other models including refs. [9–13] have been proposed in recent years for crowd counting in image and videos.

Most existing models, including the aforementioned ones, focus on crowd counting in natural images (images captured from on-ground cameras). However, there has been limited research on crowd counting using drone imagery. Crowd counting in drone imagery holds significant importance due to the unique capabilities of drones. For example, drone imagery provides a comprehensive view of large and dense crowds that might be challenging to analyze from natural images captured by on-ground cameras. Additionally, UAVs can access areas that are difficult for human operators to reach, ensuring real-time monitoring and rapid response.

Recently, Ptak et al. [14] conducted a study evaluating the performance of different deep learning models deployed on edge devices for crowd counting in drone images. Nag et al. [15] introduced an encoder–decoder framework, namely, Attention-based Real-time CrowdNet (ARCN), for crowd counting in drone imagery. Bakour [16] optimized CSRNET (with VGG-16 as the backbone) for real-time crowd density estimation using drone video sequences. Elharrouss et al. [17] presented a framework that employs dilated and scaled neural networks to perform feature extraction and density estimation. Peng et al. [18] proposed a framework for crowd counting in adverse conditions, particularly at night and in haze. Their study introduced a novel dataset, the RGB-Thermal dataset (DroneRGBT), and proposed the MMCCN that leverages visible and thermal infrared information. Liu et al. [19] compiled a high-quality dataset for crowd counting in drone-captured images, called Visdrone-CC2020, and organized a competition that attracted numerous researchers to develop models and techniques to address the challenges of crowd counting in drone imagery. They utilized the existing crowd counting models, LCFCN [20], CSRNet [21], Switch-CNN [22], and DM-Count [23], originally developed for natural images, and tested these models on drone images for crowd counting.

Despite the success of the aforementioned models for crowd counting in drone images, they face difficulties due to the inherent challenges associated with drone imagery. (1) Drone images are captured from varying altitudes and angles, thus leading to changes in target size, shape, and perspective. (2) The background in drone imagery can be highly cluttered and complex. (3) In high-resolution drone imagery, the targets of interest often appear very small, occupying only a few pixels in the image. This presents a significant challenge for deep learning-based target detectors, as the image is downsampled after passing through a series of convolutional and pooling layers, resulting in a loss of information for small targets.

For small target detection in drone imagery, the recently introduced YOLOv8 has achieved significant results [24–27]. Despite its high detection accuracy, the network faces challenges when applied to the detection and counting of people in drone imagery. For detecting tiny targets in aerial images, YOLOv8 faces challenges primarily because of the target's extremely small size (just a few pixels) within a larger scene. Because of the small size of the targets, these targets often lack distinctive contextual cues that help the model to distinguish them from the background. Furthermore, YOLOv8 uses downsampling techniques to boost speed, which can result in a loss of fine details crucial for identifying small targets. Additionally, the receptive field of the model may not be adequately tuned to focus on these small features. To address this issue with YOLOv8, we introduce a context enrichment module (CEM) that enhances the receptive field of the convolutional layers through the use of an atrous convolutional layer. The main contributions of this paper are listed as follows:

1. We propose a modified YOLOv8-based framework specifically tailored for crowd counting in aerial images. The model is capable of accurately detecting, localizing, and counting the individuals in a complex environment with varying crowd densities and altitudes.

2. We enhance YOLOv8 by introducing a CEM, which significantly improves its ability to detect small targets. The CEM effectively captures multiscale contextual information and increases the model's ability to differentiate the tiny targets from complex backgrounds.
3. To illustrate the efficacy of the proposed framework, we apply the model to a complex and challenging dataset, VisDrone-CC2020 [19]. However, the dataset provides dot annotations, which is incompatible with YOLOv8. To facilitate the training, we introduce a method that converts dot annotations into four-tuple bounding box annotations.

The rest of this paper is organized as follows: The related work is discussed in Section 2. Section 3 discusses the proposed methodology and different components of the framework and Section 4 discusses the detailed results. Section 5 concludes this paper.

2. Related Work

In this section, we will discuss the related work for crowd counting. Generally, we divide the related work into two groups: (1) crowd counting in natural images and (2) crowd counting in drone images.

2.1. Crowd Counting in Natural Images

The early approaches in crowd counting and density estimation primarily utilized regressors [28–30] to map extracted features to the count. These features were derived using hand-crafted feature extractors [31]. However, these models required perspective normalization to estimate the scale of the person in the image, which requires additional efforts and computation.

Recent advancements have shifted toward using CNNs to generate density maps for crowd counting tasks. The multicolumn architecture was introduced to address scale variation by employing receptive fields of different sizes to extract features at multiple scales [32,33]. An enhanced CNN architecture, known as the switching-multicolumn architecture, was later proposed to handle significant variations in crowd density [22]. These models were designed to address scale variations by using receptive fields of different sizes to extract features at multiple scales; however, multiple columns cause computational overhead in selecting the appropriate number of columns and receptive field sizes.

Similarly, a feed-forward network was used for crowd counting, which takes a low-resolution density map as input and generates high-resolution density maps [34]. This model improves the process by taking a low-resolution density map and generating detailed density maps, which are useful in crowd counting; however, the generation of high-quality density maps leads to computational complexity. Although the multicolumn architecture effectively addresses scale variation, it incurs computational overhead due to the need to select the appropriate number of columns and their receptive field sizes [35]. A single-column approach, such as MSCNN [36], employs a scale aggregation block to manage scale variation. Although MSCNN reduces computational complexity compared to multicolumn networks by using a scale aggregation block, the model may be less effective in handling large scale variations compared to the multicolumn method. Building on the concept of scale aggregation, Cao et al. [37] introduced composition loss and local pattern consistency loss to enhance crowd counting accuracy. The model requires fine-tuning and careful implementation of the loss functions, making it harder to generalize across different datasets. Sam et al. [38] proposed a growing CNN approach that recursively splits into child CNNs for improved crowd counting performance. Expanding on this idea, Sindagi et al. [39] introduced a multitask CNN with a cascaded approach to classify crowds into different density levels. Although the model enables the CNN to classify crowds into different density levels, improving both the counting and classification of crowded scenes, this approach may introduce additional complexity, making the model more challenging to train. Idrees et al. [40] incorporated DenseNet blocks and multiple loss functions to optimize ground-truth crowd density maps, allowing the architecture to compute the crowd count, density, and localization simultaneously. Xiong et al. [41] proposed a convolutional LSTM to utilize temporal information for counting people. However, most datasets consist

of still images and lack temporal correlation. An end-to-end encoder–decoder, called the Automatic-Scale Network (AMSNet) [42], was developed through NAS. Similary, Zhai et al. [43] presented a crowd counting framework, namely, FPANet, which uses a lightweight feature pyramid, attention, and multiscale aggregation modules to improve accuracy and efficiency in real-world applications. Wang et al. [44] proposed CAFNet, which enhances crowd counting by integrating local, cross-level, and cross-layer context information through specialized modules, resulting in a high-resolution density map. Du et al. [45] presented a novel crowd counting framework that incorporates a hierarchical mixture of density experts. Wang et al. [46] introduced a self-supervised crowd counting framework that reduces the burden of heavy annotations and leverages a large number of easily obtainable unlabeled images. This framework reduces the need for heavy annotations by leveraging self-supervised learning, making it more practical in scenarios where labeled data are scarce. Zhang et al. [47] presented a new crowd counting framework called CrowdGraph. This graph-based method redefines crowd counting by approaching it from a graph-to-count perspective. One of the disadvantages of a graph-based model is that these methods may struggle with scalability, especially when applied to large and densely populated scenes. Chen et al. [9] proposed a metric learning approach to estimate crowd characteristics from a single annotated image of a scene. The approach employs a Multi-Prototype Learner, trained via Expectation-Maximization, to capture foreground and density prototypes. Yan et al. [48] introduced DFNet for accurate crowd counting in crowded and noisy scenes. Although DFNet offers robustness in challenging environments where other models may fail, the model requires additional computational resources due to its complexity.

