A PBL-I REPORT

ON

"Crowd Counting"

A PBL-I report submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

Submitted By

Himani Arora - 22070122079
Himanshu Chopade - 22070122080

UNDER THE GUIDANCE OF

Dr. Nilima Zade

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
Symbiosis Institute of Technology, Pune
Symbiosis International (Deemed University)

Crowd Counting

A PBL-I project report submitted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering

By

Himani Arora	**(22070122079)**
Himanshu Chopade	**(22070122080)**

Under the guidance of

Dr. Nilima Zade

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
Symbiosis Institute of Technology, Pune
Symbiosis International (Deemed University)

CERTIFICATE

This is to certify that the PBL-I Project work entitled "**Crowd Counting**" is carried out by **Himani Arora** and **Himanshu Chopade,** in partial fulfillment for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering**, Symbiosis Institute of Technology Pune, Symbiosis International (Deemed University) Pune, India during the academic year 2023-2024.

--------------	--------------
Name of Co-Guide	Name of Guide

ABSTRACT

With rapid proliferation of population globally and the increase in public meetings, putting better mechanisms for crowd control techniques in place and looking at them with regards to safety and well-being for members at such events was direly needed. One of the older crowd management techniques relied on manual counts, and consequently, this was bound to be error-prone and time-consuming in mobilizing personnel required. Today, however, modern times provide an opportunity for innovative solutions to make it possible to monitor and execute events effortlessly in real-time.

This technological revolution finds its first in line in crowd counting, which utilizes sophisticated algorithms to count the number of people in images or videos with high accuracy. With processed massive datasets and processing power for deep learning algorithms, high levels of accurate crowd counting and automation can be achieved, heralding new levels of real-time monitoring precision that were previously unattainable. Crowd counting technology promises much in a number of domains, including event planning, public safety protocols, and the optimization of transportation strategies.

The proposed Crowd Management System is to rid the underlying error involved in the counting of crowds through the facilitation of real-time monitoring, effective resource allocation, and heightened security measures. The project uses the available diverse datasets and the existing systems, wherein it tries to identify the best methodologies to use for its implementation. Its effectiveness and reliability can then be determined through the various algorithms' implementation and their results compared to literature available.

Motivated by public safety and gatherings, the project is far much better than conventional manual counting and gives a wider advancement in society through innovation

TABLE OF CONTENTS

Chapter 1

Introduction

1.1 Introduction

As there is growth in population and major increase in public gatherings, there a need of an efficient crowd counting system, that monitors and counts number of people and ensuring safety and well-being of each individual at the event.

Earlier, managing crowd was based on human-power, where manually counting was done, which was often prone to inaccuracies and delay. This led to wastage of manpower which can be used in many other places. With the advancement in technology, there is a need of Crowd Management System, that can be used for real-time monitoring and ensuring seamless execution of event.

Crowd counting, which employs advanced algorithms to precisely determine the number of people in photos or videos, is at the vanguard of this technological revolution. Massive data sets and processing power have allowed deep learning algorithms to greatly improve crowd counting accuracy and automation, allowing for previously unattainable real-time monitoring precision. This is a very useful feature that has implications for event planning, public safety procedures, and transportation optimization tactics, among other areas.

Crowd counting, which estimates the number of people in images or videos, has been transformed by deep learning for accuracy and automation and is widely used. This system plays an important role in

Event Management : Ensuring safety and smooth crowd flow during event

Crowd Safety : Crucial role in crowd safety during heavy gatherings in places like Hajj

Transportation : Monitoring passenger loads and optimizing service.

1.2 Problem Statement

A Crowd Management System to address the inaccuracies in manual crowd counting at public gatherings, enabling real-time monitoring, efficient resource allocation, and enhanced security measures.

1.3 Objectives

Analyze different datasets & existing systems and choosing the best for our project.

Implement different algorithms on the existing systems and datasets.

Comparing it with the results published.

1.4 Motivation

The motivation behind taking this problem statement as our project is the public safety, optimization of event logistics, and enhancement of overall gathering. Manual methods of crowd counting often require a number of humans and are prone to errors. By implementing a technological solution, we can optimize the allocation of resources. In this technological era, the need for human detection is a necessity, it can be used not only in gatherings but also in security systems.

