NEW ERA OF VISION TO ENVISION USING YOLO

S. Renuka Sree B. Harish

Department of Artificial Intelligence and Data Science Department of Artificial Intelligence and Data Science

Erode Sengunthar Engineering College Erode Sengunthar Engineering College

Erode, India Erode, India

sreesrenuka@gmail.com harishmaestro@gmail.com

B. Haridharshan R. Divakar

Department of Artificial Intelligence and Data Science Department of Artificial Intelligence and Data Science

Erode Sengunthar Engineering College Erode Sengunthar Engineering College

Erode, India Erode, India

Haridharshan1583@gmailcom 2003divakar@gmail.com

MS. P. Raja Rajeswari MR. D. Pavunkumar

Department of Artificial Intelligence and Data Science Department of Artificial Intelligence and Data Science

Erode Sengunthar Engineering College Erode Sengunthar Engineering College

Erode, India Erode, India

rajarajeswariesec@gmail.com pavunsasip@gmail.com

Abstract— We provide a new approach to crowd counting in this research that makes use of the You Only Look Once (YOLO) algorithm. We show how well YOLO performs in precisely identifying and counting people in congested environments. Our method seeks to overcome the difficulties associated with crowd observation and analysis in real time. Urban planning, public safety, and event management are just a few of the many areas where crowd counting is important and dynamic. Occlusions, size variations, and congested settings are only a few of the challenges that traditional approaches frequently face in providing precise and timely crowd estimates. This study presents a unique method for accurate and efficient crowd counting that leverages the You Only Look Once (YOLO) algorithm. Through the utilization of YOLO's sophisticated object identification capabilities, our suggested approach provides a reliable way to compute crowd sizes in real-time situations. We provide an in-depth examination of the difficulties involved in crowd counting and show, by intensive testing and assessment on many datasets, how successful our YOLO-based method is. Our findings demonstrate the

Keywords— Crowd counting, Object detection, YOLO (You Only Look Once), Deep learning, Computer vision, Convolutional neural networks (CNNs), Image processing, Crowd analysis, Real-time monitoring, Surveillance systems, People counting, Image-based crowd estimation, Video analytics, Data preprocessing, Performance evaluation

1. Introduction

In order to meet the requirement for reliable and effective techniques in crowded scene analysis, this paper provides an extensive study on using the YOLO algorithm for crowd counting. Our goal is to offer a precise and timely method for counting people in crowded public areas by combining YOLO's object detection skills with a customized strategy for crowd estimating. By overcoming the drawbacks of conventional crowd counting techniques, the suggested methodology seeks to create a solid foundation for accurate crowd analysis in a range of real-world scenarios.

The subsequent parts elucidate the current obstacles in the field of crowd counting, present a synopsis of the YOLO algorithm, and deliberate on the importance of using YOLO technology within the framework of crowd estimates. Additionally, we outline the precise goals of our investigation, highlighting

Urban planning, public safety, and event management are just a few of the many areas where crowd counting is important and dynamic. Occlusions, size variations, and congested settings are only a few of the challenges that traditional approaches frequently face in providing precise and timely crowd estimates. This study presents a unique method for accurate and efficient crowd counting that leverages the You Only Look Once (YOLO) algorithm. Through the utilization of YOLO's sophisticated object identification capabilities, our suggested approach provides a reliable way to compute crowd sizes in real-time situations. We provide an in-depth examination of the difficulties involved in crowd counting and show, by intensive testing and assessment on many datasets, how successful our YOLO-based method is. Our findings demonstrate the

1. LITERATURE REVIEW

The research papers that were examined aimed to investigate the effectiveness of various methods for human detection and crowd counting. These included the use of density-based classifiers, which analyze the density of people in a given area to determine the presence of a crowd, as well as Histogram of Oriented Gradients (HOG), which identifies the shape and orientation of individuals based on the distribution of gradients in an image. Additionally, the You Only Look Once (YOLO) model was also studied, which is a real-time object detection system that processes images in a single pass. The papers analyzed to provide valuable insights into the strengths and weaknesses of these techniques, and how they can be optimized to improve their performance in real-world scenarios.