2.2. Crowd Counting in Drone Images

UAVs are gaining popularity for crowd monitoring due to their easy deployment, low cost, and ability to provide high-resolution real-time images. Most approaches discussed in the literature, however, rely on datasets captured by static cameras. Elharrouss et al. [17] proposed a framework for crowd counting utilizing drone-collected data. Their approach employs dilated and scaled neural networks to extract features and estimate crowd density. Kuchhold et al. [49] proposed a scale-adaptive approach for crowd detection and counting in drone images. The framework utilizes local feature points and density estimation across various image scales. However, its reliance on local feature points may limit its effectiveness in extremely dense crowds where feature points overlap. Zhang et al. [50] introduced an Enhanced Multi-Modal Crowd Counting Network (I-MMCCN), which integrates a hard example mining module along with a new Block Mean Absolute Error (BMAE) loss function. The BMAE enhances local spatial correlation and aligns with evaluation metrics. Although the introduction of the BMAE enhances the performance, it increases the model complexity. Castellano et al. [51] proposed a method using a fully convolutional network to detect and track crowd movement in video sequences by clustering crowd-dense areas and identifying their centroids. The method is useful for video sequences and the performance of the model is compromised in static crowd counting or scenarios where individuals are stationary. Chen et al. [52] presented Flounder-Net, an efficient deep learning model that uses interleaved group convolution to reduce network redundancy and employs rapid feature map shrinkage to effectively manage high-resolution images. The model is fast and effective; however, the reduced redundancy may result in a loss of detail and may not be able to detect small objects. Castellano [53] proposed a lightweight fully convolutional neural network for real-time crowd detection that combines classification and regression tasks to accurately identify and focus on crowded areas. The model lowers computational costs while maintaining reasonable accuracy; however, the simplicity of the model might limit its ability to handle more complex scenarios. Bai et al. [54] introduced SACANet, an innovative network tailored for crowd counting that adapts to scales and incorporates long-range context awareness. The model is particularly effective in managing both small-scale and large-scale features; however, the incorporation of long-range context awareness may

increase the computational cost. Zhao et al. [55] proposed PDNet, a novel network that includes a multiscale backbone, a Dilated Feature Fusion (DFF) module to manage small targets and scale variations, and a Density Map Attention (DMA) module to focus on target locations within complex backgrounds. The addition of DFF and DMA modules may slow down the processing times and increase resource consumption. Bahmanyare et al. [56] introduced the DLR Aerial Crowd Dataset (DLR-ACD), which comprises 33 large aerial images with 226,291 annotated persons. Additionally, they introduced the Multi-Resolution Crowd Network (MRCNet), an encoder–decoder CNN built upon VGG-16, aimed at precise crowd counting and density map estimation. The model relies on VGG-16 as the feature extractor, which is an outdated model and may not be as efficient in terms of computation and resource usage. Husman et al. [57] presented a literature review on drone specifications, on-board sensors, power management, and analysis algorithms and discussed the ethical and privacy issues associated with using UAVs for crowd monitoring. Gu et al. [58] presented a novel framework for crowd counting for drones. This framework fuses visible and thermal infrared images to accurately count dense populations and guide drone flights. Almagbile et al. [59] developed a method that employs the Feature from Accelerated Segment Test (FAST) algorithm to identify crowd features in drone images. Although the method is fast due to its reliance on computing the rapid corner detection and feature extraction capabilities, the model might struggle to manage complex scenes or in situations with severe occlusions. Castellano [53] proposed a lightweight FCN-based model for crowd detection in crowd images. The model leverages spatial graphs and a clustering technique to improve the detection performance. Although the model is suitable for deployment on UAVs due to its lightweight architecture, a clustering-based approach might introduce bias by predicting the presence of a crowd even with the presence of a few individuals in the scene.

Despite the growing popularity of UAVs for crowd monitoring due to their easy deployment and low cost, many current methods have significant limitations. For instance, some frameworks focus on specific image scales or feature points, which may not perform well under different conditions. Techniques involving complex modules can be computationally intensive, potentially hindering real-time processing. Additionally, methods using algorithms like FAST may have difficulties with varying camera orientations and positions, reducing their robustness in diverse scenarios.

3. Proposed Methodology

In this section, we will discuss the proposed architecture for small target detection. Generally, the proposed architecture uses YOLOv8 as the base architecture; however, to address the challenges associated with YOLOv8, we have modified the architecture to better handle small target detection.

YOLOv8 typically comprises the backbone, the neck, and the head. The backbone module is a CNN that processes the input image to extract detailed, multiscale hierarchical features. After extracting the hierarchical features, the resultant feature maps are then provided as input to the neck module. The neck module further refines the feature maps obtained from the backbone module. This module enhances the spatial and semantic features by employing additional convolutional layers and the feature pyramid pooling module. The feature maps with different resolutions are provided as input to the head module where target detection is performed. The head module includes detection subnetworks that analyze the features provided by the neck to produce predictions for each potential target. Subsequently, non-maximum suppression (NMS) is used to eliminate overlapping predictions and keep only the most reliable detections.

Although YOLOv8 performs real-time target detection with high accuracy in natural images, it faces significant challenges in identifying small targets within aerial images. To accurately detect small targets, it is crucial to incorporate contextual information that defines the target based on its surrounding environment. Small targets often lack distinctive features, so understanding the context in which these targets appear can significantly

improve detection accuracy. For example, identifying a small vehicle on a road becomes easier when the model recognizes the road and other related surroundings. In order to enable YOLOv8 to detect small targets in aerial images, we modify YOLOv8. The detailed architecture of the proposed framework is illustrated in Figure 1.

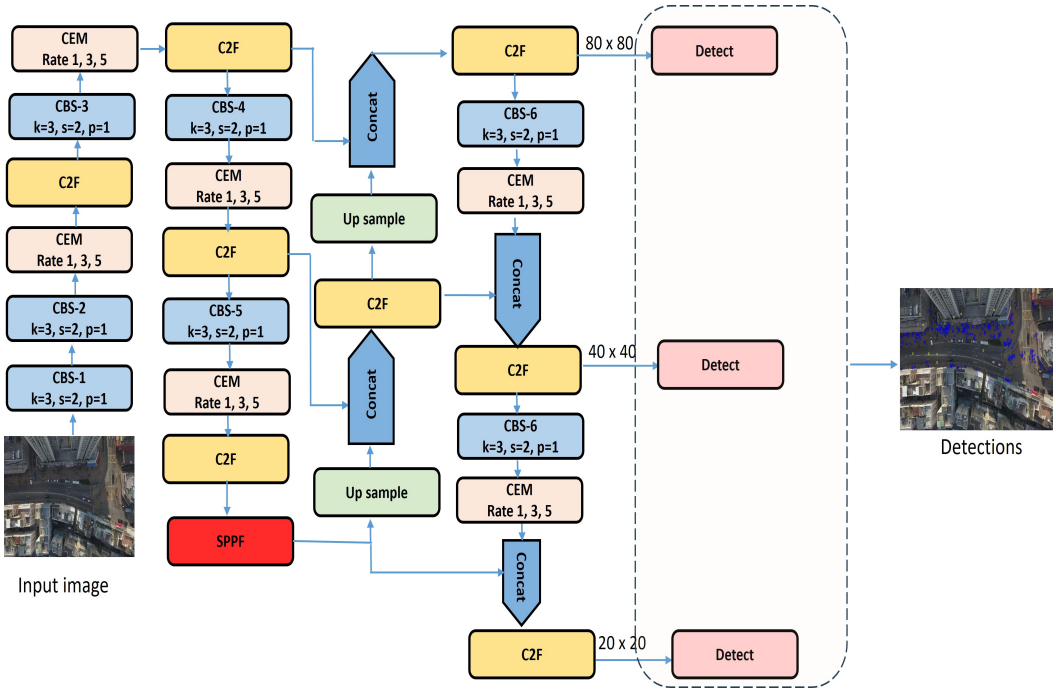


Figure 1. Detailed pipeline of proposed framework for small object detection (zoomed in for best view).

As illustrated in Figure 1, the input image is applied to the backbone of the network. The backbone consists of five convolution–batch normalization–SiLU (CBS) blocks, four C2F blocks, and the four proposed CEMs. The details of the modules are provided below:

The CBS module in the YOLOv8 architecture is designed to enhance feature extraction and improve the overall efficiency and performance of the network. The CBS module generally comprises a convolutional layer, followed by batch normalization and the SiLU activation function. Specifically, the convolutional layer uses a kernel size $k = 3$, stride $s = 2$, and padding $p = 1$. The convolutional operation of the CBS module downsamples the input feature map while retaining spatial information. After convolution, batch normalization is applied to normalize the output, stabilizing the learning process and accelerating convergence. Finally, the SiLU activation function is employed, introducing non-linearity and allowing the network to learn more complex features.

The C2F module, newly introduced in YOLOv8, improves the integration of features with contextual information, which leads to improved detection accuracy [60]. It effectively incorporates the principles proposed by the ELAN module [61], optimizing the network structure by controlling the shortest and longest gradient paths, which also improves the network training [62].