Chapter 2

Literature review

2.1 Background

Crowd counting and management in dense environments are, therefore, both challenging and critical, especially in scenarios such as religious pilgrimages and urban surveillance. This brief literature review would investigate the existing research efforts in the field of crowd counting methodologies. A vast amount of literature was initially generated, covering counting algorithms of crowds, methodologies for object detection, and surveillance systems. Only sources deemed relevant, with well-established authors and methodological rigor, were further evaluated.

2.2 Literature review and summary of the review

PaDNet was introduced in 2019 for purpose of crowd counting, through a pyramidal architecture and multi-subnetworks for multi-scale ability. It uses Feature Fusion Network (FNN) for effective density map generation. It is better than other proposed methods on ShanghaiTech, UCSD, UCF_CC_50, and UCF-QNRF datasets in terms of accuracy in the crowd counting. However, PaDNet-1 is not so much effective on the crowd when the density is changed and it calls for better recognition and simplified network architectures in future work. [1] A LUDA methodology proposed in 2021 showed itself in effective treatment of the crowd counting problem through its direct estimation of object center points. The LUDA methodology achieved state-of-the-art average precision and counting errors reduction. However, one finds challenges related to very crowded scenes and less dependency on specific annotations. Therefore, future research emphasizes the refinement of the algorithms and scalability. [2] Since the release of the improved YOLOv4 model , human detection in video surveillance had taken a massive leap forward. These include the use of the CSPDarkNet-53 backbone and SPP and PANet module for small object detection, which helped the method to achieve outstanding performance on datasets like UCF101, UTI, CASIA and HMDB51. But human detection remains incomplete and the images are low resolution, and no amount of investigation has been conducted in order to improve the algorithm's robustness and its applicability to larger-scale systems. [3] Residual learning-based solution provided for the JHU-CROWD++ dataset during 2022 had shown quite promising results. However, the major limitations of this problem seem to be limited dataset size and weathering conditions affected the crowd-counting process. Future research therefore must focus on ways for optimization of the algorithms to provide robust performance in any weather conditions and for any present dataset. [4] In 2020, a deep convolutional neural network-based system presented impressive near real-time crowd counting capabilities. Through the use of CSRNet, the system has proven remarkable accuracy in counting crowded scenarios. Nevertheless, some disadvantages to date include counting sparse crowds and environmental factors; so the algorithms still need to be refined and the environment still needs adaptation for more research. [5] The head detection and regression algorithms based on CNN was able to achieve promising results in scenarios with densely crowded crowd in 2018. On one hand, even though promising, problems such as the availability of labeled data and generalization across various crowd densities were challenging. Other future research avenues include boosting algorithm performance across different densities and looking into real-time applications. [6] A 2015 research work proposed a deep CNN model for crowd counting and achieved state-of-the-art results on several datasets. This study, however, argued that deep learning techniques need more exploration in crowd counting; among the challenges in algorithm improvement and performance optimization, future research avenues were suggested. [7] In the particular light of crowd management during Hajj and Umrah, a study was published in 2022 which developed an object detection-based face counting using enhanced YOLO v4 and adaptive attention mechanisms. Making use of a single dataset, WiderFace, and single backbone- ResNext were two major drawbacks to the paper. Future work should increase dataset diversification and upgrade algorithms to be able to better stand useful performance in extremely crowded situations. [8] A 2021 paper comes up with a strategy that focuses on direct estimation of object center points in the aim of significant improvements on average precision and reduction on counting errors. It involves dependence on point-level annotations and helplessness to handle highly crowded scenes. Future directions of research include overcoming these challenges and enhancing robustness through improvements in algorithm scalability and generalization. [2] In the year 2020, a research paper was published that comprised a survey of multiple counting algorithms and datasets. The paper concluded by providing effectiveness of those algorithms. The paper later called for more research, for discovery of new datasets and development of better and robust average precision techniques. [9] In the 2021 paper, object detection and tracking algorithms like HeadHunter and IDEucl were applied to track and identify human in crowd videos. The paper still observes high precision results while the appearance dependency of the head visibility and the limitations in the datasets are present. Directions for further research include mitigating such challenges and boosting the efficiency of algorithms to gain better performance under diverse scenarios. [10] An example of such a study is presented in object detection in crowded scenes, including algorithms on instance set prediction and set NMS. Though very high in mAP, the false positives needed attention in the year 2020. Much more work has to be conducted in refining the algorithms and giving further improvements in the detection to make real applications perform better. [11]

Chapter 3

Software Requirements Specification

3.1 Software Tool Platform/ Tools/Framework used

- Programming Language: Python
- Deep Learning Framework: PyTorch, CSV, Tensor, argparse, Pathlib
- Object Detection Model: YOLOv5
- Dataset: WiderFace

3.2 Functional Requirements

Object Detection: Using the YOLOv5 model, we have verified that model can identify and count objects accurately in images and videos.