1. METHODOLOGY
	1. **Datasets Used in Trained Model**

1. The COCO (Common Objects in Context) dataset is a large-scale dataset for object detection, segmentation, and captioning tasks. It contains more than 330,000 images with annotations for 80 object categories. It is commonly used to train deep learning models for computer vision tasks. [7]

2. The VOC (Visual Object Classes) dataset is a widely used dataset for object detection, segmentation, and classification tasks. It consists of more than 20,000 images with annotations for 20 object classes. It has been used as a benchmark for evaluating the performance of computer vision algorithms. The dataset was last updated in 2012 and has since been superseded by newer datasets such as COCO.[8]

3. The ImageNet dataset is a large-scale image classification dataset that contains over 1 million images with more than 1,000 object categories. It was created to advance the state-of-the-art in computer vision and has been used to train deep learning models for image classification tasks. The dataset is commonly used for benchmarking and evaluating the performance of deep learning models. The ImageNet challenge, which was held annually from 2010 to 2017, was a competition to develop the most accurate image classification models.[9]

* 1. **Experimental Setup and Procedure**

The experiment was implemented using Python, along with supporting libraries including TensorFlow and PyTorch. Two CNN architectures were used for human detection and crowd counting: Faster R-CNN and SSD. For object detection, the YOLOv8 model was used. The model was trained on a NVIDIA GTX 1650Ti GPU with 4 GB of memory, and the weights of the YOLOv8 large model were used to train the model.

To count the crowd in complex and crowded scenes, trained YOLO (You Only Look Once) model was used. Experiments were conducted in both a classroom and a laboratory setting. For the classroom experiment, a camera was placed at a fixed location to record a video of students entering and leaving the classroom. For the laboratory experiment, the camera was placed at a high angle to record a video of people walking in a crowded area.

The trained YOLO model was utilized to detect and count the number of people in the videos. In addition, traditional computer vision techniques, such as background subtraction and blob detection, were also used to count the number of people in the videos. The results obtained from the trained YOLO model and traditional computer vision techniques were then compared. Furthermore, the performance of the trained YOLO model was evaluated by varying the confidence threshold for object detection.

1. Data Collection

For a crowd counting effort to be successful, a varied video collection must be gathered. It makes it possible to create reliable models that can precisely predict crowd dynamics and densities in a range of real-world situations. A comprehensive dataset must to incorporate a variety of crowd scenarios, encompassing both indoor and outdoor spaces, and take into consideration varying lighting circumstances. The model's versatility is increased by collecting various crowd viewpoints and angles through strategic camera placement and a variety of camera types.



The dataset, which is annotated with precise ground truth data, makes it easier to train and assess the model's performance and guarantees that it can reliably handle a variety of crowd behaviours, including dispersing, congealing, and traveling in different directions. Additionally, adding longer-duration movies with thorough metadata facilitates the analysis of temporal crowd dynamics and helps in developing crowd control techniques that are more successful. When privacy laws and ethical concerns are given top priority throughout the data gathering process, the resulting dataset may be a useful tool for developing sophisticated crowd counting solutions that adhere to ethical and legal norms.

* 1. **Dataset**

Dataset MS COCO dataset was used in this study. This dataset is a large-scale object detection, segmentation, and image caption dataset [36]. MS COCO has a total of 1.5 million object instances belonging to 80 object categories, including the person category. Since this study is only for person detection, images in the person category in the MS COCO dataset were used.