The architecture of the C2F module is illustrated in Figure 2. As shown, the C2F module divides the input feature maps into two separate paths. This split aligns with the CSPNet (Cross-Stage Partial Network) [63] concept, ensuring that one part of the feature map is processed through the bottleneck modules while the other part remains unchanged.

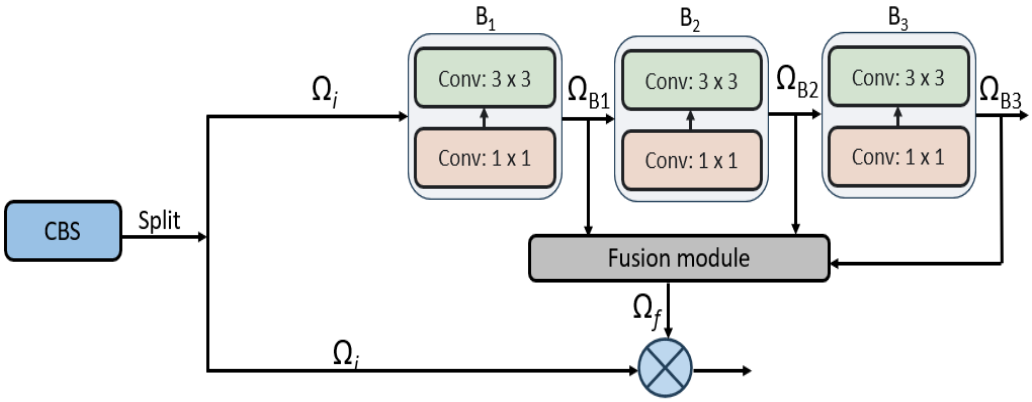


Figure 2. Detailed architecture of C2F module.

Let Ω_i be the input feature map to the C2F module. The original map Ω_i is subsequently processed through three bottleneck modules, namely, B_1, B_2, and B_3. Each bottleneck module employs two convolutional layers, 1×1 followed by 3×3. Let Ω_i be provided as input to the bottleneck module B_1. The output feature map Ω_{B_1} is provided as input to B_2. Similarly, the output feature map Ω_{B_2} is then provided as input to the bottleneck module B_3. Let the output feature map of the bottleneck module B_3 be Ω_{B_3}. Finally, all three output feature maps $\{\Omega_{B_1}, \Omega_{B_2}, \Omega_{B_3}\}$ are fused together to obtain the fused feature map Ω_f. The fused feature map Ω_f and original input feature map Ω_i are then combined together through the concatenation block.

3.1. Context Enrichment Module

As discussed above, to accurately detect small targets, it is imperative to capture contextual information around small targets in aerial images, which can be achieved through the CEM. The CEM captures contextual information by expanding the receptive field of the convolutional layers through the use of atrous convolutional layers, without increasing the computational load. Generally, the CEM increases the receptive field of the model and ensures that even minute details and the larger context are considered, which boosts the detection accuracy of small targets.

The detailed architecture of the CEM is illustrated in Figure 3. From Figure 3, it can be seen that the CEM consists of three parallel branches, and each branch consists of an atrous convolutional layer with a kernel size of 3×3 pixels but with different dilation rates of 1, 3, and 5. We keep the kernel size 3×3 pixels, which is small enough to be computationally manageable while still being large enough to capture essential features and local context around each pixel.

The reason for using different dilation rates in the parallel branches is to capture multiscale contextual information. A dilation rate of 1 corresponds to a standard convolution, capturing fine-grained details and local features. A dilation rate of 3 expands the receptive field moderately, enabling the detection of larger patterns and more context around the small targets without losing too much fine detail. The highest dilation rate of 5 significantly increases the receptive field, allowing the model to capture even broader contextual information, which is essential for understanding the surroundings of small targets within the vast aerial images.

The outputs of these differently dilated convolutions are then fused via concatenation, effectively combining multiscale contextual information into a fused feature map. This fusion ensures that the model can leverage detailed local features and broad contextual cues. By integrating these varied scales of context, the CEM significantly improves the model's ability to distinguish and accurately detect small targets in complex aerial scenes.

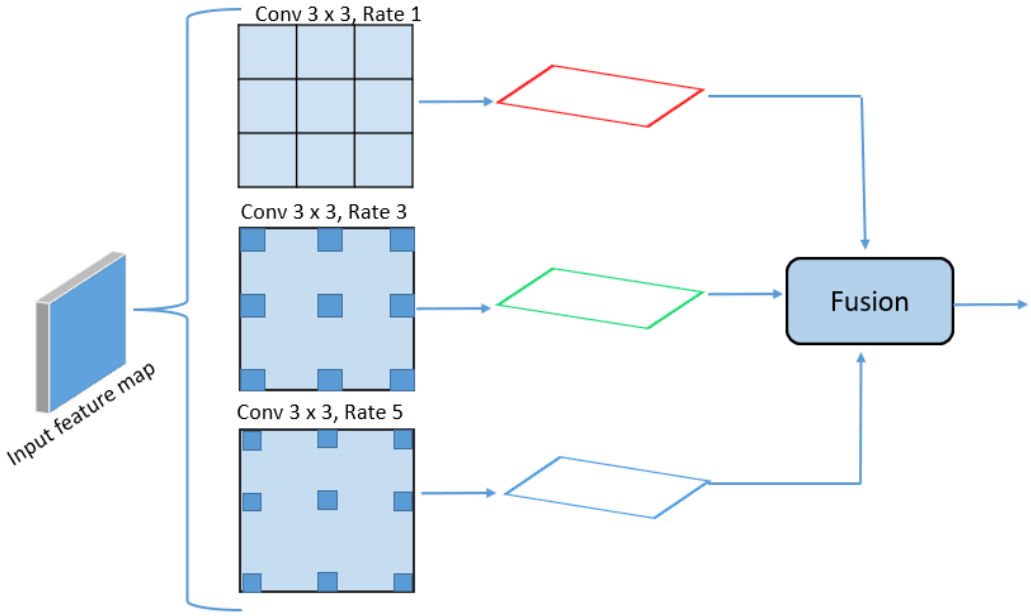
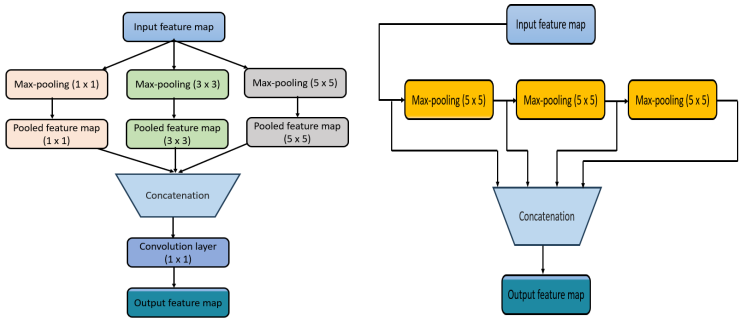


Figure 3. Detailed architecture of CEM.

3.2. Spatial Pyramid Pooling Fast (SPPF)

The spatial pyramid pooling fast (SPPF) module is similar to the spatial pyramid pooling (SPP) module, which aids in handling targets at different scales. Specifically, the SPP was introduced to generate a fixed-length feature representation of the input regardless of the input image's size. The SPPF in YOLOv8 is the optimized version of SPP, which is designed to capture multiscale features while reducing the computation complexity.

The overall architectures of the SPPF and SPP modules are illustrated in Figure 4.



(**a**) SPP (**b**) SPPF

Figure 4. Detailed architectures of SPP and SPPF.

The SPP module takes the input feature map generated by the backbone network of the model. The input feature map is subsequently passed through three parallel Max-Pooling layers, each with a different kernel size, 1, 3×3, and 5×5, each with a stride of 1. The 1×1 pooling layer does not change the feature map. The 3×3 pooling layer, with a padding of 1, captures a broader context by considering neighboring pixels. Similarly, the 5×5 pooling layer, with a padding of 2, captures an even larger context. Each of these pooling operations produces a pooled feature map at their respective scales. These pooled feature maps, together with the original input feature map, are then concatenated along the channel dimension, creating a feature map that integrates multiscale information. This concatenated feature map is subsequently passed through a 1×1 convolution layer, which serves to reduce the number of channels and fuse the multiscale features into a unified representation. The output of this convolution layer is the refined feature map, which is

then subsequently passed to the detection head of the network to predict bounding boxes, target classes, and confidence scores.

The SPP module incurs computational costs due to parallel pooling operations with different sizes. In contrast to the SPP module, the SPPF module replaces the parallel pooling layers with a 5x5 pooling operation applied in a serial fashion. These pooling layers enlarge the receptive field, allowing the model to capture information at different scales without the need for multiple separate pooling operations, as in the traditional SPP.

