**SOLENT UNIVERSITY, SOUTHAMPTON**
**Faculty OF BUSINESS, LAW, AND DIGITAL TECHNOLOGIES**

**MSC Applied Artificial Intelligence and Data Science**
**Academic year 2021-2022**

**Keyur Patel**

# "SMART CROWD COUNTING USING DEEP LEARNING"

**Supervisor: DR. Hamidreza Soltani**
**September-2022**

**This Report is submitted in partial fulfilment of the requirements of Solent university for the degree of MSc. Applied Artificial Intelligence and Data Science.**

## Acknowledgment

I would like to express my gratitude and appreciation for Dr Hamidreza sultani. whose guidance, support and encouragement has been invaluable throughout this study. Dr. Hamidreza Sultani has been an excellent instructor, mentor, and thesis supervisor, providing guidance and inspiration with the proper balance of wisdom and levity. I'm appreciative of and pleased of my time spent working with Dr. Hamidreza.

## Abstract

The field of artificial intelligence (AI) known as computer vision has demonstrated cutting-edge benchmark results in a number of necessary fields, including object identification, picture classification, image segmentation, etc. Many industrial sectors have benefited as a result of the development of this field of AI. The merger of computer vision with deep learning also referred to as DL, which is also an area of AI that deals with algorithms inspired by the human brain, has brought machines to a point where machines can easily recognize & detect objects in an image. Convolutional neural networks (CNNs) are the algorithms in deep learning that are responsible for machine vision, which has allowed computers to provide decisions on data based on images. One of the most valuable aspects of computer vision is object detection, which counts the number of items in an image by creating a bounding box around each one and then counting the bounding boxes. Crowd counting is a crucial application of object detection where the goal is to count the number of objects, especially people, to make it worthwhile for a variety of things like pedestrian counting, depending on how many people entered a market at a specific time, counting the number of people at a specific location at a specific time, and so forth. Applications in this sector are essential in regard to security, protecting populated areas from criminal activity, such as walkways and retail malls. The purpose of this research is to perform analysis, critical evaluation & development an intelligent system using object detection (OD) techniques to apply the concept of crowd counting, particularly in the context of pedestrians. For applying crowd counting in the context of pedestrians, pre-trained CNN models will be used by applying the concept of transfer learning (TL). The dataset used in this study will be obtained from Kaggle, an online data science and machine learning platform, and will be used to apply this principle. The model's effectiveness will be assessed by measuring its capacity to draw bounding boxes around images and using loss error. The targets of this research are to critically go through the applications of computer vision, i.e., object detection & also to outline new challenges for upcoming researchers aiming to work in this field.



**Keywords:** Convolutional Neural Networks (CNNs) Computer Vision (CV), Artificial Intelligence (AI), Crowd Counting, Deep Learning (DL), Object Detection, Transfer Learning (TL)


https://github.com/keyurp132/Crowd_counting

# Table of Contents

# List of Figures

# Chapter-1:Introduction:

In The field of computer vision (CV) has opened up a number of new vistas as a result of recent developments in artificial intelligence (AI). Computer vision has advanced significantly with the integration of AI and deep learning (DL) techniques, enabling machines to distinguish between a cat and a dog from a given image by identifying distinguishing features in it. After benchmark findings on object identification programs, researchers' attention has shifted to counting the elements in an image or video.[46]

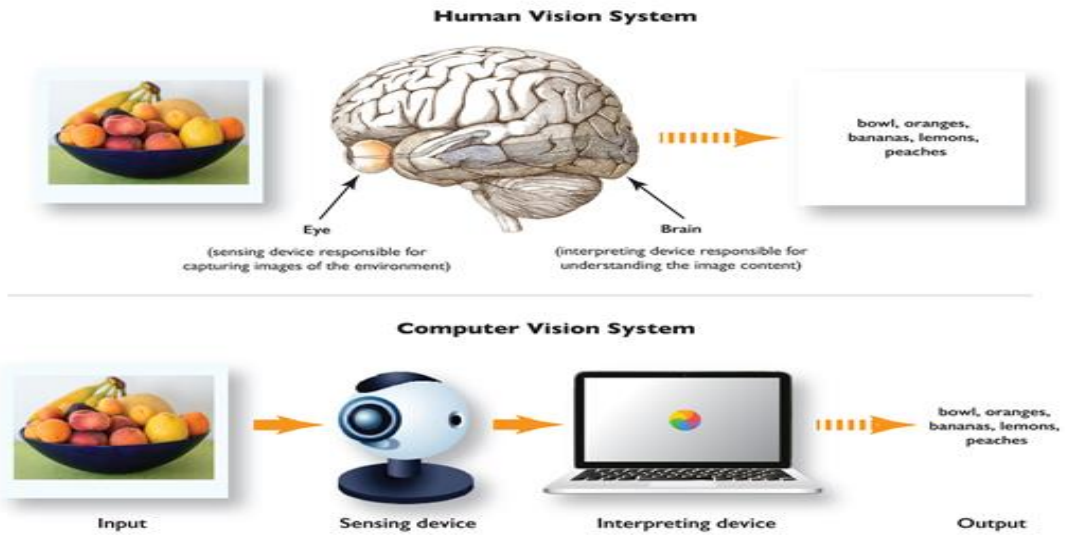



*Figure 1 Developments in Computer Vision[47]*

Researchers in AI and deep learning have found counting objects in an image to be challenging because it combines two tasks: first, identifying the objects in an image by creating bounding boxes around them using object detection, and second, counting those bounding boxes for the machine to make a final judgement and output the number of people in a crowd. Only pedestrian counting, which uses object detection to count people passing by a specific sidewalk, road, or footpath, is permitted to be done using crow counting.[46]

Crowd counting, for instance, is the practice of counting the number of people in a specific photograph or video. As implied by the procedure's name, it entails two steps: first, it intelligently draws a bounding box around any individuals it finds in an image or video frame, and then it adds a counter object or function to count those bounding boxes to produce the final output, which is the total number of people. In the context of our research, as we are interested in pedestrian counting, the application of crowd counting may be specially adapted to pedestrians. The objective is to count the number of pedestrians in a photo of a street, sidewalk, or other public area.

Prior to the introduction of deep learning in computer vision applications, it was thought to be extremely difficult to enable machines to do tasks like object detection in videos and picture categorization and identification. With the development of object detection algorithms using convolutional neural networks (CNNs), many new models have been produced, for example EfficientDet [1] that are able to detect objects in that image with higher accuracy. Preferably, pre-trained models are used by researchers as mentioned above, since they have already been trained on a much larger dataset, which show good results on testing datasets. YOLO v5 is yet another CNN model for object detection first developed by[2], but for this research we're aiming to use EfficientDet for performing object detection.[46]

variety of disciplines, including traffic system control, water resource management, various security applications, crisis management, etc. Traditional procedures like manual crowd counting and employing registers to capture people's information haven't proven to be effective or beneficial since crowd counting demands collecting dynamic aspects like people's movement. Using standard measurements might have a detrimental impact because of its generalizability and flexibility limits. These issues have led to the development of sophisticated crowd counting techniques that may be derived from CCTV footages as well as the necessity for them. One of the most significant and notable features of an intelligent crowd counting system is its ability to discern between dynamic human gestures, which is impossible with a traditional manual way of counting[4].