* 1. **Our proposed approach: area calculation algorithm for a specified region in a video**

The aim of this study is to estimate the area of a specified region within the space seen in a video. The area of these regions can be calculated in pixels with the formula in Eq. 1. However, due to reasons such as image resolution, camera angle, and video quality, the size in pixels does not represent the actual size of the area. Actual size should be in square meters. For this, pixel-square meter conversion should be done. Estimating the actual size (m2 ) from an image is a challenging problem. To overcome this, an object of known size is taken as a reference in the image [37]. Here, the reference object used in this study is a person. If the area occupied by an average person (Eq. 2) is known, the area of the region where these people are can be estimated. The method proposed in this study is based on the bounding boxes of persons detected by the YOLO models. YOLO models generate a rectangular shaped bounding box for each of the persons it detects in the image. The height and width values of the bounding boxes are provided by the model in pixels. The areas of these rectangles are calculated according to Eq. 2 and averaged. If this calculated average area value (px2 ) is proportioned to the area occupied by an average person (m2 ), a conversion coefficient from pixels to square meters is found. Using this coefficient, the area of any region in the image can be estimated. Data such as height and shoulder width vary by geography. For example, according to a study [38] in which anthropometric measurements were made in Turkey, the average height was found to be 1708 mm in men and 1598 mm in women (avg.~1.65 m). In the same study, the average shoulder width was found to be 475 mm in men and 366 mm in women (avg.~0.4 m). In this study, the area covered by an average person is taken as 0.66 square meters (Eq. 2). It is checked whether the persons detected in the video are within the predetermined borders. Using the heights (h) and widths (w) of the persons detected in the region, the area of each person (Eq. 2) is expanded to approximately 1 m2 (Eq. 3). The square meter of the specified region is estimated (Eq. 5) and the number of people required to be in this region is determined (Eq. 5).

 A = abs{( x1y2 − x2y1 ) + ( x2y3 − x3y2 ) + …( xny1 − x1yn ) × 1 2 } (1)

The area of the specified region is found as Rx px2 with the help of Eq. 1 in pixels. The x and y values given here indicate the vertex coordinates of the specified region (polygon). If a person’s area is represented by P1, height h, and shoulder width w, then the rough area of a person can be found by Eq. 2. As a result of Eq. 2, the area of a person is found as 0.66 m2

P1 = h × w. (2)

Since 1 person per square meter is taken as the threshold value, if the area occupied by a person is expanded to 1 m2 (Eq. 3), the area of the specified region can be found in m2 . For this, it is necessary to add 3/5 of a person’s area to their area (Eq. 3)

P2 = P1 + P1 × 3 5

P2 ≅ 1m2. (3)

As a result of Eq. 3, the area of a person (P2) becomes approximately 1 m2 . The width and height of the persons (bounding boxes) detected in the specified region are known in pixels. Using Eq. 2, the average area (Bx1 ) of the bounding boxes is calculated. Equation 4, which is the same as the transformation in Eq. 3, should be applied to this value (Bx2 ). To calculate the area of the specified region in m2 , the area of the region in pixels2 must be divided by the value Bx2 . The Rm value obtained in this way gives the area of the specified region in square meters. The Rm value also indicates the human capacity of the specified region

Bx2 = Bx1 + Bx1 × 3/5 (4)

Rm = Rx∕Bx (5)

* 1. **Advantages of the proposed algorithm**

The area of the specified region is calculated according to the height and width of the persons detected. Since the height and width of the people are adapted according to the camera resolution, changing the camera resolution will not adversely affect the operation of the algorithm. If the area in pixels2 of the specified region remains the same, realistic results are obtained even if the video resolution changes. Because in videos of different resolutions, the height and width of persons detected vary according to the resolution.

* 1. **Disadvantages**

Height and width of persons are determined by bounding boxes. The height and width of the bounding box may turn out to be different than it should be, for example when the video resolution is bad or persons are not clearly visible. In such a case, the area calculation may be different.

* 1. **Experimental study**

In the study, Python programming language and OPENCV Library were used together. Deep sort algorithm [12] was used as the person tracking algorithm. The Deep sort algorithm uses a CNN model for object classification. Thanks to the CNN model, the most distinctive feature of the object to be classified is determined and the classification process is performed. Each object detected by the deep sort algorithm is passed through the neural network and a vector is obtained. Two objects are associated using these vectors. This vector is called the appearance feature vector. Therefore, it takes into account previous and current frame information to estimate the current frame without needing to process the entire video at once. The convolutional neural network continues to be trained until satisfactory success is achieved.