4. Experiment Results

In this section, we initially present the details of the dataset utilized, followed by a comprehensive evaluation of the proposed framework. We then compare the proposed framework with other related methods. These experiments are designed to showcase the effectiveness and advantages of our approach compared to the existing methods.

The proposed framework was developed using the PyTorch library. The experimental setup for our study included a hardware configuration featuring an Intel Core i5 CPU and an NVIDIA TITAN V GPU with 12 GB of memory. The operating system used was Ubuntu 22. For the deep learning framework, we utilized PyTorch version 1.9.2, along with CUDA 11.4 and cuDNN 11.4 for GPU acceleration.

To optimize the loss function, we employed stochastic gradient descent (SGD), a widely used optimization technique known for its efficiency and effectiveness in training deep learning models. The training process spanned 100 epochs, with an initial learning rate set at 0.001. To ensure optimal convergence and to adaptively fine-tune the learning rate throughout the training process, we utilized a cosine annealing algorithm. This method gradually reduced the learning rate, enhancing the model's performance and stability during the training phase.

4.1. Dataset

To assess the effectiveness of the proposed framework, we utilized the Visdrone-CC2020 dataset [64,65], which was collected by the AISKYEYE team from the Lab of Machine Learning and Data Mining at Tianjin University, China. The dataset was gathered using drones equipped with cameras, covering 70 different scenarios. It contains 112 video sequences, with 82 sequences used for training and the remaining 30 sequences used for testing. Each sequence consists of 30 images, each with a resolution of 1920 $\times$ 1080 pixels. Thus, 2460 images are used for training, and the remaining 900 frames are used for testing. In each image, human heads are annotated with points, resulting in a dataset containing 4.8 million head annotations.

The dataset presents significant challenges for crowd counting models due to its complexity. One notable feature is the variety of scenes with different crowd densities, ranging from sparse gatherings to highly congested areas. Additionally, the presence of various targets, such as vehicles, trees, and buildings, adds to the challenge by introducing background clutter and occlusions, which can significantly impact the performance of crowd counting models. Sample images from different video sequences are illustrated in Figure 5.

Figure 5. Sample frames from diverse scenes represent comprehensive coverage of different environments and conditions.

4.2. Evaluation Metrics

In this work, we assess the performance of the proposed framework in two distinct areas: (1) localization (detection) accuracy, and (2) counting accuracy.

For localization performance, we use the widely adopted metric for target detection, the mean average precision (mAP). The mAP determines whether a prediction is a true positive (TP) or a false positive (FP) based on the threshold value of the Intersection over Union (IoU). The mAP is formulated as Equation (1).

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i \tag{1}$$

where N represents the number of predictions, and precision is defined as Precision $= \frac{TP}{TP+FP}$

To assess counting performance, we employ two commonly used evaluation metrics: the Mean Absolute Error (MAE) and Mean Squared Error (MSE).

The MAE measures the average magnitude of errors between the predicted counts and the actual counts, providing an overview of the model's performance in predicting the count. The MAE is formulated in Equation (2).

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{2}$$

Conversely, the MSE quantifies the average squared difference between the predicted count and actual count. In this way, the MSE gives more weight to larger errors, thus penalizing significant discrepancies more severely. The MSE is formulated in Equation (3).

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

where y_i represent the ground-truth count, $\hat{y}_i$ is the predicted count, and n represents the number of samples.

4.3. Conversion of Dot Annotations to Bounding Boxes

To train the proposed framework, bounding boxes are required. However, the annotations provided in the publicly available dataset are in the form of dot annotations, where each dot represents a person in the scene, given as (x, y) coordinates. In contrast, bounding boxes need four points: the location (x and y coordinates) and the dimensions (width and height) of the box, which correspond to the human head in the scene. For simplicity, we assume that the height and width of the bounding box are equal.

To convert these dot annotations to bounding boxes, we need to estimate the size of each head. Due to perspective distortions, the size of a person varies across different regions of the image. To address this challenge, we generate a perspective map K by manually annotating a few human heads from the top to the bottom of the image. Using these annotated samples, we then apply a regression technique adopted in [66] to estimate the head size across the entire image and generate a scale map. The scale map is a 2D representation with dimensions equal to the input image. Each pixel in the map indicates the estimated head size (in pixels) at the corresponding location in the image.

Let p_i be a dot annotation representing a person at location (x, y). The perspective map K provides the size of the head at each point (x, y). We then generate the bounding box b_i for dot annotation as [x, y, K(x, y), K(x, y)]. This approach allows us to accurately convert the dot annotations into bounding boxes, taking into account the varying scales and ensuring the proposed framework is effectively trained.

The overall process of converting the dot annotations into bounding boxes is illustrated in Figure 6.

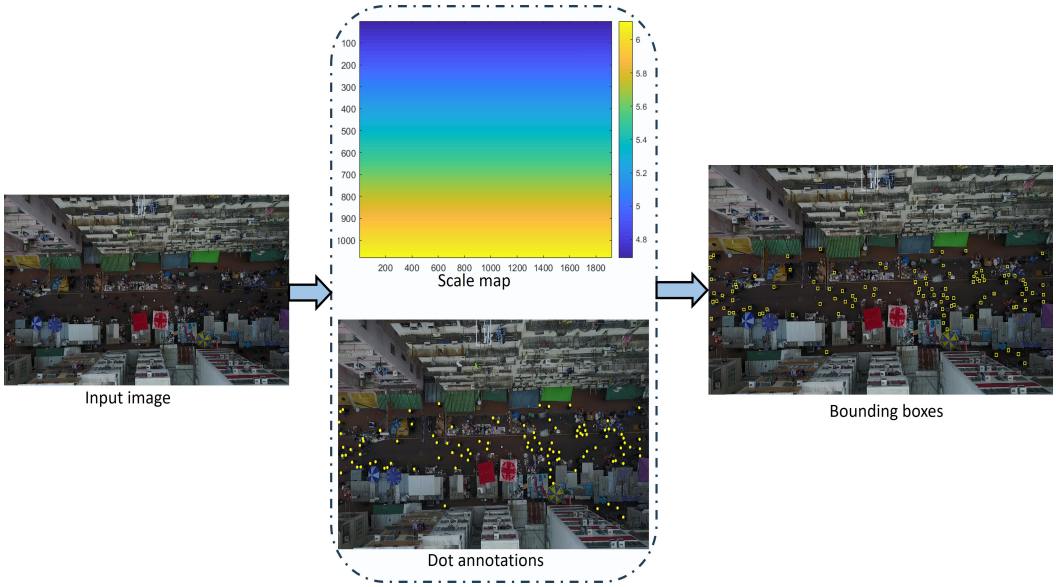


Figure 6. Pipeline of converting dot annotations into bounding boxes.

4.4. Comparisons with Different Variants of YOLOv8

To effectively evaluate the performance of the proposed framework, we compare different variants of YOLOv8 with our framework. The details of these comparisons are provided in Table 1. From Table 1, we observe a clear trend: as the model size and complexity increase, so does the accuracy. The smallest variants, YOLOv8n and YOLOv8s, achieve mAP@50 scores of 0.51 and 0.57, respectively, and mAP@70 scores of 0.42 and 0.51, which are lower compared to the more complex variants of YOLOv8. This is due to their simplified architecture, which is designed to optimize speed. Tiny targets, such as pedestrians in this case, require high-resolution feature maps and the ability to capture fine-grained details across various scales. The reduced depth and breadth of YOLOv8n and YOLOv8s limit their ability to generate and process these detailed feature maps, making it challenging to accurately detect tiny targets. Additionally, these models produce lower-resolution feature maps and have fewer detailed features, making it harder for them to detect small targets in drone images.

In contrast, the more complex variants of YOLOv8, such as YOLOv8l and YOLOv8x, extract higher-resolution feature maps due to their significantly greater number of parameters and layers. This allows these models to preserve fine-grained details about small targets, which is crucial for detecting tiny objects in high-resolution drone images.

Despite the complex variants of YOLOv8 achieving better results compared to their simpler counterparts, the proposed framework surpasses other complex variants of YOLOv8 in detecting tiny objects in high-resolution drone images by addressing the inherent limitations of the original model. The proposed model effectively integrates the CEM that enhances the receptive field of the convolutional layers, allowing the model to capture more contextual information surrounding small objects. Additionally, the improved receptive field helps the model focus on even the smallest features, making it better at distinguishing tiny objects from complex backgrounds.