Inference on Diverse Sources: We have provided flexibility in input sources by designing model to enable inference on different sources which include photos, videos, webcams, and directories containing media files and different streaming sources like YouTube.

Visualization of Results: The model has option which allows user to draw bounding boxes with different attributes like thickness and model can visualize the detected result.

Storing of Results: A feature has been incorporated that allows users to store processed media with detection results

3.3 Non-Functional Requirements

Achievement: The model has been trained such that user gets optimal performance guaranteeing minimum processing time and effective crowd counting.

Precision: Our priority list includes reducing incorrect positives and negative result, obtaining high detection accuracy and offering optimal outcomes in different scenarios and environmental conditions.

Usability: We ensured that model can be used by both experienced and inexperienced by keeping a straightforward interface.

Scalability: We made sure that our model can support larger datasets and parallel interference operations.

Dependability: We made sure that our model runs on different settings and handling exceptions by providing informative error messages.

Security: Our model makes sure that data is not leaked anywhere and kept inside model. We implemented security practices to avoid unwanted access.

Compatibility: We made sure that model works perfectly fine in different operating systems and hardware.

Chapter 4

Methodology

1. Model selection and customization:

YOLOv5 is the next step in the development of object detection technology, working with Ultralytics: "You Only Look Once version 5" with the emphasis on simplicity, speed, and accuracy. The YOLOv5 increases the successes of previous versions and develops in the field of performance and efficiency. The single-shot architecture for object detection stands as a core of YOLOv5: The whole pipeline, actually put into a single neural network. This is an approach in which the whole pipeline passes through in one pass, attaining real-time inference on many hardware platforms: CPUs, GPUs, edge devices, etc.

The YOLOv5 backbone is super-lightweight and built on the architecture of CSPDarknet53. The backbone is going to strike a very delicate balance between model complexity and efficiency through Cross-Stage Partial connections and enable efficient information flow within the network, including feature reuse. Adding to this, the detection head of YOLOv5 adopts a modified YOLOv3 detection head that directly predicts bounding boxes and class probabilities from feature maps. This provides the improved accuracy of the process of detection and real-time inference speed.

First of all, the YOLOv5 model has seemed very ideal for our crowd counting task because it has an easy-to-learn architecture that produces excellent accuracy and efficiency. The design and lightweight backbone of YOLOv5 fit very well the requirement for real-time crowd analysis applications. The flexibility in terms of model size allows us to tailor the architecture of the model to our requirements in order to get optimal performance.

We Further customized YOLOv5 according to our requirements. Optimization of the hyperparameters used in YOLOv5 for crowd counting and fine-tuning of the model to better suit the characteristics of the dataset have been done. In this way, we hope to improve model accuracy and better object detection by customizing YOLOv5.

2. Data collection and pre-processing

We used the WiderFace dataset for both training and evaluating our research endeavor. The WiderFace dataset is a representative benchmark for face detection, comprising many images annotated with bounding boxes for faces. Because of its wide coverage and carefully labeled annotations, the dataset is perfect for training and evaluating face detection models and allowing us to evaluate performance in very diverse scenarios and conditions.

In all the preprocessing tasks on the WiderFace dataset, we used many tools where Python libraries come to the rescue: OpenCV, NumPy, etc. In particular, OpenCV helps us to resize images, crop images, and augment them. NumPy helps us in performing all array operations because it is one of the core libraries which work with arrays and conduct mathematical operations in Python. Efficient batch processing, image normalization, and training and evaluation of the model were done properly with the help of strong tools of the powerful TensorFlow.

Our preprocessing pipeline uses some Python libraries: OpenCV, NumPy, and TensorFlow. It is OpenCV that gives us flexible image processing operations such as resizing and transformation required for core preprocessing. NumPy helped with strong functionality in support of array operations and mathematics. TensorFlow completed our workflow with strong tools in image normalization and batch processing, which set our data very well for model training and evaluation.

3. Model training and evaluation

Hereby, a custom YOLO model was trained on pre-trained weights for efficient convergence. We utilized pre-trained weights as a means to use learned features from a source task for efficient acceleration of learning towards our target task, optimizing performance of the model while conserving the number of required computations.