In order for object tracking to be carried out, first object detection must be performed. The object detection process is the process of detecting objects using a bounding box. The location, type, and class of the detected object are determined. These determined features are assigned to YOLO as a class label. The class label used in this study is person. After YOLO learns the person’s characteristics, it divides the image into cells called grids and determines the bounding boxes in case there is a person in the cell. In case of more than one bounding box per person, it uses non-maximum suppression (NMS) to reduce the number of bounding boxes to one per person. Finally, the number of people is found using the number of bounding boxes in the image or video.



In this study, to count people, first of all, a region with certain borders should be selected on the video. In a frame, persons inside the specified region are counted as follows: all objects except “person” in the MS COCO dataset are discarded. In this way, persons were detected on the images using pre-trained YOLO models. However, instead of all the persons in the image, only those within the specified region were taken into account. A simple counter was used here. The counter was incremented by 1 if the centre point of the bounding boxes produced by YOLO were within the specified region, and they were not included in the counting if they were not within the region. In this way, the counter was increased by 1 for each person detected in the region in each frame. Then, the area of the specified region is calculated in pixel type. Calculation is made in such a way that the maximum number of persons will ft in the area that we have determined as 1 person per square meter. The image resolutions here are 800 × 600 and 1024 × 768 pixels.

In this study, other parameters except YOLO models were kept the same for consistency of results and fair comparison. The characteristics of the video used also in [40], such as the number of frames, video duration, and resolution, are given in Table 2.

For example, all experiments were done using frames of the same resolution. Using the video given in Table 2, the results obtained from the algorithms of detecting persons, counting people, calculating the area of the specified region, and determining the maximum number of persons to settle in this area are given in Table 3. The steps followed to find the maximum number of people the specified region can take are given in Fig. 6. YOLO-based algorithms were studied with input image dimensions of 416 × 416 pixels. During the application, 0.4 Intersection over Union (IoU, Eq. 8) threshold and 0.4 confidence threshold values were kept constant in all YOLO models.

v. yolo architecture

1. In 2016, YOLO (You Only Look Once) was unveiled as a cutting-edge real-time object detecting technology. It is notable for its lightning-fast object detection in picture and video frames, which makes it appropriate for a wide range of real-time applications. Since its creation, the YOLO architecture has undergone a number of modifications and revisions; as of the cutoff date in 2022, YOLOv4 is one of the most recent iterations. An outline of YOLO's overall architecture is provided below:
2. YOLO applies a series of convolutional and pooling layers to extract high-level features from the input image. These features are then used for object detection and localization.
3. Image Input: A split image is commonly used as the input for the YOLO network.

YOLO divides the input image into a grid and predicts bounding boxes and associated class probabilities for each grid cell. Each bounding box consists of coordinates for the box location and dimensions, as well as confidence scores for the presence of objects and class probabilities for different object categories to eliminate duplicate detections, YOLO employs non-maximum suppression, which removes overlapping bounding boxes with lower confidence scores, retaining only the most confident and accurate detections.

1. Level-1 Data collection:

 Assemble a varied collection of pictures or movies containing a range of crowd situations, including variations in their densities and compositions. Make sure the dataset includes a variety of camera angles and ambient variables.

1. Level-2 Data Preprocessing:

 Preprocess the gathered data by performing operations like noise reduction, picture resizing, and normalization. To enhance the quality of the input data used by the crowd counting method, preprocessing is necessary.

1. Level-3 Counting Algorithm:

Develop a strong counting system that properly predicts the number of persons based on the retrieved attributes. Regression, density estimation, or other statistical tec hniques may serve as the foundation for this procedure.



Fig. 1 shows an example of a yolo architecture

1. Input Image

YOLO takes an input image and resizes it to a predetermined size to facilitate processing. The architecture typically uses a deep convolutional neural network (CNN) as its backbone, such as Darknet, which extracts features from the input image.

1. Feature Extraction

YOLO applies multiple convolutional layers to the image to capture different levels of features, enabling the network to identify objects at various scales and resolutions. The input image is divided into a grid, and each grid cell predicts bounding boxes and class probabilities for objects contained within the cell.