A common computer vision challenge is crowd picture analysis, which attempts to count the number of people in a scene captured on camera or in an image. Crowd counting is a branch of crowd analysis that focuses on calculating the population of a crowd based on how many

individuals are present there. Systems for counting crowds that are accurate and efficient are crucial in many aspects of urban life, including as safety and space management.[3]

Following is a picture that describes the concept of crowd image analysis in deep learning.

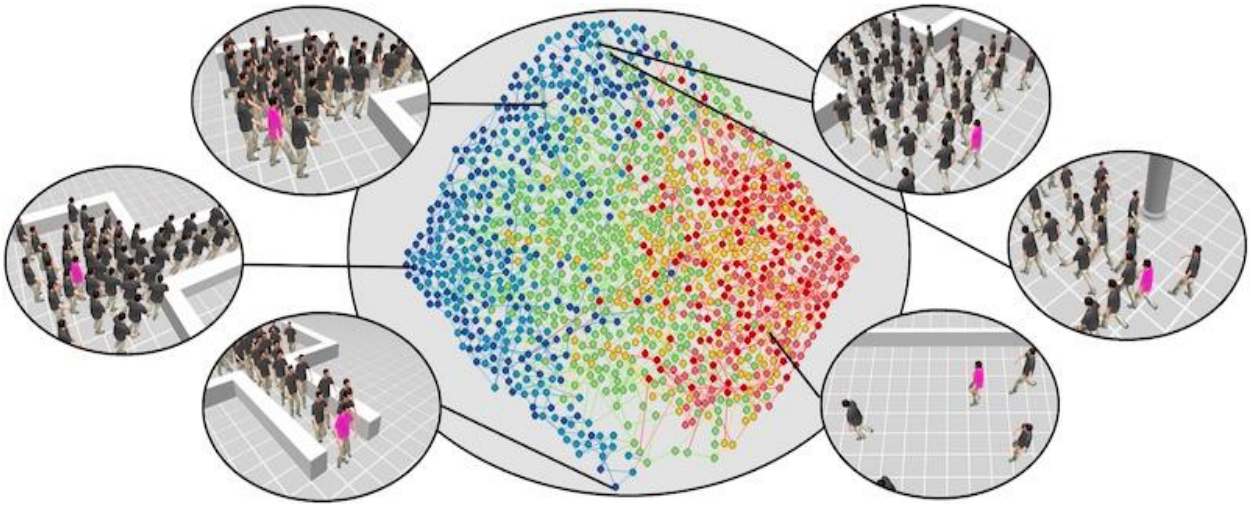


*Figure 2The Concept of Crowd Counting[48]*

Artificial neural networks (ANNs), also known as deep learning (DL), are a subfield of machine learning, which is a subfield of artificial intelligence (AI). Deep Learning (DL) is motivated by ANNs (ANNs). Input, hidden, and output layers can be used to define the structure of ANNs. The use of ANNs is widespread, particularly in the fields of vision, text, and optimum action. In order to compute over images, deep learning is combined with computer vision, another area of AI that deals with images and aims to give robots a comprehension of the visual world. Machines produce judgments over pictures, such as categorization, detection, etc., using deep learning and computers jointly. The image below illustrates the development of computer vision (CV) in terms of object

detection.

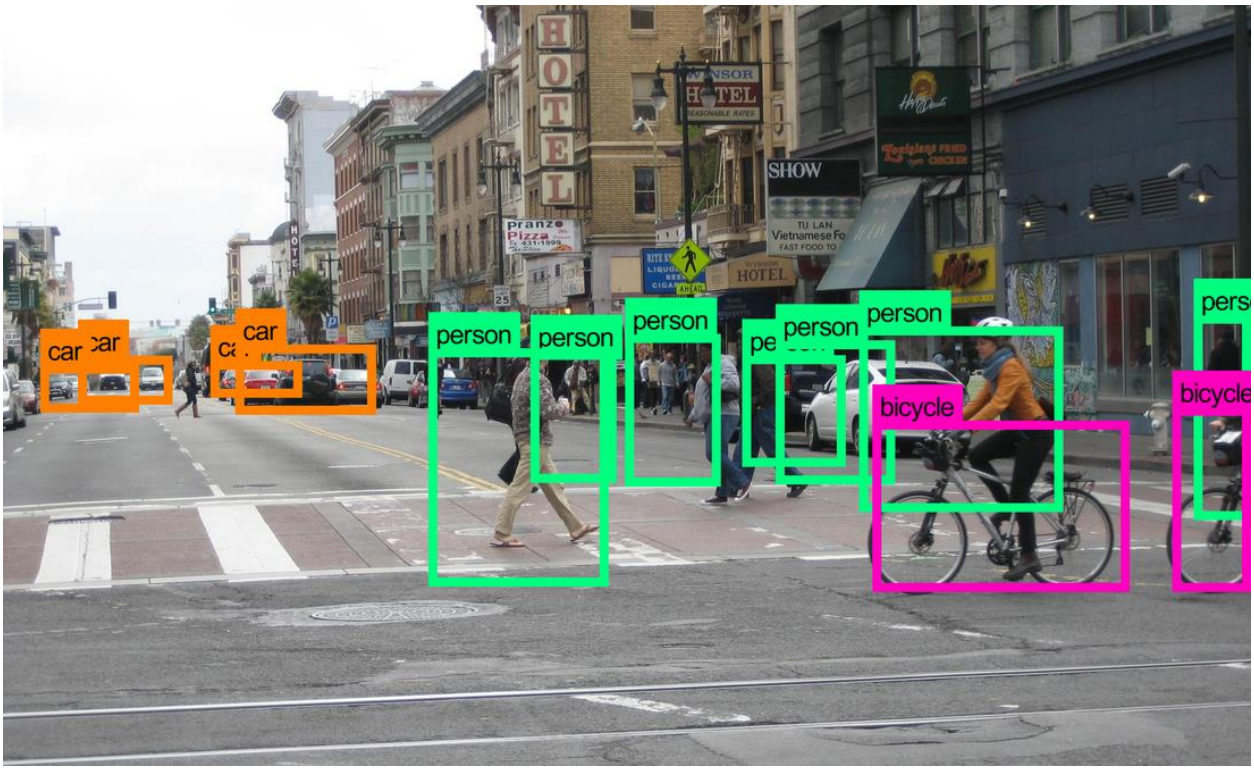



*Figure 3 Object Detection in Computer Vision*

In order to draw useful findings from brief investigations, the purpose of this study is to explore, investigate, and critically evaluate crowd counting methods in the context of pedestrian counting. In addition to outlining the current challenges in the area, this research will also explain upcoming challenges and offer guidance for upcoming scholars. Convolutional neural networks (CNNs), object identification, deep learning disciplines, as well as probable future challenges in image recognition and object detection, will be the main subjects explored in this research.

## 1.1:Aim(s):

The aims & objectives for undergoing this research work are as under:

- To assess and create an intelligent AI-based system that can count people in a crowd using deep learning

## 1.3: Objectives:

- To employ pre-trained deep learning models with the use of transfer learning (TL) for object recognition
- To comprehend the significance of AI applications in the fields of computer vision and deep learning
- To see how artificial intelligence may be used to solve difficult problems like object detection

## 1.4: Research Questions:

- How does deep learning handle computer vision use like crowd counting?
- Why is employing object detection models crucial for a crowd's population count?
- How may utilizing a trained CNN be more effective & significant than a manual technique in perfuming object detection?