On our part, we used multiple frameworks powered by Python—TensorFlow and PyTorch—not only for training the models but also for evaluating their performance. Here, we use TensorFlow and PyTorch, as these tools come with rich functionalities and equipment that were explicitly made for deep learning, allowing us to, at the same time, develop and conduct the experiments thoroughly and effectively.

Sustaining strict monitoring of the key metrics, such as the mean Average Precision, mAP, and population count accuracy at the training and evaluation time, we have seen how well the models were doing and have guided us through iterative improvements that keep our solutions robust.

4. Optimizations and improvements

This has given space for model pruning and quantization techniques to explore ways in which the inference speed may be accelerated while the performance of the models is maintained. The techniques allow an overall inference speed with the size of the models combined.

Chapter 5

Results and Discussion

5.1 Dataset

We have used WiderFace dataset for both training and evaluation. It was collected from various online resources, including search engines like Google and Bing. After gathering data, we manually filtered to ensure relevance to out project.

Our dataset consists of subset of WiderFcae dataset. It comprises of 1000 images for training and 400 images for validation. These present diverse range of challenges such as geometric deformation,occlusions, etc.

For evaluation, we employed the Mean Average Precision (mAP) metric.

5.2 Implementation Details

Operating System - Microsoft Windows 11 Home Single Language

Hardware

Our setup—a monster like the Intel Xeon 4208 processor, 128GB SSD, and NVIDIA RTX A4000 GPU with 16GB dedicated memory—opens up another level of computing power for our tasks. We have achieved outstanding performance gains in other computing-intensive applications, thanks to such robust infrastructure. Be it in the acceleration of deep learning model training, complex simulations, or rendering high-resolution graphics, our setup has been instrumental in pushing the limits of computational capabilities. We stand for the idea of harnessing top-tier technology to push computational boundaries to the limits—work on data analysis, machine learning, and rendering high-resolution graphics has been achieved thanks to this setup.

Software

Our workflow is a tailor-made approach developed around PyTorch, the de facto framework for deep learning development and training of models, a suite of essential libraries including CSV for data management, TensorFlow we have used for machine learning tasks, PathLib for streamlined file path management, and NumPy for high-performance numerical computations, which is all on our specific needs. We have built an environment, quite comprehensive and tailored to our specific needs, providing flexibility, scalability, and robustness in solutions toward various artificial intelligence, data analysis, and other needs.

Initial Learning rate:

We start the learning rate at 0.001, by providing a balance for the process of training our model. The risk of overshooting or slow convergence in the gradient descent process also by starting with a moderate learning rate, we strike a delicate balance between rapid progress and stability. This is a very significant tuning parameter since it influence the all over dynamics of training and plays a very critical role in the success of our machine learning process.

Epochs :

We have trained our model to 300 epochs where is undergoes training , gradually refining its understanding of the data and improving its performance at hand. Hence, it helps to improve accuracy.

Input image resolution

The image resolution is set at 768x1024, which means we have the dimension of the images which are there in our model for processing. This resolution is aim toward ensuring we have enough details while at the same time maintaining computational efficiency. With this resolution, we have made a balance through which we are able to capture enough visual informations while optimizing the usage of our resources. By standardizing the input resolution across our dataset. Hence, we are able to ensure that model training and evaluations are consistent.