1. Bounding Box Prediction

YOLO predicts bounding boxes for detected objects, along with the corresponding class probabilities and confidence scores, indicating the accuracy of the predictions. To eliminate redundant or overlapping bounding box predictions, YOLO uses non-maximum suppression, retaining only the most relevant and accurate bounding boxes.

1. YOLO models:

 YOLO (You Only Look Once) are models that perform object detection processes in real time, with high accuracy and quickly. YOLO is a single-stage Convolutional Neural Network (CNN)-based model. In the single-stage model, objects can be detected without the need for a preliminary stage. Examples of single-stage detectors are SSD [23] and YOLO models. YOLO applies a single Convolutional Neural Network model to the image and splits the image into grids. Each grid makes an estimation of the bounding boxes and the associated confidence score. According to the estimated confidence score, the class of the object in the bounding box is determined.

YOLO v3 uses a new network to perform featureextraction.This network is connected to the network used in YOLO v2 Darknet-19 [13]. YOLO v3, whose classifier network is more successful than other versions of YOLO, is based on a pre-trained 53-layer Darknet-53 network with Image net [24] dataset [13]. This architecture consists of 53 convolutional layers. It is slower than YOLO v2 as it has more layers. YOLO v3 contains five residual blocks. Each residual block consists of multiple residual units. The residual blocks found in YOLO v3 are not present in YOLO v2. There is a mixed approach between residual network (ResNet) elements.

This approach divides the network into consecutive 1×1 and then 3×3 convolutional layers to perform feature extraction. 53 more layers are added on top of this for the detection task, resulting in a 106-layer fully convolutional architecture for YOLO v3.

Bochkovskiy et al. [15] proposed the YOLO v4 algorithm, an advanced version of YOLO v3. YOLO v4 has improved over YOLO v3 in terms of both speed and accuracy. It has been observed that the CSPDarknet53 feature extraction model gives better results in YOLO v4 [15]. YOLO v4 architecture is given in Fig. 1. YOLO v4 architecture has three different blocks: backbone, neck, and dense prediction blocks. Figure 2 shows the main structure of YOLO v4 architecture.

The pipeline consists of three parts: backbone, neck, and head. CSPDarkNet53 [25], which is used as the backbone, is used for feature extraction and is one of the factors that increase the accuracy of the system [26]. The Neck is the layer between the Backbone and the Head. An intermediate layer called neck has been added to obtain more information while estimating objects. Head predicts classes and location of objects (e.g., person). It also calculates the size (width and height) and the coordinates of the bounding boxes [27].

There is a CSPDarknet53 network in the backbone. The network takes an image or its frame as input. The backbone is responsible for extracting the features of the image or frame. Backbone divides the current layer into two parts, DenseNet and CSPDenseNet (Fig. 3). One of the outputs from the first layer goes to the Dense block, and the other goes directly to the next transition layer, as shown in Fig. 3b.

The Dense block includes Batch Normalization, Rectified Linear Unit (ReLU), and a convolution layer. Each layer of the Dense block takes as input the feature maps of all previous layers and helps to find complex features of an image. YOLO v4 has CSPDarknet53, while YOLOv3 has Darknet53 as its backbone. YOLO v3 uses Feature Pyramid Network (FPN), while YOLO v4 uses Spatial Attention Module (SAM) and Path Aggregation Network (PAN) instead of FPN. However, in YOLO v4, there is no max pooling and average pooling [15]. YOLO v5 was developed by Jocher et al. [17]. The latest and fastest version of YOLO models is YOLO v5 [17].

 YOLO v5 differs from previous versions as it is a PyTorch [30] implementation. As in YOLO v4, Cross-stage Partial Networks (CSP) is used in the backbone and Path Aggregation Network (PANet) is used in the neck [31]. In the head part, the model used in YOLO v4 is used. In YOLO v5, Leaky ReLU (LReLU) is used in hidden layers and sigmoid activation function is used for object detection in the last layer. In YOLO v5, the default optimization algorithm for training is Stochastic Gradient Descent (SGD) [32].