## 1.5: Ethical Considerations

Big Data holds an importance place while doing research in data-related projects and research. For undergoing a smooth, steady and un-interrupted research, the UK Data Service Department has provided crucial guidelines, which also form the basis of ethical considerations for this research also.[46]

### 1.6: Project Philosophy

The philosophy of this project lies in the background of deep learning (DL) & computer vision (CV), which are themselves the areas of artificial intelligence (AI), and which are also the key elements on which the philosophy of this project is based. Crowd counting is a challenging problem in computer vision which requires a pre-defined and well-maintained dataset to be trained with a deep learning model, and the projected algorithms of deep learning like convolutional neural networks (CNNs) are projected to be applied on the dataset to make an intelligent model for crowd counting, which in return makes decisions on unseen or test dataset after being trained on training dataset. The logic of this project is structured so that, after a thorough analysis of previous work in the field (crowd counting) and after gathering insightful information about how other researchers have made a contribution to this subject, a new model will be implemented that may be comparable to the previous work of other researchers in this area of study.[46]

# Chapter-2: Literature Review

Following are the research papers, journals, and resources that have been gone through while writing the literature review of this research.

[5] presented a spatial-temporal convolutional neural network for counting objects in videos, in which the researchers outlined that classical or traditional approaches used for crowd counting in videos are less effective, since they deal with each frame independently without detecting the spatial or temporal correlation among them, which in result makes a less effective system for crowd counting. To fill this research gap, this research introduces a new spatial & temporal CNN that unifies the 2D CNN for images & 3D CNN for videos to make a unified model. This approach is utilized for calculating spatial & temporal features of a video. The model described in this research is applied on two benchmark datasets for testing, and it outperforms previous approaches by yielding a least amount of mean squared & absolute error rates.

(Li et al., 2021) provided a systematic review on various approaches being carried out in the context of crowd counting & density estimation. In this research, various approaches to deal with the challenging task of crowd counting has been addressed. The algorithms used for crowd counting are divided into detection, regression, CNN-based & video-based in this research work. For making an extensive study in this field, each approach is compared with each other in terms of efficiency & effectiveness on various benchmark datasets. At the end, future challenges in crowd counting have been summarized along with their possible solutions and also some development trends for the future are also presented.

[6] provided a research contribution by introducing scale-aware attention convolutional neural networks. In this work, the density map of each input image is calculated in which each pixel value indicates the corresponding crowd density. For overcoming the challenge of scale variation in images, an attention framework has been produced in this research. Keeping in front the recent development in deep learning & especially in attention framework, an attention mechanism is added in the proposed model which is capable of automatically focusing on local & global image scales, thus making the network to be ‘self-aware’. The count of crowd is made by summing up all values in the obtained density map. The model proposed in this research is claimed to be outperforming all existing architectures by summing up the local and global scale attentions.

[7] developed a CNN model in their research, known as deep-scale purifier network responsible for crowd image counting, known as DSP network. This network is capable of multi-scale feature encoding which can reduce the loss function for contextual crowd counting. The neural network presented in this research comprises of two parts, i.e., front-end & back-end. The front-end part of this network is a convolutional neural network, while back-end is comprised of a unified deep neural network. These two networks collectively work to learn the scale information of images at various levels. The neural network developed by researchers in this work also learns image-based inference for smooth model training & obtaining a minimum loss. The experiments for checking the skill of model are checked on tree challenging datasets and the network presented in the research provides state-of-the-art benchmark results.

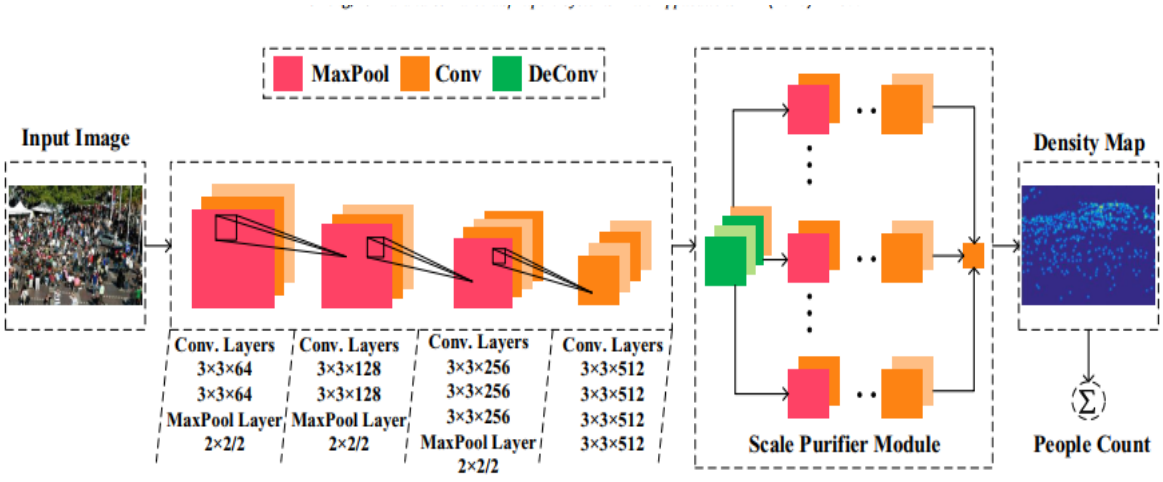



*Figure 4 CNN Architecture Used by[7]*

For counting people in dense crowds, [8] provided a research contribution by formulating a type of neural network, namely as spatial context learning network. The researchers highlight the counting persons in a crowd is a challenging task, therefore producing a neural network that could accurately count the number of people is required. For solving this issue, the network presented in the research is composed to three parts, i.e., feature encoding, context learning decoder & density regression. The features from input image are extracted from the feature encoder & then these features are passed to the context learning decoder to capture the spatial context information from various regions of input image. The decoder comprises of three attention modules which apply

attention mechanism to calculate the spatial context information. At the end, the context information is passed to density regression module for density estimation. Experiments of this research are conducted on three benchmark crowd counting datasets and obtained results indicate the effectiveness & efficiency of this proposed model as compared to other researches.

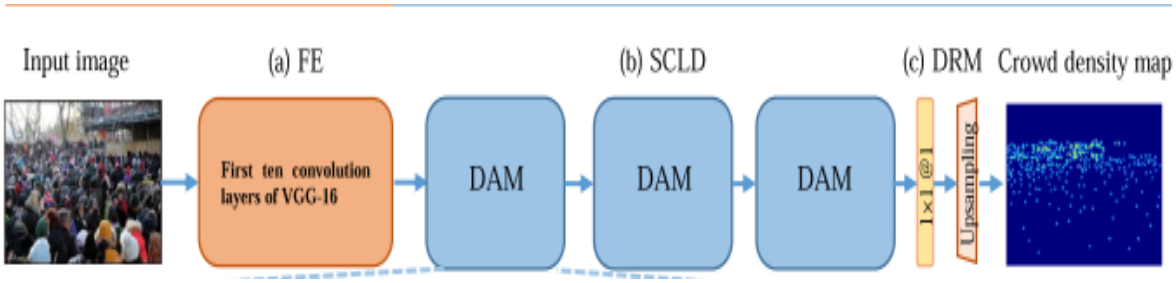



*Figure 5 Methodology Used by [8]*

[9] that convolutional neural networks (CNNs) employed for crowd counting have shown tremendous progress by making a density map of persons, in which each person is identified by a Gaussian Blob, and the final count is made by making an integration of the whole density map. Though standing with its effectiveness, still it raises a problem of accurate predictions on dense regions. These factors lead to considerable pattern shifting and pattern variation in the density map, which produces a long-tailed distribution of pixel-wise density values. The Learning to Scale (L2S) module, which can automatically scale crowded regions into acceptable proximity levels by reflecting image-plane distance between surrounding persons, is used in this study to address the problem with the density map. The suggested learning-to-scale approach may dissect the collected information in the ground-truth density map and dynamically separate the overlapping blobs by directly normalizing the proximity in various patches. As a result, it reduces pattern changes and the long-tailed distributions of density values, which enables the model learn the density map more effectively. The proposed model was evaluated on three benchmark packed datasets and demonstrated that it improves localization and regression benchmarks while also exhibiting cutting-edge speed.