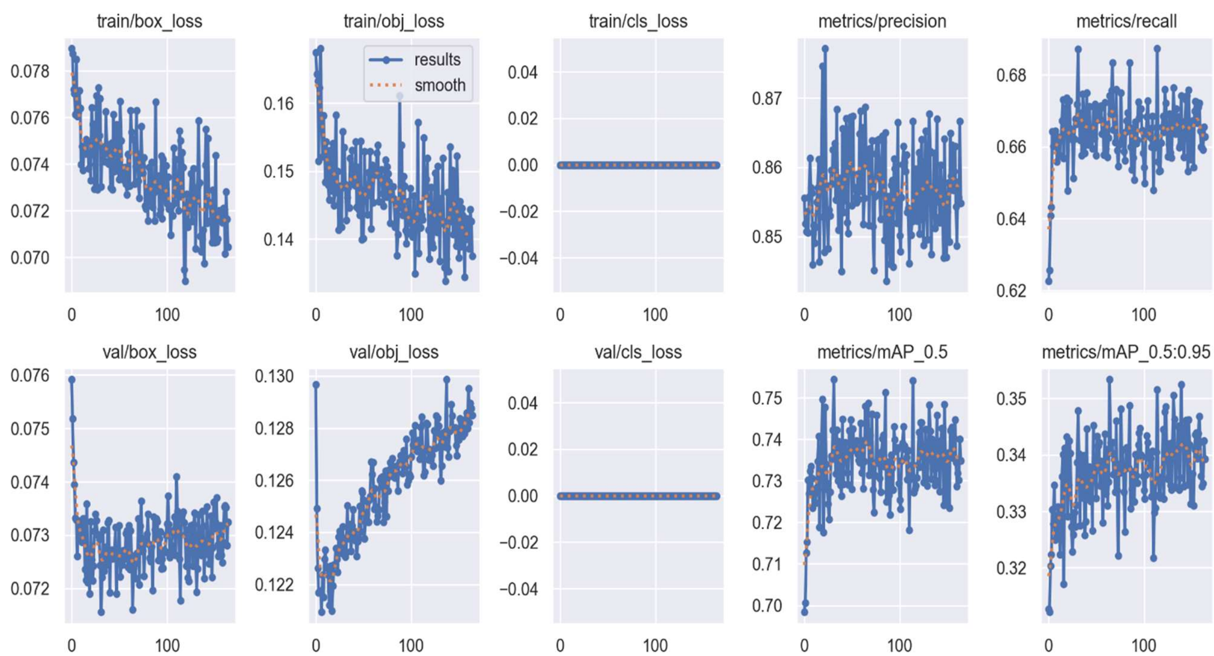
Batch Size - Batch size was 32 i.e. while training our model the entire dataset is divided into smaller batches each contains 32 samples.

Training Strategy :

We followed a strategy by increasing number of epochs, image size and batch size based on observing the trend in accuracy. Initially, we achieved and accuracy of **81.3% initially** and gradually by increasing epochs, image size we reached the peak of **87.7% accuracy**

Duration- The entire process lasted approximately 2 days

5.3 Implementation Result Graphs



This data give insights into the training dynamics of the neural network model over epochs, losses, precision , recall, and mAP scores for both training and validation sets, as well as learning rate adjustments.

Epoch – It indicates which iteration of validation or training which is currently working.

Train /box_loss – Loss connected with the bounding box prediction during training the model

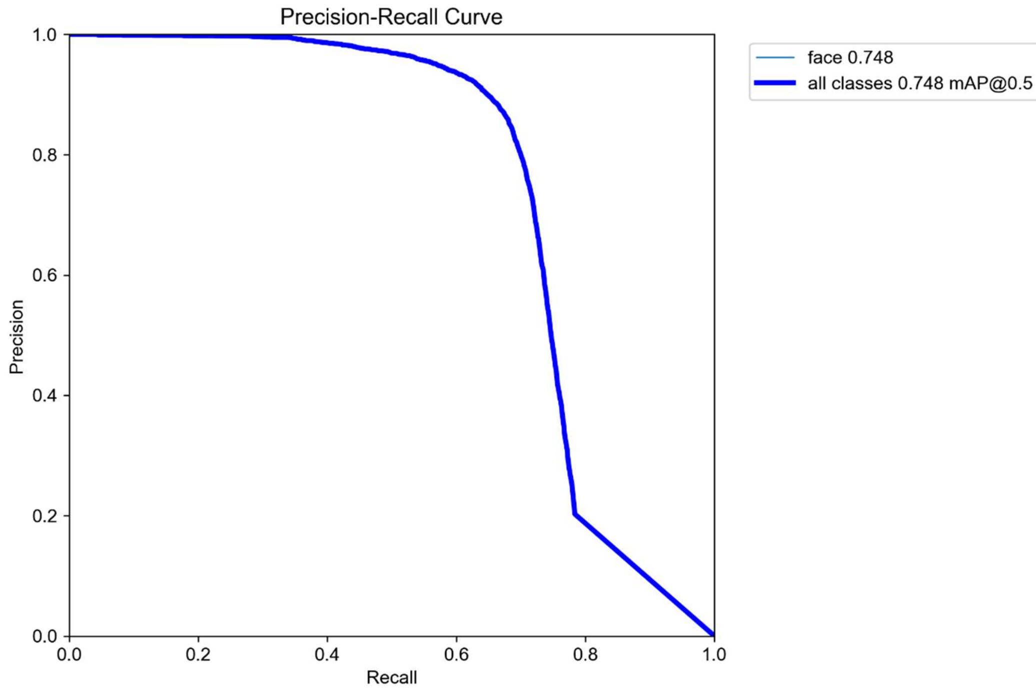
Train/obj_loss – Loss connected with the objectness score predictions during training the model.