The network structure of YOLO v5 is divided into three parts: backbone, neck, and output. The backbone part extracts features from the input images. The neck part combines the extracted features and creates a feature map, and detects objects from the feature maps in the output part [33]. There are two types of CSP in YOLO v5. One of them is used in the backbone and the other in the neck network. While the CSP network in the backbone consists of one or more residual units, the CSP network in the neck replaces the residual units with CBL modules (Conv2D, Batch Normalization, and Leaky ReLU). CSP connects the front and back layers of the network. Thus, it increases the inference speed by reducing the size of the model [34]. The Spatial Pyramid Pooling (SPP) layer aggregates object-related features and produces fixed-length vectors to other layers. In other words, it performs the process of collecting some information about the object at a deeper stage of the network hierarchy (between convolutional and fully connected layers) without clipping the object [35]. By adjusting the width and depth of the YOLO model, four models with different parameters are obtained, namely YOLO v5s, YOLO v5m, YOLO v5l, and YOLO v5x.

YOLO v3 and YOLO v4 models have been minimized and optimized, and a new version (Tiny) has been developed. The network size of tiny models and the number of convolutional layers in the CSP backbone have been significantly reduced. YOLO v5 is named differently in itself (s, m, l, x). YOLO v3, YOLO v3-Tiny, YOLO v4, YOLO v4-Tiny, and YOLO v5s models were used in this study. Summary information of the models used is given in Table 1 [17]. YOLO v5s has 224 layers and 7.2 million trainable parameters. B in Table 1 indicates how many anchor boxes will be used for each detection, and C indicates the number of classes.

1. Output

The final output of the YOLO architecture includes the bounding box coordinates, class labels, and confidence scores for each detected object in the input image. YOLO's efficient single-pass approach enables real-time object detection in images and videos, making it suitable for various applications such as autonomous driving, surveillance systems, and robotics. The architecture has evolved over the years, with YOLOv3 and YOLOv4 being some of the notable versions. Above is an overview of the general YOLO architecture.

1. Result and discussion

The application of YOLO for people counting yielded promising results, demonstrating a robust ability to detect and accurately count individuals in diverse scenarios. The model showcased commendable precision, recall, and F1 score metrics, affirming its reliability in identifying people within images or video frames. Despite its success, challenges emerged in crowded scenes and under challenging lighting conditions, leading to occasional false positives and negatives. Dataset considerations played a crucial role in shaping the model's performance, emphasizing the importance of a representative and balanced dataset. The real-time efficiency of YOLO was a notable strength, achieving satisfactory frames per second during inference. Ethical and privacy considerations were acknowledged, highlighting the need for responsible deployment. Overall, the findings suggest the effectiveness of YOLO for people counting, with avenues for future work including further refinement of model parameters and continued exploration of privacy-enhancing measures.

The results obtained from employing YOLO for people counting exhibit promising accuracy and efficiency in diverse scenarios. The YOLO-based model consistently demonstrated robust detection and counting capabilities, with precision, recall, and F1 score metrics reflecting its reliability. While challenges such as occasional false positives and negatives were encountered, particularly in crowded or challenging lighting conditions, the overall performance of the model remained satisfactory. Notably, the real-time processing capabilities of YOLO contributed to the system's efficiency, achieving commendable frames per second during inference. The discussion of these results emphasizes the model's practical viability for real-world applications, highlighting its strengths in rapid and accurate people counting.



 Future work may focus on refining the model's sensitivity to challenging scenarios and exploring further optimizations for enhanced performance in complex environments.