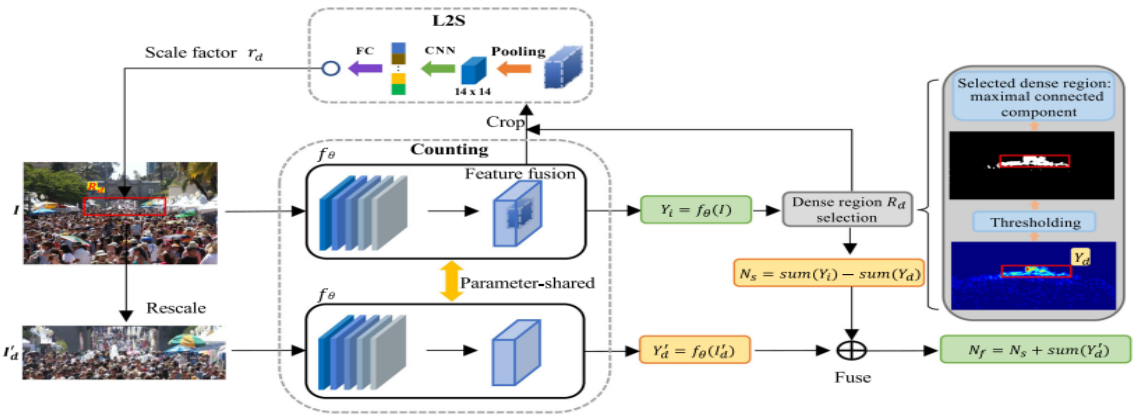



*Figure 6 L2S Methodology Used by [9]*

The notion of crowd counting was utilised by [8] by using a multi-scale dilated convolution in a convolutional neural network (CNN). According to this study, there have been many uses of crowd density in fields including crowd safety, one-side control, and others. A convolutional neural network (CNN)-based technique for estimating the density of a single static picture has been developed in this paper. It is known as Multi-scale Dilated Convolution of Convolutional Neural Network or (Multi-scale-CNN). This suggested technique used density maps regression to determine how a unique picture and density maps map to one another using convolutional neural networks (CNN). A convolutional neural network is used for general feature extraction in the proposed network structure, and a multi-scale dilated convolution network is used to address the scale change problem. These two components work together to adapt to changes in character sizes in crowd photos. The multi-scale dilated convolution network module uses dilated convolution without lowering the receiving domain to aggregate multi-scale context information in a systematic manner, integrating the underlying relevant details into the high-level semantic features to enhance the network's ability to perceive and count small targets. The skill of above multi-scale dilated convolutional network has been tested on two benchmark dataset such as ShanghaiTech and UC_CC_50, which are considered to be challenging to be experimented with crowd counting. The final results of these models showed an MAE of 83.7 and MSE of 124.5 on ShanghaiTech Dataset, while showed an MAE of 264.9 and MSE of 382.1 on UCF_CC_50 dataset.

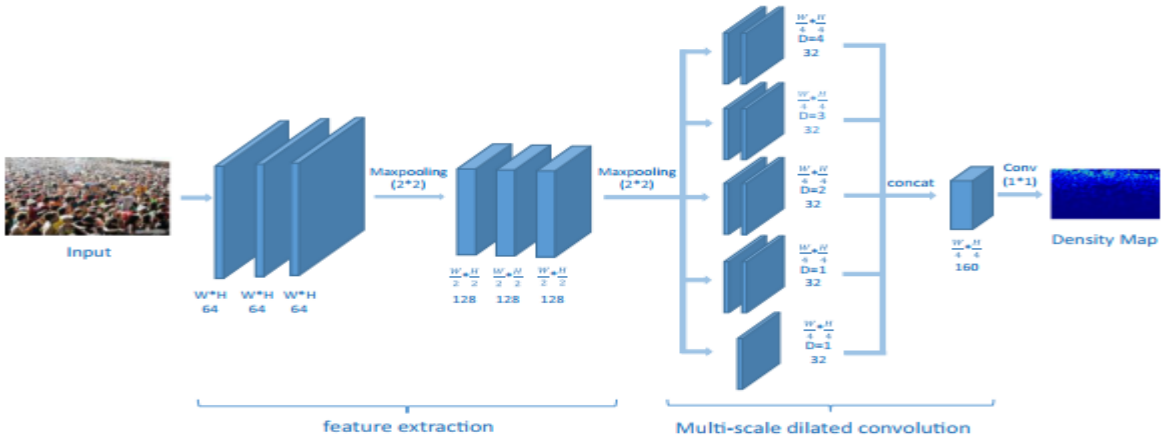



*Figure 7 Multi-Scale Dilated CNN Used by[8]*

[20]said that crowd counting has been showing great benchmark performances in deep learning (DL) and its applications have been increased dramatically in recent years. But still there remains a problem of cluttered backgrounds and changing scales of people in an image, which needs to be addressed. To address this issue, this study has proposed a Shallow-Feature-Based Dense Attention Network, also referred to as SDANet, which, while capturing the multiscale information by densely linked hierarchical visual features, reduces the influence of backdrops by incorporating a shallower feature-based attention model. Additionally, the researchers chose to base their attention model on shallow-feature maps in order to accurately detect background pixels since they observed that backdrops and human crowds typically exhibit markedly distinct reactions in shallow features. Furthermore, it is suggested to densely connect hierarchical image features of various layers and then encode them in order to preserve all of the most relevant features of images of people across various scales that may appear at different layers of a feature extraction network. This will allow for the estimation of crowd density. The experimental finding of this approach has been tested on benchmark datasets such as UCF_CC_50 which yielded mean-absolute-error (MAE) of 11.9 which is said to be outperforming previous approaches.