From Table 1, it is also evident that the complexity of the proposed framework increases as well as the size of the proposed framework (due to addition of the CEM), which also leads to longer inference times compared to other models. However, this added complexity results in better accuracy compared to its counterparts.

Table 1. Performance comparison of various YOLOv8 variants and proposed model in terms of mAP.

Model	Parameters	mAP@50	mAP@70	I.T (ms)	Size (MB)	GFLOPs
YOLOv8n	3.01	51.27	42.51	7.20	6.5	8.2
YOLOv8s	11.12	57.65	51.41	12.42	22.6	28.4
YOLOv8m	25.84	68.82	56.34	20.50	52.1	78.7
YOLOv8l	43.61	76.10	65.29	16.40	87.8	164.8
YOLOv8x	68.12	79.86	70.46	19.00	136.9	257.4
Proposed	72.54	82.10	76.23	21.10	164.2	294.5

4.5. Comparisons with Different Generic Detectors

To evaluate the detection performance of the proposed framework, we compared it with other similar generic detectors, and the results are presented in Table 2. From Table 2, it is evident that all state-of-the-art detectors perform lower compared to the proposed framework. The experimental results show that Faster R-CNN achieves an mAP@50 of 23.74 and an mAP@70 of 15.32, which are significantly lower than other detectors, like the Normalized Wasserstein Distance (NWD) [67], YOLOv9n, YOLOv10n, YOLOv3-spp, Cascade R-CNN, and the proposed method. This indicates that Faster R-CNN struggles with this particular crowd counting task. One limitation of this type of two-stage detector is that it uses the last convolutional layer for feature extraction. The problem with this strategy is that the receptive field of the last convolutional layer is very large, which causes information about small objects to be lost due to the subsequent pooling and convolutional layers. Since the size of people in the VisDrone2020 dataset is extremely small, Faster R-CNN faces challenges in detecting these small objects in drone images. On the other hand, one-stage detectors like YOLOv9n, YOLOv10n, YOLOv3-SPP, and YOLOv5s perform better compared to other methods, as evidenced by their higher mAP@50 and mAP@70 scores. Upon comparing the performance of YOLOv9n with YOLOv10n, we noticed that YOLOv10n achieves better performance compared to its counterpart. This is because YOLOv9n struggles to detect small objects and is unable to extract contextual information due to its small receptive field. YOLOv9n also relies on non-maximum suppression, which makes it slower than YOLOv10n as it introduces additional computational overhead during inference. These issues were addressed in YOLOv10n by incorporating large-kernel convolutions and the Partial Self-Attention (PSA) module, which enhance the model's ability to perform better feature extraction and capture global context. Among these methods, the NWD method performs well compared to YOLOv9n and YOLOv10n. This is because the NWD is specifically designed for detecting tiny objects in drone images. Instead of relying solely on the IoU, the NWD models bounding boxes as 2D Gaussian distributions and computes the Wasserstein Distance (between predicted and ground-truth bounding boxes) to detect small objects. Despite its good performance, the NWD method still lags behind the proposed method. This is because the NWD lacks mechanisms for enriching the surrounding context of small objects, which is critical for distinguishing tiny objects in cluttered or complex scenes. In contrast, the proposed method utilizes the CEM to capture multiscale contextual information for detecting small objects.

Their relatively good performance is attributed to the fact that YOLOv3-spp incorporates spatial pyramid pooling (SPP), which pools features at different scales. This helps the model capture finer details of objects at various scales. This strategy improves the model's ability to detect small and dense objects. YOLOv5 achieves better results than YOLOv3-spp by further refining the spatial pyramid pooling approach and preserving spatial information across layers through the use of path aggregation techniques.

Table 2. Performance comparison of various state-of-the-art detectors.

Model	mAP@50	mAP@70
Faster-R-CNN [68]	23.74	15.32
YOLOv3-spp [69]	48.52	44.28
Cascade R-CNN [70]	42.40	32.23
RetinaNet [71]	34.12	29.42
ATSS [72]	41.75	30.17
RefineDet [73]	28.62	19.72
YOLOv5s	54.39	49.58
CenterNet [74]	40.64	29.25
NWD [67]	77.24	72.52
YOLOv9n [75]	68.41	64.19
YOLOv10n [76]	74.20	69.72
Proposed	82.10	76.23

By comparing the results in Tables 1 and 2, it is observed that the variants based on YOLOv8 in Table 1 consistently significantly outperform the methods in Table 2. This is due to reason that YOLOv8 incorporates enhanced feature pyramids and attention mechanisms that help the model maintain high-resolution feature maps and details of small targets across various scales.

The proposed framework, on the other hand, achieves superior performance compared to other state-of-the-art detectors. This is due to the following reasons: (1) YOLOv8 serves as the baseline for the proposed framework, inheriting all the benefits of YOLOv8. (2) We incorporate a CEM, which improves the model's capacity to detect small objects by broadening the receptive field of the convolutional layers through the use of atrous convolution. Moreover, this effective integration enables the model to capture more contextual information surrounding small objects, which is crucial for accurate detection in the cluttered and complex scenes.

4.6. Comparisons with Crowd Counting Methods

To evaluate the counting performance of the proposed framework, we compare its performance with other models, and the results are reported in Table 3. The models in Table 3 are specifically designed for crowd counting tasks. We observe that AMDCN [77] shows relatively low performance with high MAE and MSE values. This is due to the fact that the model consists of a fixed and limited number of columns that process the input image at restricted scales, making it struggle to extract fine-grained details in densely populated scenes. Similarly, LCFCN [20] underperforms because it relies on point-level supervision for target localization rather than employing a direct regression-based strategy for crowd counting. Among the competing methods, we observed that MTE (SFANet) [78] achieves relatively good performance. This is due to the reason that the model leverages temporal information between frames to enhance the counting accuracy. Golda et al. [78] propose strategies, TE-M20 (Temporal Extension with Many-to-One) and MTE (Merging Temporal Embeddings), to leverage the temporal information across the frames to smooth the crowd count. By comparing the performance of SFANet with its MTE-SFANet version and MRCNet with its MTE-MRCNet counterpart, we noticed that by integrating MTE with the original models, the performance is improved.

From Table 3, it is evident that the proposed framework outperforms most crowd counting methods by a significant margin; however, it still slightly lags behind CSRNet. This is because CSRNet is specifically designed for highly congested scenes and consists of a dilated CNN that uses dilated kernels to achieve larger receptive fields, which effectively

captures detailed spatial information without the need for pooling operations. However, CSRNet has one drawback compared to the proposed framework: it is based on regression, which estimates the count by inputting an image but does not provide precise localization of people in the environment. In other words, CSRNet is limited to estimating the count and does not precisely localize objects in the scene. In contrast, the proposed framework accurately localizes objects, which is crucial for crowd management.

Table 3. Performance comparison of various state-of-the-art crowd counting models.

Model	MAE	MSE
LCFCN [20]	136.90	150.60
AMDCN [77]	165.60	167.70
MSCNN [36]	58.00	75.20
StackPooling [79]	68.8 0	77.20
SwitchCNN [22]	66.50	77.80
DA-Net [80]	36.5 0	47.30
C-MTL [39]	56.70	65.90
ACSCP [81]	48.10	60.20
SFANet [82]	39.70	48.30
MRCNet [56]	46.70	58.30
TE-M2O (MRCNet) [78]	46.70	59.80
TE-M20 (SFANet) [78]	46.00	55.50
MTE-(MRCNet) [78]	44.30	56.90
MTE-(SFANet) [78]	33.20	41.80
CSRNet [21]	19.8 0	25.60
Proposed	25.42	34.73

4.7. Ablation Study

To evaluate the effect of different modules on the performance of the proposed framework, we performed an ablation study. We kept the experiment environment (training and testing samples) the same for this experiment. We evaluated six models with different configurations, as detailed in Table 4.

Table 4. Effect of CEM on model's performance.

Method	mAP@50	mAP@70
YOLOv3	48.52	44.28
YOLOv3 + CEM	52.10	46.37
YOLOv8s + SPPF	57.65	51.41
YOLOv8s + SPPF + CEM	59.28	55.62
YOLOv8s + SPP	59.64	54.44
YOLOv8s + SPP + CEM	61.12	56.42

Table 4 discusses the effect of the context enhancement module, spatial pyramid pooling fast, and spatial pyramid pooling.