1. RELATED WORK

In the realm of people counting using YOLO-based object detection, previous research has explored various strategies to address challenges associated with crowd analysis, surveillance, and human tracking. Notably, studies such as [Author et al., Year] have leveraged YOLOv3 architecture for efficient people detection in crowded scenes, achieving commendable accuracy. Others, such as [Author et al., Year], have extended YOLO's capabilities by incorporating temporal information for tracking individuals across frames, contributing to improved counting accuracy in dynamic environments. Additionally, [Author et al., Year] proposed novel post-processing techniques to refine people counting results obtained through YOLO, mitigating issues related to false positives and negatives. These endeavors collectively underscore the versatility of YOLO in addressing the complexities of people counting tasks and highlight ongoing efforts to enhance its performance through innovative adaptations and methodologies.

In the landscape of related work concerning people counting using YOLO, numerous studies have delved into advancing the capabilities of object detection for crowd analysis and surveillance. Pioneering research by [Author et al., Year] demonstrated the effectiveness of YOLOv4 in densely populated scenes, showcasing superior accuracy in identifying individuals amidst challenging occlusions. Further contributions from [Author et al., Year] explored the integration of multi-sensor data with YOLO for enhanced people counting accuracy, addressing environmental complexities. Noteworthy investigations, such as [Author et al., Year], have extended YOLO's applications to real-time video streams, enabling dynamic tracking and counting of individuals in motion. Additionally, [Author et al., Year] proposed novel methodologies for adapting YOLO to varying environmental conditions, emphasizing the model's adaptability to diverse scenarios. Collectively, these endeavors underline the continual evolution of YOLO-based approaches and their significance in advancing the state-of-the-art in people counting applications.

Conclusions

An overview of the study's main conclusions and contributions. A summary of the YOLO-based approach's efficacy in crowd counting. Concluding thoughts regarding the importance of the suggested approach in the domain of surveillance and crowd analysis. Overall, the successful application of YOLO technology in crowd counting marks a significant advancement in the field of computer vision, opening up new avenues for the development of intelligent and efficient crowd analysis systems for various real-world applications.

Acknowledgment

We would like to express our gratitude to all those who contributed to the successful completion of this crowd counting project. Our sincere appreciation goes to the research team for their dedication and valuable insights throughout the course of this study. We extend our thanks to the institutions and organizations that provided the necessary resources and support, enabling the smooth execution of this research.

We express our sincere gratitude to all those who have contributed to the success of this research on people counting using YOLO. Our appreciation extends to the researchers and developers behind the YOLO (You Only Look Once) object detection framework, whose pioneering work has laid the foundation for efficient and accurate real-time detection. We acknowledge the efforts of the dataset creators, whose meticulously annotated datasets have been invaluable in training and evaluating our model. We extend our thanks to the academic community for fostering an environment of collaborative learning, and we appreciate the feedback and insights received from colleagues and mentors throughout the course of this research. Additionally, we would like to recognize the support provided by [Organization/Institution Name] and the funding agencies that have made this research possible. This collaborative effort underscores the collective dedication to advancing the field of computer vision and its applications in people counting.

References

The heading of the References section must not be numbered. All reference items must be in 8 pt font. Please use Regular and Italic styles to distinguish different fields as shown in the References section. Number the reference items consecutively in square brackets (e.g. [1]).

When referring to a reference item, please simply use the reference number, as in [2]. Do not use “Ref. [3]” or “Reference [3]” except at the beginning of a sentence, e.g. “Reference [3] shows …”. Multiple references are each numbered with separate brackets (e.g. [2], [3], [4]– [6]).

1. Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2015). Cross-scene crowd counting via deep convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 833-841).
2. Boominathan, L., Srinivasan, G., & Jawahar, C. V. (2016). Crowdnet: A deep convolutional network for dense crowd counting. In Proceedings of the British Machine Vision Conference (BMVC) (pp. 1-12).
3. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 779-788).
4. Sindagi, V. A., & Patel, V. M. (2017). Generating high-quality crowd density maps using contextual pyramid CNNs. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 1879-1888).
5. Lempitsky, V., & Zisserman, A. (2010). Learning to count objects in images. In Advances in Neural Information Processing Systems (NeurIPS) (pp. 1324-1332).
6. Idrees, H., Saleemi, I., Seibert, C., & Shah, M. (2013). Multi-source multi-scale counting in extremely dense crowd images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 2547-2554).