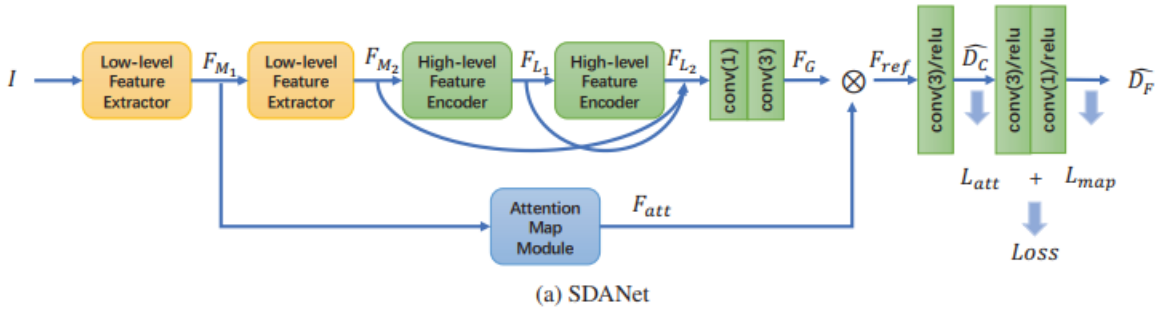



*Figure 8 Proposed SDANet Architecture Used by[20]*

[18] used attention scaling for crowd counting by highlighting a point that traditional convolutional neural networks (CNNs) perform crowd counting by taking it as a regression task, which in return outputs crowd density estimation. These CNNs word by learning representations between image content and density maps, and this technique has proved to be effective in crowd counting, but the issue with this technique is that this methodology over or under-estimates count of people that belong to various density patterns. To deal with this issue, this research has introduced attention scaling mechanism for crowd counting that could bypass this problem. The two networks used in this method are the Attention Scaling Network (ASNet) and the Density Attention Network (DANet). Attention masks pertaining to areas of various densities are sent to ASNet via the DANet network. In order to produce distinctive attention-based density maps, ASNet must first build density maps and scaling factors, which it then multiplies by attention masks and all of these density estimation maps are then summed to get the final crowd. Furthermore, the study's attention scaling factor may aid in minimizing estimating blunders. Testing of this approach's results on datasets including UCF CC 50, UCF QNRF, and WorldExpo'10 clearly demonstrates its superiority to earlier methods, with an MAE of 57.78 and MSE of 90.13. Furthermore, the study's attention scaling factor may aid in minimizing estimating blunders. Testing of this approach's results on datasets including UCF CC 50, UCF QNRF, and WorldExpo'10 clearly demonstrates its superiority to earlier methods, with an MAE of 57.78 and MSE of 90.13.

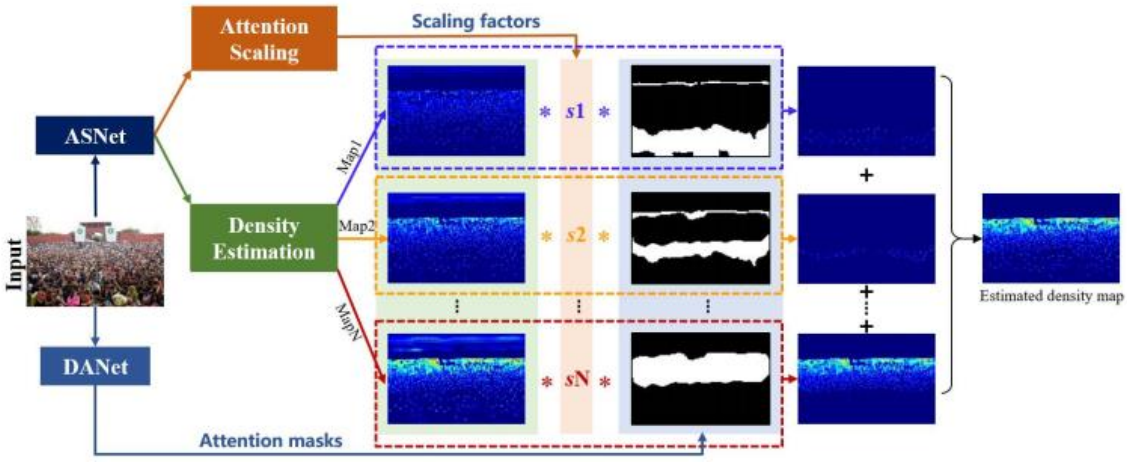



*Figure 9 Proposed CNN Model Architecture Used by [18]*

[12] provided a comprehensive survey of crowd-scene analysis using deep learning techniques. Recent events all over the world have proved that the importance of crowd counting has increased. Especially due to COVID-19 pandemic, the urge of crowd counting based systems have increased due to a lot of reasons especially in terms of crowd tracking, but it seems to be a very challenging task due to diversity of crowd scenes and to correctly identify each person in a crowd. Crowd counting and crowd behaviour identification were two of the key areas for analysis. The study also suggests an assessment measure, which provides an assessment of the discrepancy between the estimated crowd count and ground truth count. This survey could help to be useful in understanding the challenges in crowd counting and to comprehend this concept.

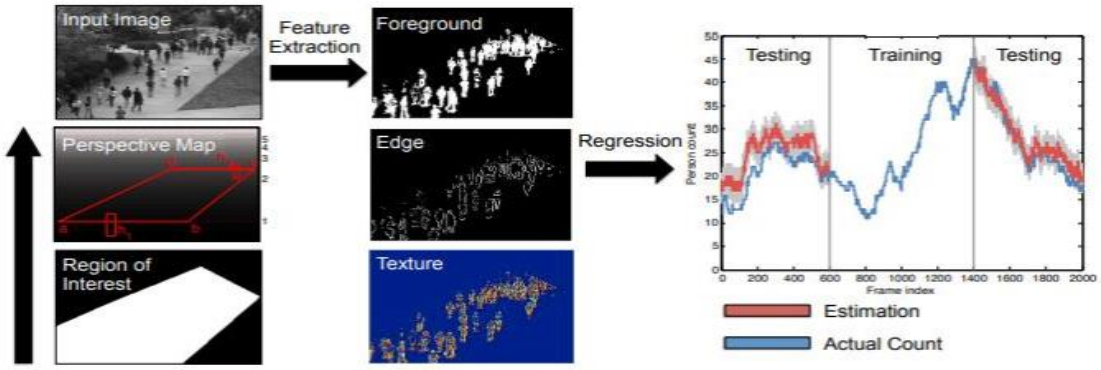



*Figure 10 A Regression-Based CNN model for crowd counting by[12]*

[49]Researcher suggested that due to the rise in counting crowd and vechicles in today's world, crowd counting has become a topic of interest to various researchers. Powered by deep learning (DL) techniques, the advancements in crowd counting have shown a great increase. This research aims to deeply understand the developments and advancements in crowd counting techniques by analyzing four major categories such as regression-based, convolutional network-based, video-based and detection-based crowd counting. For making a better go-through of this concept, applications and performances of crowd counting on various benchmark datasets are also discussed and the evaluation metrics used by various researchers for crowd counting is also elaborated. At the end, future challenges in crowd counting are described and useful insights are given to benefit other researchers that tend to work in this area.