From Table 4, we observed that incorporating the CEM significantly boosts the performance of all the methods in the table. This indicates the importance of capturing contextual information in target detection tasks. For example, in the case of YOLOv3, the base model achieves 48.52% with an mAP@50 and 44.28% with an mAP@70. However, we observed

that after integrating the CEM, these values increase to 52.10% and 46.37%, respectively. Similarly, for the YOLOv8s configurations, the CEM consistently enhances performance for all methods. For YOLOv8s + SPPF, the addition of the CEM raises the mAP@50 from 57.65% to 59.28% and the mAP@70 from 51.41% to 55.62%. In the case of YOLOv8s + SPP, which already achieves a high performance, the inclusion of the CEM further improves the mAP@50 from 59.64% to 61.12% and the mAP@70 from 54.44% to 56.42%. This improvement demonstrates that the CEM enables the model to leverage contextual information and helps the model to comprehend the relationships between objects and their environments.

By comparing the models using the SPPF and the standard SPP, it is observed that SPP achieves superior performance in terms of accuracy. For example, YOLOv8s + SPP achieves an mAP@50 of 59.64% and an mAP@70 of 54.44%, whereas YOLOv8s + SPPF achieves lower values with an mAP@50 of 57.65% and an mAP@70 of 51.41%. This suggests that the traditional SPP, with its more extensive multiscale feature aggregation, is better at capturing spatial information across different target sizes. Despite the high accuracy performance of SPP, SPPF is more computationally efficient compared to SPP. This is because SPPF is lighter and faster than SPP.

To thoroughly evaluate and understand the impact of different dilation rate configurations within the CEM, we conducted an ablation study using various combinations of dilation rates. The goal of this study is to determine how different dilation rate settings influence the model's performance. In this study, we also conducted the experiment using the same experimental environment, including the same training and testing samples, to ensure consistency.

For this experiment, we generated eight different configurations and the results of these configurations are reported in Table 5. From Table 5, it is noted that single dilation rates achieve lower performance than more than one dilation rate. Among the single dilation rate configuration, a dilation rate of 3 achieves good results compared to a single dilation rate of 1 and 5. This is due to the reason that dilation rate 3 optimally expand the receptive field to focus on tiny objects in the scene.

Table 5. Effect of different dilation rate configurations in the context enrichment module (CEM).

Configuration	Dilation Rate	mAP@50	mAP@70
1	$d = 1$	77.10	71.34
2	$d = 3$	79.24	73.75
3	$d = 5$	78.68	72.32
4	$d = 1, d = 3$	80.35	75.15
5	$d = 1, d = 5$	79.45	74.01
6	$d = 3, d = 5$	79.90	74.45
7 (Proposed)	$d = 1, d = 3, d = 5$	82.10	76.23
8	$d = 1, d = 3, d = 5, d = 7$	81.90	75.80

In the case of two dilation rate combinations, dilation rates of 1 and 3 (configuration 4) outperformed the other two-rate combinations, achieving an mAP@50 of 80.10 and an mAP@70 of 75.15. This result indicates that combining a standard convolution (d = 1) with a moderately expanded receptive field (using d = 3) captures both fine-grained details and broader contextual information effectively.

From this experiment, we further observed that the configuration using dilation rates 1, 3, and 5 yielded the highest performance, with an mAP@50 of 82.10 and an mAP@70 of 76.23. This suggests that combining three different scales allows the model to capture a wide range of contextual information, from fine details to broader patterns. However, adding a fourth dilation rate of 7 slightly decreased performance compared to the three-rate configuration, with an mAP@50 of 81.90 and an mAP@70 of 75.80. This suggests that

further expanding the receptive field may introduce excessive context which may decrease the focus of the model on relevant features.

To qualitatively assess the performance of the proposed framework, we present visualizations of the output of the proposed method and other baseline methods using sample frames from various scenes. These results are illustrated in the accompanying Figure 7. Figure 7 demonstrates that the proposed framework detects the targets close to the ground truth compared to the YOLOv8x and YOLOv8l models. Despite good detection performance, the proposed method also occasionally misses some detections and generates false positives. This is due to the reason that tiny objects in high-resolution scenes occupy very few pixels and lack sufficient visual detail and distinctive features, making the model confuse them with noise or background elements, leading to errors (missed detections and false positives). These errors highlight areas where the model could be further refined to improve its detection accuracy.

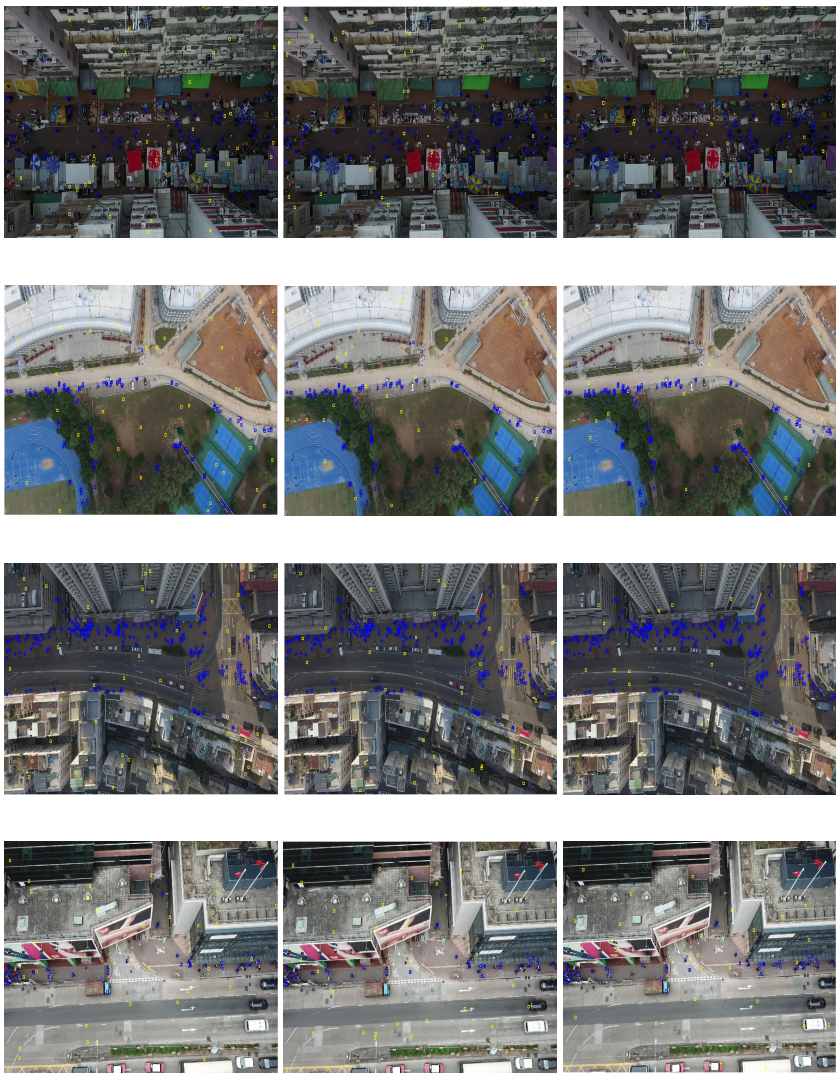


Figure 7. Visualization of the detection results by the proposed framework and other baseline methods in different scenarios. The first column represents the results of YOLOv8l, the second column shows the results of YOLOv8x, and the third column shows the results of the proposed model. Bounding boxes in the blue color represent the correct detection, and bounding boxes in the yellow color represent the false detections. (The best view is the zoomed-in view.)

5. Conclusions

In this paper, we presented an enhanced YOLOv8-based framework tailored for crowd counting in drone imagery. We introduced and effectively integrated a CEM to enhance the detection accuracy of small objects in aerial images. We assessed the performance of the proposed framework on a challenging dataset. From the experiment results, we demonstrated that the inclusion of the CEM significantly boosts the performance of the model, particularly in challenging scenarios.

In the future, we will explore more advanced context-aware modules that can further enhance the model's ability to distinguish small objects in highly cluttered and complex backgrounds. Additionally, we will incorporate attention mechanisms that may allow the model to dynamically focus on the most relevant parts of the image and boost the accuracy. Although the focus of this study is pedestrian detection for crowd counting in aerial images, we plan to extend our research to evaluate the detection of more complex and smaller objects in aerial imagery.

Author Contributions: Conceptualization, S.D.K.; Methodology, S.D.K.; Writing—original draft, S.D.K.; Writing—review & editing, F.U.R.; Supervision, A.N.A.; Funding acquisition, A.N.A. and F.U.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Custodian of the Two Holy Mosques Institute for Hajj and Umrah Research, project No. 23/113.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: The researchers extend their sincere thanks to the Custodian of the Two Holy Mosques Institute for Hajj and Umrah Research for supporting and financing this project No. 23/113, which significantly contributed to the completion of the project phases.