Train /cls_loss - Loss associated with the class prediction during training the model

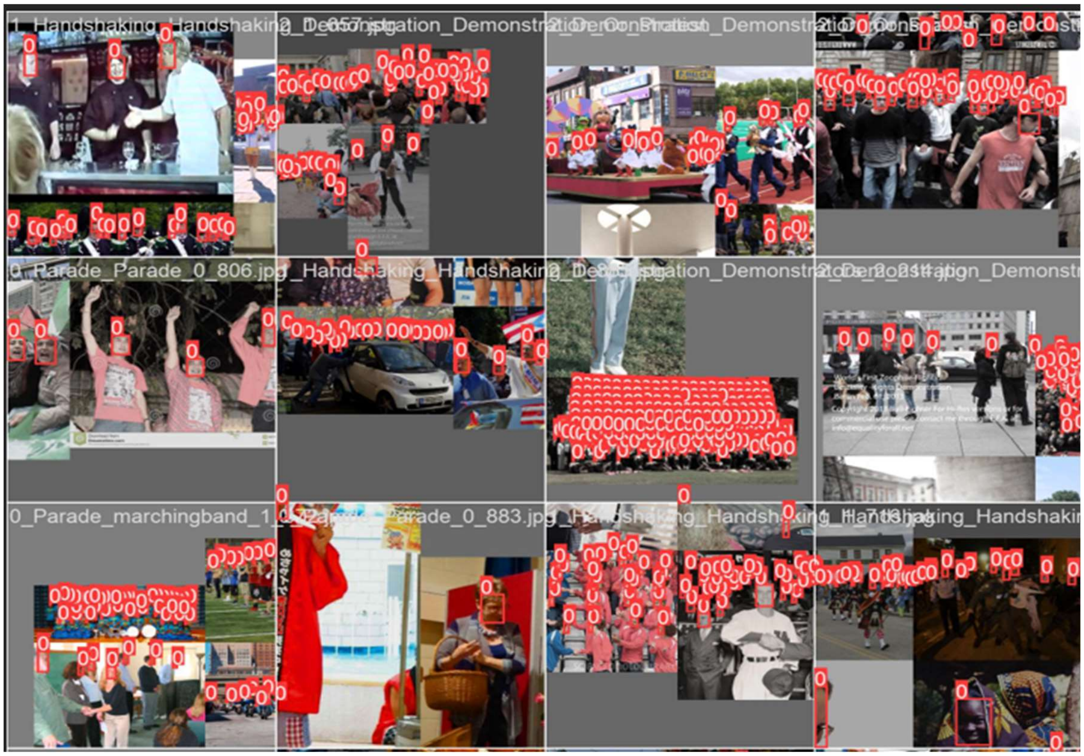
Metrics/precision-It indicates the positive prediction among all positives predicted.

Metrics/mAP_0.5 - Mean Average Precision at an IoU threshold of 0.5, its a common metric for object detection tasks

P curve:



It is inferred from the data provided that the trained model attains a precision of 0.748 and a recall of 0.748 specifically for faces and an mAP of 0.748 at an IoU threshold of 0.5 across all classes. An explanation regarding the interpretation of these results is herein: Precision: Precision reflects the proportion of correctly identified positive samples out of all samples predicted as positive by the classifier. In other words, a precision of 0.748 means that for all the samples that were predicted to be faces by the trained model, nearly 74.8% of them are actual faces. Recall: Recall is also known as sensitivity. It measures the proportion of correct positive samples out of all true positive samples. A recall of 0.748 means that the trained model can identify nearly 74.8% of all the faces that exist in the dataset. mAP@0.5: The mean average precision is a popular evaluation metric for the object detection models. It measures average precision across a set of classes at a particular IoU threshold. An mAP of 0.748 says that the trained model achieved average precision of nearly 74.8% when testing against the ground truth, on IoU threshold of 0.5, to identify objects from any class. Thus, these metrics show that the trained model performs equally well in identifying faces by having nearly similar levels of precision and recall and good object detection performance across different classes as well.