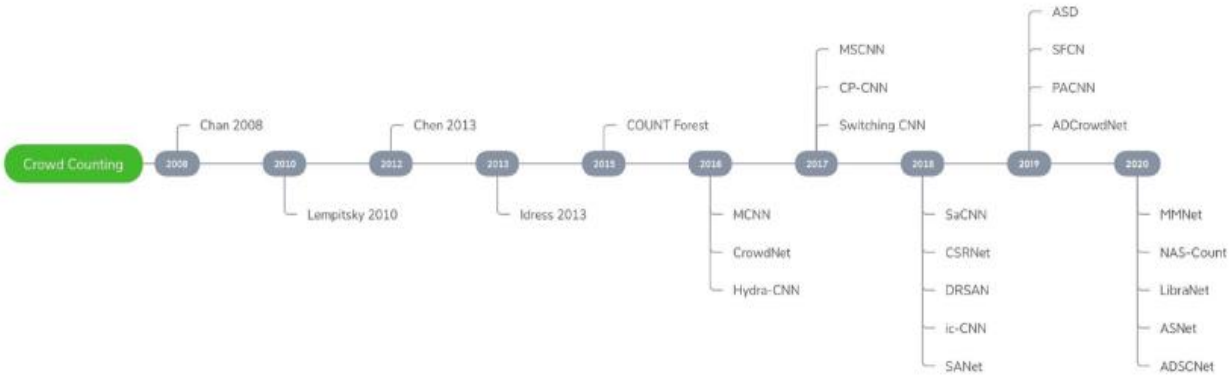



*Figure 11 Schematic Diagram of Survey Conducted by [49]*

[43] applied context-aware mechanism for applying crowd counting in images and videos. This study has shown how deep neural networks are used to assess crowd density in state-of-the-art approaches for counting individuals in crowded settings. The same filters are applied to the entire image or to sizable image patches for this purpose, and they simply estimate local scale to correct for perspective deformation in this situation. This is commonly handled by training an extra classifier to pick the best kernel size out of a constrained list of options using predetermined picture examples. This paper introduces a method that learns the significance of each feature at each place in an image by merging features gathered using diverse receptive field widths. The method is an end-to-end trainable deep architecture which yielded an MAE of 62.3 and RMSE loss of 100.0 on ShanghaiTech Dataset and is claimed to be outperforming existing approaches.

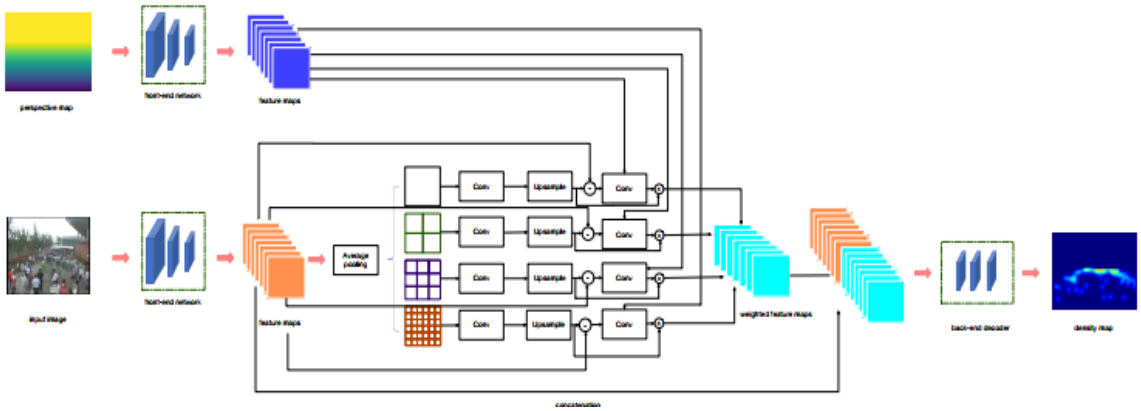


*Figure 12Context-Aware Convolutional Network by [43]*

[50] used deep-negative correlation learning by applying deep convolutional neural networks (CNNs) by stating that CNNs have been widely used in a wide range of computer vision applications, but still these procedures lead to overfitting due to single-image adaptation in crowd counting. A novel learning method that uses deep negative correlation learning to build generalizable features has been proposed to address this weakness. More particular, by controlling their inherent diversities, the network thoroughly learns a set of decorrelated regressors with good generalisation properties. The suggested approach, known as decorrelated Conv-Net (D-Conv-Net), is an end-to-end trainable model that is independent of the topologies of the underlying fully-convolutional networks. After holding extensive experiments with deep VGGNet, the proposed D-Conv-Net produces benchmark results with RMSE loss of 404.7 and MAE loss of 288.4.

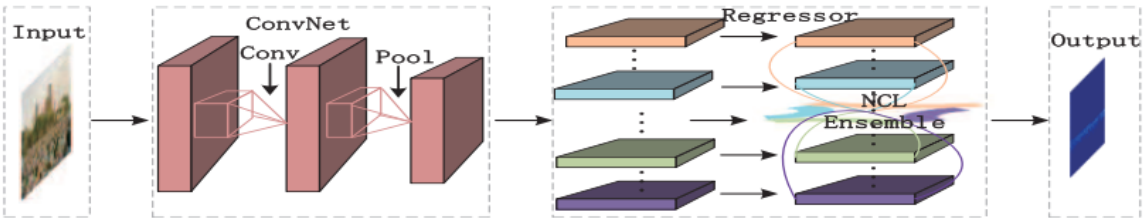



*Figure 13 Architecture of Convolutional neural network[50]*

With the use of residual error estimates, [35] created a novel crowd counting method that could gradually produce crowd density maps. The proposed technique makes use of the VGG16 as the backbone network and uses the density map produced by the final network layer as a coarse prediction to gradually improve and provide a fine quality of density maps by utilising residual learning. An uncertainty-based confidence weighting technique that permits the flow of just high-confidence residuals in the refining path also supports residual learning. The suggested confidence-guided deep residual network for crowd counting has been evaluated on various datasets which has shown a notable improvement in results by showing an MAE of 66.1 and MSE of 195.5. A new crowd counting dataset, JHU-CROWD, which has 4250 crowd photos and 1.11 million annotations, has also been donated by the researchers in addition to the development of a network for crowd counting. The fact that this dataset was compiled in a variety of different circumstances, including deteriorated weather situations and illumination fluctuation, makes it far more difficult to use than other datasets.

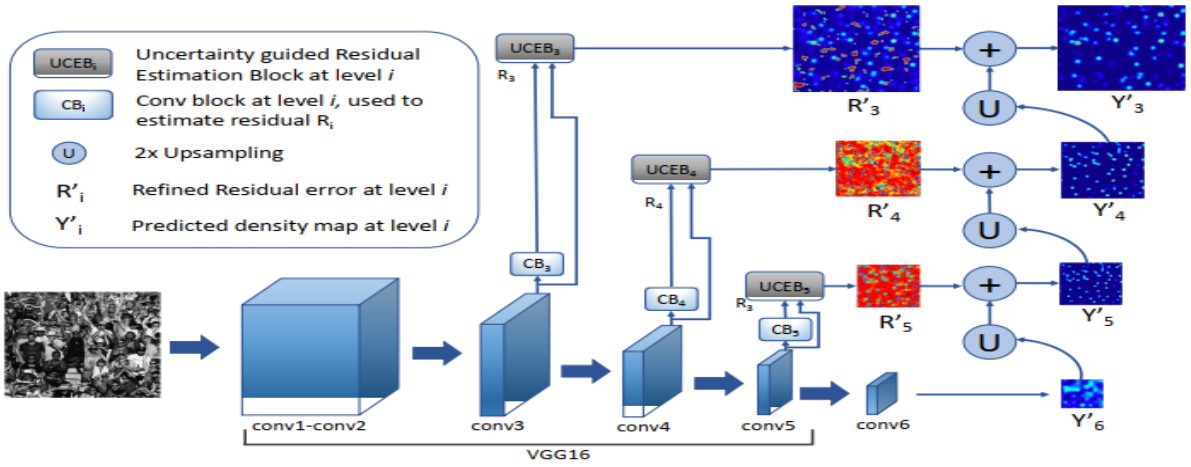



*Figure 14 CNN Architecture Used by [35]*

Wan and Chan, (2019) demonstrated that crowd counting is a fascinating application in computer vision by keeping in view its applications in security applications such as surveillance management. Most crowd counting methods operate in two parts, the first of which involves creating a ground-truth levelling density map. The second step involves deep learning prediction models creating new density maps, which are then compared to the first. Convolutional neural networks are effective in producing density maps, although the issue of density map production has not been resolved. In context of end-to-end-training, manually generated features for making

density maps may not be enough for a dataset. This research suggests a method to get around this problem by first demonstrating the effects of various density maps and then demonstrating that better ground-truth density maps can be acquired by improving the current ones using a learned refinement network, which is collectively trained with the counter. Following this, an adapting density map generator is created, which learns a density map depiction for a counter using the annotated dot map as input. Within an end-to-end structure, the counter and generator can both be educated simultaneously. Experimental findings show the superiority this this approach by attaining an MAE of 64.2 and MSE of 99.7.