Conflicts of Interest: The authors have no conflicts of interest.

References

1. Li, T.; Chang, H.; Wang, M.; Ni, B.; Hong, R.; Yan, S. Crowded scene analysis: A survey. *IEEE Trans. Circuits Syst. Video Technol.* **2014**, *25*, 367–386. [CrossRef]
2. Klatt, K.; Serino, R.; Davis, E.; Grimes, J.O. Crowd-Related Considerations at Mass Gathering Events: Management, Safety, and Dynamics. In *Mass Gathering Medicine A Guide to the Medical Management of Large Events*; Cambridge University Press: Cambridge, UK, 2024; p. 268.
3. Kok, V.J.; Lim, M.K.; Chan, C.S. Crowd behavior analysis: A review where physics meets biology. *Neurocomputing* **2016**, *177*, 342–362. [CrossRef]
4. Zhu, F.; Wang, X.; Yu, N. Crowd tracking with dynamic evolution of group structures. In Proceedings of the Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part VI 13; pp. 139–154.
5. Khan, M.A.; Menouar, H.; Hamila, R. Revisiting crowd counting: State-of-the-art, trends, and future perspectives. *Image Vis. Comput.* **2023**, *129*, 104597. [CrossRef]
6. Basalamah, S.; Khan, S.D.; Felemban, E.; Naseer, A.; Rehman, F.U. Deep learning framework for congestion detection at public places via learning from synthetic data. *J. King Saud-Univ. Comput. Inf. Sci.* **2023**, *35*, 102–114. [CrossRef]
7. Wang, J.; Guo, X.; Li, Q.; Abdelmoniem, A.M.; Gao, M. SDANet: scale-deformation awareness network for crowd counting. *J. Electron. Imaging* **2024**, *33*, 043002. [CrossRef]
8. Guo, H.; Wang, R.; Zhang, L.; Sun, Y. Dual convolutional neural network for crowd counting. *Multimed. Tools Appl.* **2024**, *83*, 26687–26709. [CrossRef]
9. Chen, J.; Wang, Z. One-Shot Any-Scene Crowd Counting With Local-to-Global Guidance. *IEEE Trans. Image Process.* **2024** . [CrossRef]
10. Tripathy, S.K.; Srivastava, S.; Bajaj, D.; Srivastava, R. A Novel cascaded deep architecture with weak-supervision for video crowd counting and density estimation. *Soft Comput.* **2024**, *28*, 8319–8335. [CrossRef]
11. Alhawsawi, A.N.; Khan, S.D.; Ur Rehman, F. Crowd Counting in Diverse Environments Using a Deep Routing Mechanism Informed by Crowd Density Levels. *Information* **2024**, *15*, 275. [CrossRef]
12. Gao, M.; Souri, A.; Zaker, M.; Zhai, W.; Guo, X.; Li, Q. A comprehensive analysis for crowd counting methodologies and algorithms in Internet of Things. *Clust. Comput.* **2024**, *27*, 859–873. [CrossRef]
13. Chavan, R.; Rani, G.; Thakkar, P.; Dhaka, V.S. CrowdDCNN: Deep convolution neural network for real-time crowd counting on IoT edge. *Eng. Appl. Artif. Intell.* **2023**, *126*, 107089. [CrossRef]

14. Ptak, B.; Pieczyński, D.; Piechocki, M.; Kraft, M. On-board crowd counting and density estimation using low altitude unmanned aerial vehicles—Looking beyond beating the benchmark. *Remote Sens.* **2022**, *14*, 2288. [CrossRef]
15. Nag, S.; Khandelwal, Y.; Mittal, S.; Mohan, C.K.; Qin, A.K. ARCN: A real-time attention-based network for crowd counting from drone images. In Proceedings of the 2021 IEEE 18th India Council International Conference (INDICON), Guwahati, India, 19–21 December 2021; pp. 1–6.
16. Bakour, I.; Bouchali, H.N.; Allali, S.; Lacheheb, H. Soft-CSRNet: Real-time dilated convolutional neural networks for crowd counting with drones. In Proceedings of the 2020 2nd International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH), Boumerdes, Algeria, 9–10 February 2021; pp. 28–33.
17. Elharrouss, O.; Almaadeed, N.; Abualsaud, K.; Al-Ali, A.; Mohamed, A.; Khattab, T.; Al-Maadeed, S. Drone-SCNet: Scaled cascade network for crowd counting on drone images. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *57*, 3988–4001. [CrossRef]
18. Peng, T.; Li, Q.; Zhu, P. Rgb-t crowd counting from drone: A benchmark and mmccn network. In Proceedings of the Asian Conference on Computer Vision, Kyoto, Japan, 30 November–4 December 2020.
19. Liu, Z.; He, Z.; Wang, L.; Wang, W.; Yuan, Y.; Zhang, D.; Zhang, J.; Zhu, P.; Van Gool, L.; Han, J.; et al. VisDrone-CC2021: The vision meets drone crowd counting challenge results. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 2830–2838.
20. Laradji, I.H.; Rostamzadeh, N.; Pinheiro, P.O.; Vazquez, D.; Schmidt, M. Where are the blobs: Counting by localization with point supervision. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 547–562.
21. Li, Y.; Zhang, X.; Chen, D. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1091–1100.
22. Babu Sam, D.; Surya, S.; Venkatesh Babu, R. Switching convolutional neural network for crowd counting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5744–5752.
23. Wang, B.; Liu, H.; Samaras, D.; Nguyen, M.H. Distribution matching for crowd counting. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1595–1607.
24. Wang, G.; Chen, Y.; An, P.; Hong, H.; Hu, J.; Huang, T. UAV-YOLOv8: A small-object-detection model based on improved YOLOv8 for UAV aerial photography scenarios. *Sensors* **2023**, *23*, 7190. [CrossRef] [PubMed]
25. Yi, H.; Liu, B.; Zhao, B.; Liu, E. Small object detection algorithm based on improved YOLOv8 for remote sensing. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2023**, *17*, 1734–1747. [CrossRef]
26. Ma, M.; Pang, H. SP-YOLOv8s: An improved YOLOv8s model for remote sensing image tiny object detection. *Appl. Sci.* **2023**, *13*, 8161. [CrossRef]
27. Zhai, X.; Huang, Z.; Li, T.; Liu, H.; Wang, S. YOLO-Drone: an optimized YOLOv8 network for tiny UAV object detection. *Electronics* **2023**, *12*, 3664. [CrossRef]
28. Chan, A.B.; Vasconcelos, N. Counting people with low-level features and Bayesian regression. *IEEE Trans. Image Process.* **2011**, *21*, 2160–2177. [CrossRef]
29. Chen, K.; Loy, C.C.; Gong, S.; Xiang, T. *Feature Mining for Localised Crowd Counting*; BMVC: Glasgow, UK, 2012; Volume 1, p. 3.
30. Wang, Y.; Lian, H.; Chen, P.; Lu, Z. Counting people with support vector regression. In Proceedings of the 2014 10th International Conference on Natural Computation (ICNC), Xiamen, China, 19–21 August 2014; pp. 139–143.
31. Saqib, M.; Khan, S.D.; Blumenstein, M. Texture-based feature mining for crowd density estimation: A study. In Proceedings of the 2016 International Conference on Image and Vision Computing New Zealand (IVCNZ), Palmerston North, New Zealand, 21–22 November 2016; pp. 1–6.
32. Zhang, Y.; Zhou, D.; Chen, S.; Gao, S.; Ma, Y. Single-image crowd counting via multi-column convolutional neural network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 589–597.
33. Boominathan, L.; Kruthiventi, S.S.; Babu, R.V. Crowdnet: A deep convolutional network for dense crowd counting. In Proceedings of the 24th ACM International Conference on Multimedia, Amsterdam, The Netherlands, 15–19 October 2016; pp. 640–644.
34. Ranjan, V.; Le, H.; Hoai, M. Iterative crowd counting. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 270–285.
35. Sindagi, V.A.; Patel, V.M. A survey of recent advances in cnn-based single image crowd counting and density estimation. *Pattern Recognit. Lett.* **2018**, *107*, 3–16. [CrossRef]
36. Zeng, L.; Xu, X.; Cai, B.; Qiu, S.; Zhang, T. Multi-scale convolutional neural networks for crowd counting. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 465–469.
37. Cao, X.; Wang, Z.; Zhao, Y.; Su, F. Scale aggregation network for accurate and efficient crowd counting. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 734–750.
38. Babu Sam, D.; Sajjan, N.N.; Venkatesh Babu, R.; Srinivasan, M. Divide and grow: Capturing huge diversity in crowd images with incrementally growing cnn. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3618–3626.