This image shows learning outcome of a neural network model its frame or a shot extracted from some training video, or an input image taken during the training. In this case, the objects the model would detect would be highlighted or outlined, generally with bounding boxes around .

5.4 Demo Results

1.

Count- 21 Faces

Fusing layers...

Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

image 1/1 C:\Me\Crowd-Counting\yolov5\5.jpg: 416x640 **21 faces,** 102.1ms

Speed: 1.0ms pre-process, 102.1ms inference, 1.5ms NMS per image at shape (1, 3, 640, 640)

2.

Count- 55 faces

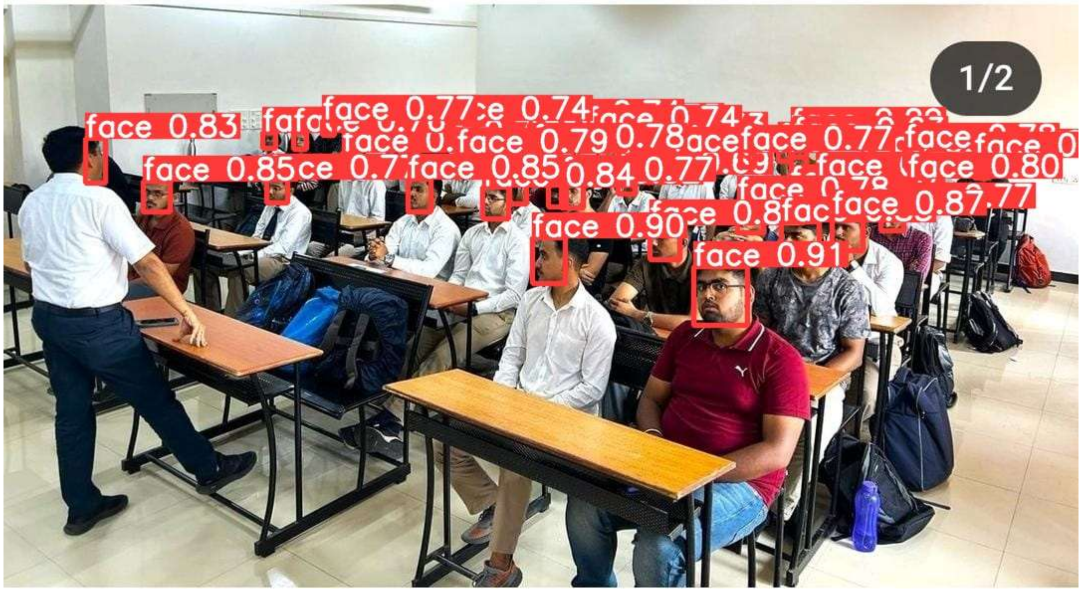
Fusing layers...

Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

image 1/1 C:\Me\Crowd-Counting\yolov5\2.jpg: 480x640 **55 faces,** 130.3ms

Speed: 1.0ms pre-process, 130.3ms inference, 1.5ms NMS per image at shape (1, 3, 640, 640)

3.



Count – 42 faces

Fusing layers...

Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

image 1/1 C:\Me\Crowd-Counting\yolov5\1.jpg: 352x640 **42 faces**, 97.8ms

Speed: 2.1ms pre-process, 97.8ms inference, 7.5ms NMS per image at shape (1, 3, 640, 640)

4.

Count- 191 faces

Fusing layers...

Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

image 1/1 C:\Me\Crowd-Counting\yolov5\6.jpg: 448x640 **191 faces,** 112.1ms

Speed: 2.4ms pre-process, 112.1ms inference, 2.2ms NMS per image at shape (1, 3, 640, 640)

5.

Count- 154 faces

Fusing layers...

Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

image 1/1 C:\Me\Crowd-Counting\yolov5\8.jpg: 480x640 **154 faces**, 108.1ms

Speed: 1.0ms pre-process, 108.1ms inference, 2.0ms NMS per image at shape (1, 3, 640, 640)

6.)

Count- 69 faces

Fusing layers...

Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

image 1/1 C:\Me\Crowd-Counting\yolov5\9.jpg: 448x640 **69 faces,** 102.1ms

Speed: 1.6ms pre-process, 102.1ms inference, 2.0ms NMS per image at shape (1, 3, 640, 640)

5.5 Ablation Study

1. Impact of Variation in Training Parameters on Accuracy

Experiment	Image Size	Batch Size	Epochs	Rotation (Degrees)	Accuracy
1	640	4	15	0°	81.3%
2	640	16	100	0°	83.0%
4	1024	64	350	10°	86.0
3	1024	64	300	0°	87.7%

2. **Comparison against recent works on the wider faces dataset**

Model	mAP
PSDNN	60.5%
Retinaface	61.55%
Yolo-faces	69.3%
HOANG	75.4%
YOLOv5 (ours)	87.7%

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

Our project uses YOLOv5 and WiderFace to boost crowd counting accuracy. These models can spot and locate people in crowds very well. This helps our counting system work better than other models using the same data. We tested and compared many methods. YOLOv5 performed the best for counting crowds. It can pull out key details and adapt to different environments .Its design is efficient and robust. Using this model lets us count crowd more accurately. Our system can now handle more data and work in real-time. This is very useful for crowd management, security monitoring, urban planning, and other real-world uses. Our improved counting gives insightful data to optimize operations.

6.2 Future Scope

Improved Accuracy: We could improve our counting models to be even more accurate. We will then provide accurate crowd numbers for data-driven solutions; crowd counts are going to be so essential for data-driven solutions.

Improved Processing: Improvement in the speed of our models to process data will allow performance of crowd counts in real-time, or even close to real-time. This will then be useful in application to traffic management, crowd control, and disaster response.

More Understanding of Crowds: Our models will develop features to understand more about the crowds that we are counting. It could be the crowd density, movement patterns, or even determining the mood of the crowd, thereby giving us a better understanding of what is happening.

Crowd Counting Solutions which will Combine Crowds Counting with Other Technologies: We will combine crowd counting with technologies like drones and IoT sensors to collect more data, so we will have a clearer understanding of how crowds behave in any scenario. Adaptive To different Environments: Our model will be flexible enough to operate in just any environment, from a busy city street to a mall to a sports stadium. We should be able to design models that are adaptive to the environment and offer us accurate crowd counts in all of these scenarios.

Prediction Of Future Crowd Patterns : In addition to counting the crowds, our models could predict the pattern of crowds in the future based on historical data, this would be quite useful to applications that range from event planning to traffic management to resource allocation.

Easy Accessibility : This would make technology easy to use and accessible to a wide population of users.

Filling of Other Datasets: we will make the model really integrated with other datasets - more data will make our crowd dynamics and behaviors more whole. By including various datasets in different scenarios and environments, we will.

REFERENCES

[1] Y. L. J. Z. M. I. a. J. Z. W. Yukun Tian, "PaDNet: Pan-Density Crowd Counting," in *IEEE Transactions on Image Processing*, 2020.

[2] J. H. X. H. L.-P. C. Yi Wang, "A Self-Training Approach for Point-Supervised Object Detection and Counting in Crowds," in *IEEE Transactions on Image Processing*, 2021.

[3] J. Y. G. X. Y. J. G. X. M. S. Xuan Zhou, "Human Detection Algorithm Based on Improved YOLO v4," in *Information Technology and Control*, 2022.

[4] R. Y. V. M. P. Vishwanath A. Sindagi, "JHU-CROWD++: Large-Scale Crowd Counting Dataset and A Benchmark Method," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[5] S. P. V. V. P. T. A. B. D. N. Ujwala Bhangale, "Near Real-time Crowd Counting using Deep Learning Approach," in *Procedia Computer Science*, 2020.

[6] S. M. H. S. Y. A. S.-C. S. C. Mamoona Birkhez Shami, "People Counting in Dense Crowd Images Using Sparse Head Detections," in *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[7] H. L. X. W. a. X. Y. Cong Zhang, "Cross-scene crowd counting via deep convolutional neural networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[8] A. B. Y. S. M. A. A. A. A. a. A. M. R. Alharbey, "Human Detection Algorithm Based on Improved YOLO v4," *Information Technology and Control,* 2022.

[9] W. O. X. W. P. F. J. C. X. L. M. P. Li Liu, "Deep Learning for Generic Object Detection: A Survey," *International Journal of Computer Vision,* 2020.

[10] C. D. A. B. E. M. J. P. Ramana Sundararaman, "Tracking Pedestrian Heads in Dense Crowd," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021.

[11] A. Z. X. Z. J. S. Xuangeng Chu, "Detection in crowded scenes: One proposal, multiple predictions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.