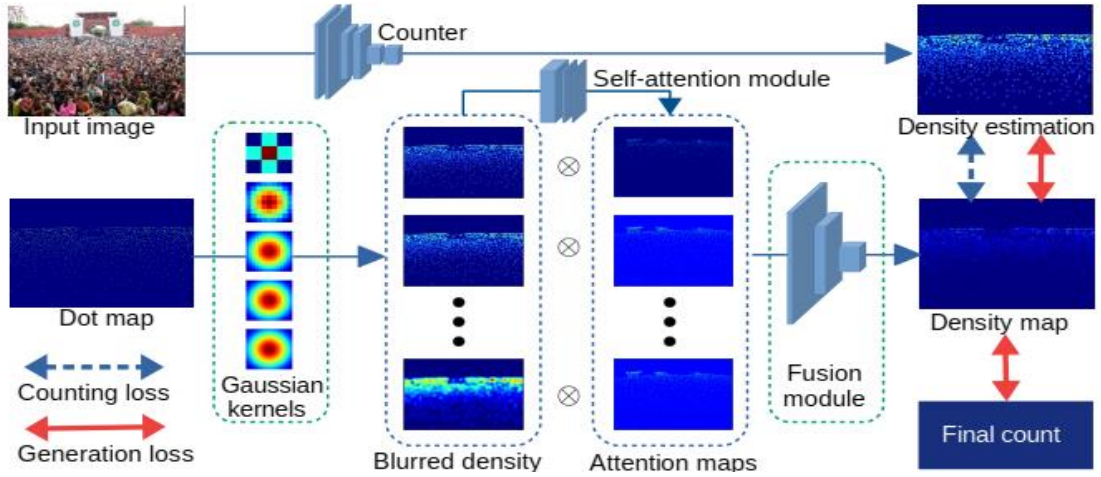



*Figure 15Density Estimation Framework Used by[52]*

[23] produced an adversarial learning approach for applying crowd counting multi-scales in complicated environment scenarios. In this study, a multi-scale GAN (MSGAN) model was developed, in which a convolutional neural network (CNN) serves as a generator and an adversarial network serves as a discriminator to count individuals in complicated scenarios by supplying a high-quality density map. In this work, a multiscale generator is used to combine traits in several hierarchical layers to discover people with large-scale variation. The density map produced by the multi-scale generator is processed by a discriminator that has been trained to perform a binary classification task between a low-quality density map and genuine ground truth ones. The proposed approach enhances the neural network's capacity for density estimation, producing precise crowd counting in still photos or moving pictures. This study also demonstrates that the recommended method beats all other traditional and modern methods in terms of accurate object counting and high-quality density estimate.

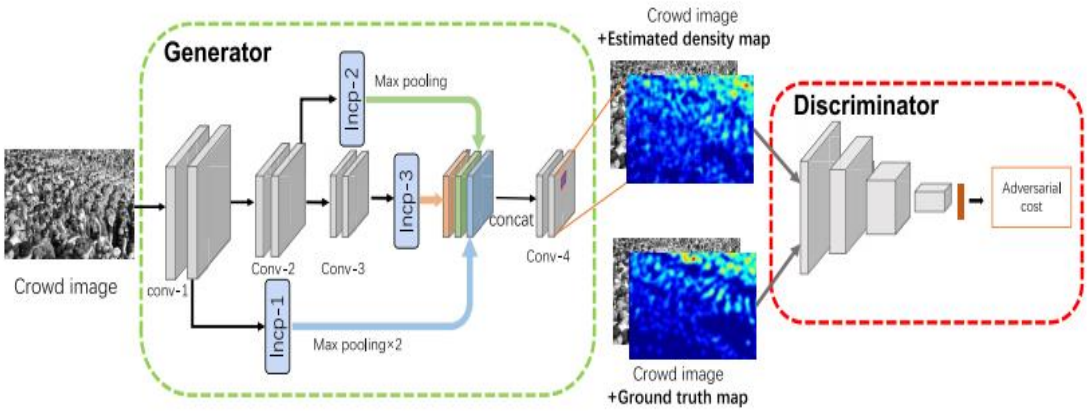



*Figure 16MS-GAN Architecture by [23]*

Crowd counting has developed into an important and fascinating area of application that has drawn computer vision and AI researchers to make new discoveries. [24] provided a systematic literature review of the techniques of computer vision and deep learning in pedestrian crowd counting. By bearing in mind the context of video surveillance, human-machine interaction, and more, vision-based pedestrian tracking systems are developed. Additionally, this study has examined both 2D and 3D systems while taking into account the stochastic nature of this field of study. Convolutional neural networks (CNNs), which are crucial for image-based data, are described as recommended and anticipated methods for handling videos and pictures for pedestrian tracking. Finally, this work also presents classification-based approaches on a variety of datasets to demonstrate the importance of pedestrian tracking.

[19] stated that using semi-supervised approaches for crowd counting have attained a notable attention of researchers due to expensive behaviour of supervised-based approaches which demand a huge number of images and dense crowd scenarios. For this purpose, this research has presented a 'spatial uncertainty aware semi-supervised' network for the purpose of crowd counting. The suggested spatial uncertainty aware teacher-student framework is distinct from current semi-supervised learning-based crowd counting techniques and for oppression of unlabeled data because it is focused on high confidence information of regions while also fully addressing the noisy

supervision from the unlabeled data. Particularly, the estimate of spatial uncertainty maps from the surrogate task of the teacher model to direct the feature learning of both the student model's surrogacy work and the primary task, also known as density regression. In the student model, a quick and efficient differential transformation layer is created to enforce the primary function's built-in regularization of spatial consistency. This layer helps the surrogate task produce more accurate prediction results and high-quality uncertainty maps. The approach can also solve the task-level perturbation issues caused by spatial inconsistencies between the principal and surrogate tasks. This strategy produced an MAE loss of 68.5 and an MSE loss of 121.9, which are considered to be noteworthy and outperforming.