39. Sindagi, V.A.; Patel, V.M. Cnn-based cascaded multi-task learning of high-level prior and density estimation for crowd counting. In Proceedings of the 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 29 August–1 September 2017; pp. 1–6.
40. Idrees, H.; Tayyab, M.; Athrey, K.; Zhang, D.; Al-Maadeed, S.; Rajpoot, N.; Shah, M. Composition loss for counting, density map estimation and localization in dense crowds. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 532–546.
41. Xiong, F.; Shi, X.; Yeung, D.Y. Spatiotemporal modeling for crowd counting in videos. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5151–5159.
42. Hu, Y.; Jiang, X.; Liu, X.; Zhang, B.; Han, J.; Cao, X.; Doermann, D. NAS-Count: Counting-by-Density with Neural Architecture Search. *arXiv* **2020**, arXiv:2003.00217 .
43. Zhai, W.; Gao, M.; Li, Q.; Jeon, G.; Anisetti, M. FPANet: feature pyramid attention network for crowd counting. *Appl. Intell.* **2023**, *53*, 19199–19216. [CrossRef]
44. Wang, T.; Zhang, T.; Zhang, K.; Wang, H.; Li, M.; Lu, J. Context attention fusion network for crowd counting. *Knowl. Based Syst.* **2023**, *271*, 110541. [CrossRef]
45. Du, Z.; Shi, M.; Deng, J.; Zafeiriou, S. Redesigning multi-scale neural network for crowd counting. *IEEE Trans. Image Process.* **2023**, *32*, 3664–3678. [CrossRef]
46. Wang, R.; Hao, Y.; Hu, L.; Chen, J.; Chen, M.; Wu, D. Self-supervised learning with data-efficient supervised fine-tuning for crowd counting. *IEEE Trans. Multimed.* **2023**, *25*, 1538–1546. [CrossRef]
47. Zhang, C.; Zhang, Y.; Li, B.; Piao, X.; Yin, B. CrowdGraph: Weakly supervised crowd counting via pure graph neural network. *ACM Trans. Multimed. Comput. Commun. Appl.* **2024**, *20*, 1–23. [CrossRef]
48. Yan, L.; Zhang, L.; Zheng, X.; Li, F. Deep feature network with multi-scale fusion for highly congested crowd counting. *Int. J. Mach. Learn. Cybern.* **2024**, *15*, 819–835. [CrossRef]
49. Küchhold, M.; Simon, M.; Eiselein, V.; Sikora, T. Scale-adaptive real-time crowd detection and counting for drone images. In Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018; pp. 943–947.
50. Zhang, B.; Du, Y.; Zhao, Y.; Wan, J.; Tong, Z. I-MMCCN: Improved MMCCN for RGB-T crowd counting of drone images. In Proceedings of the 2021 7th IEEE International Conference on Network Intelligence and Digital Content (IC-NIDC), Beijing, China, 17–19 November 2021; pp. 117–121.
51. Castellano, G.; Cotardo, E.; Mencar, C.; Vessio, G. Density-based clustering with fully-convolutional networks for crowd flow detection from drones. *Neurocomputing* **2023**, *526*, 169–179. [CrossRef]
52. Chen, J.; Xiu, S.; Chen, X.; Guo, H.; Xie, X. Flounder-Net: An efficient CNN for crowd counting by aerial photography. *Neurocomputing* **2021**, *420*, 82–89. [CrossRef]
53. Castellano, G.; Castiello, C.; Mencar, C.; Vessio, G. Crowd detection in aerial images using spatial graphs and fully-convolutional neural networks. *IEEE Access* **2020**, *8*, 64534–64544. [CrossRef]
54. Bai, H.; Wen, S.; Gary Chan, S.H. Crowd counting on images with scale variation and isolated clusters. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, Seoul, Republic of Korea, 27–28 October 2019.
55. Zhao, L.; Bao, Z.; Xie, Z.; Huang, G.; Rehman, Z.U. A point and density map hybrid network for crowd counting and localization based on unmanned aerial vehicles. *Connect. Sci.* **2022**, *34*, 2481–2499. [CrossRef]
56. Bahmanyar, R.; Vig, E.; Reinartz, P. MRCNet: Crowd counting and density map estimation in aerial and ground imagery. *arXiv* **2019**, arXiv:1909.12743 .
57. Husman, M.A.; Albattah, W.; Abidin, Z.Z.; Mustafah, Y.M.; Kadir, K.; Habib, S.; Islam, M.; Khan, S. Unmanned aerial vehicles for crowd monitoring and analysis. *Electronics* **2021**, *10*, 2974. [CrossRef]
58. Gu, S.; Lian, Z. A unified multi-task learning framework of real-time drone supervision for crowd counting. *arXiv* **2022**, arXiv:2202.03843.
59. Almagbile, A. Estimation of crowd density from UAVs images based on corner detection procedures and clustering analysis. *Geo Spat. Inf. Sci.* **2019**, *22*, 23–34. [CrossRef]
60. Zhu, J.; Hu, T.; Zheng, L.; Zhou, N.; Ge, H.; Hong, Z. YOLOv8-C2f-Faster-EMA: An Improved Underwater Trash Detection Model Based on YOLOv8. *Sensors* **2024**, *24*, 2483. [CrossRef]
61. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H. Designing network design strategies through gradient path analysis. *arXiv* **2022**, arXiv:2211.04800.
62. Zhang, Z. Drone-YOLO: an efficient neural network method for target detection in drone images. *Drones* **2023**, *7*, 526. [CrossRef]
63. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 390–391.
64. Wen, L.; Du, D.; Zhu, P.; Hu, Q.; Wang, Q.; Bo, L.; Lyu, S. Detection, tracking, and counting meets drones in crowds: A benchmark. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 7812–7821.
65. Zhu, P.; Peng, T.; Du, D.; Yu, H.; Zhang, L.; Hu, Q. Graph regularized flow attention network for video animal counting from drones. *IEEE Trans. Image Process.* **2021**, *30*, 5339–5351. [CrossRef]

66. Zhang, C.; Li, H.; Wang, X.; Yang, X. Cross-scene crowd counting via deep convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 833–841.
67. Xu, C.; Wang, J.; Yang, W.; Yu, H.; Yu, L.; Xia, G.S. Detecting tiny objects in aerial images: A normalized Wasserstein distance and a new benchmark. *ISPRS J. Photogramm. Remote Sens.* **2022**, *190*, 79–93. [CrossRef]
68. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
69. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
70. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
71. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22—29 October 2017; pp. 2980–2988.
72. Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; Li, S.Z. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9759–9768.
73. Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; Li, S.Z. Single-shot refinement neural network for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2018; pp. 4203–4212.
74. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. Centernet: Keypoint triplets for object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27–28 October 2019; pp. 6569–6578.
75. Wang, C.Y.; Yeh, I.H.; Liao, H.Y.M. Yolov9: Learning what you want to learn using programmable gradient information. *arXiv* **2024**, arXiv:2402.13616.
76. Wang, A.; Chen, H.; Liu, L.; Chen, K.; Lin, Z.; Han, J.; Ding, G. Yolov10: Real-time end-to-end object detection. *arXiv* **2024**, arXiv:2405.14458.
77. Deb, D.; Ventura, J. An aggregated multicolumn dilated convolution network for perspective-free counting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Salt Lake City, UT, USA, 18–22 June 2018; pp. 195–204.
78. Golda, T.; Krüger, F.; Beyerer, J. Temporal Extension for Encoder-Decoder-based Crowd Counting Approaches. In Proceedings of the 2021 17th International Conference on Machine Vision and Applications (MVA), Virtual, 25–27 July 2021; pp. 1–5.
79. Huang, S.; Li, X.; Cheng, Z.Q.; Zhang, Z.; Hauptmann, A. Stacked pooling: Improving crowd counting by boosting scale invariance. *arXiv* **2018**, arXiv:1808.07456.
80. Zou, Z.; Su, X.; Qu, X.; Zhou, P. DA-Net: Learning the fine-grained density distribution with deformation aggregation network. *IEEE Access* **2018**, *6*, 60745–60756. [CrossRef]
81. Shen, Z.; Xu, Y.; Ni, B.; Wang, M.; Hu, J.; Yang, X. Crowd counting via adversarial cross-scale consistency pursuit. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2018; pp. 5245–5254.
82. Zhu, L.; Zhao, Z.; Lu, C.; Lin, Y.; Peng, Y.; Yao, T. Dual path multi-scale fusion networks with attention for crowd counting. *arXiv* **2019**, arXiv:1902.01115.