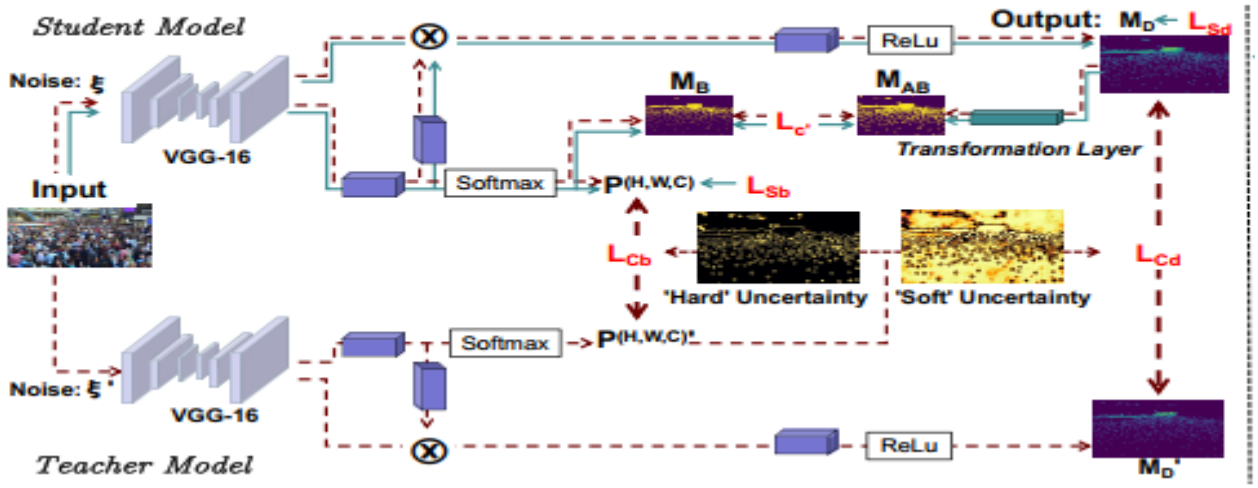



*Figure 17Spatial Uncertainty Aware Approach for Crowd Counting by [19]*

[21] introduced a unique perspective-guided convolution, also known as (PGC) for convolutional neural network (CNN) referred to as PGC-Net, for use in crowd counting. The substantial intra-scene size differences of persons in an image caused by the perspective effect are addressed by this concept. Even though the majority of cutting-edge methods employ many scales or columns to address this problem, they typically fall short when modelling continuous scale variation since only finite representative scales are taken into account. The PGC-Net model, which is what is being presented in this study, makes use of the pertinent data to properly direct the spatially-variant smoothness factor of the feature maps before feeding them into subsequent convolutional procedures. In addition to all of these, PGC-Net has an effective perspective estimate component that, if pre-trained, may be trained in either supervised mode or moderate supervised mode. The

suggested PCG-Net works a less increase in demand of computational power and after extensive experiments on benchmark datasets, it turns our that this model beats other state-of-the-art techniques.

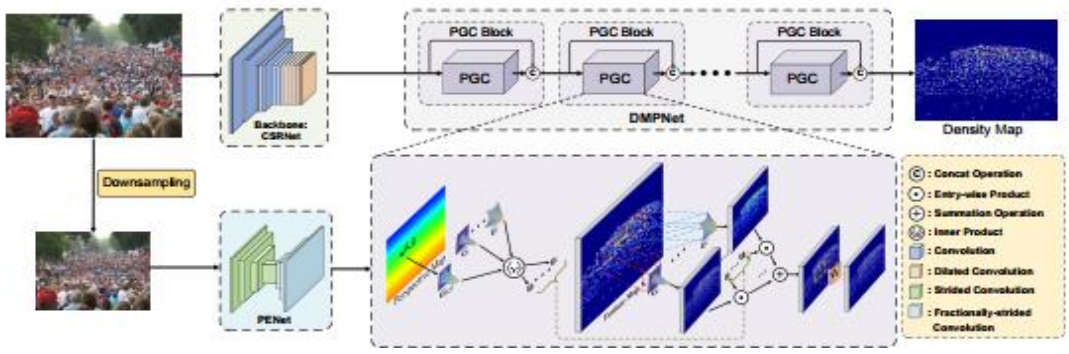



*Figure 18Proposed PGC-Net Architecture Developed by [21]*

[15] elaborated that the topic crowd counting has attracted a lot of attention due to its wide range of applications such as controlling congestion, security of public and in many other areas. The shortcoming in currently employed deep neural networks for crowd counting is that they become too difficult to manage while training due to their huge size and a lot number of parameters, plus taking a lot of time to train efficiently. This article has provided a light model to handle this issue that consists of an image feature encoder and a straightforward yet efficient decoder known as a pixel shuffle decoder, or PSD. A pixel shuffle operator developed by PSD can display higher density data without adding extra convolutional layers. To fully leverage the capability of crowd counting models, a density-aware curriculum learning training mechanism is included in the second stage. Each predicted pixel was given a weight by DCL to determine its predictability difficulty and to aid in developing a better generalisation on hypothetical situations. After attaining an MSE of 107.96 and MAE of 64.97, the experimental findings of this approach has shown benchmark performances.

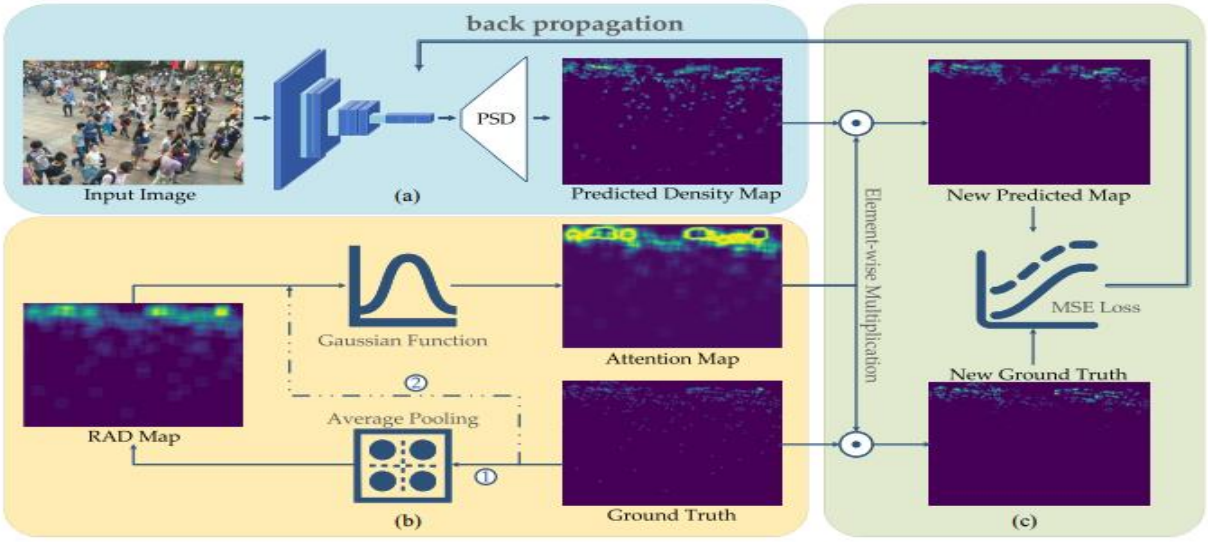



*Figure 19The Architecture of DCL Network Proposed by [15]*

[51]The EfficientDet network, which was built by drawing inspiration from the original EfficientNet network for image classification, was offered as a new network for object detection. It was scalable and effective enough for detection. Because model efficiency has become critical in computer vision applications, especially in object detection, architectural design decisions for this application need to be improved. For this purpose, a weighted bi-directional feature pyramid network (BiFPN) is proposed for quick and simple multi-scale feature fusion. The resolution, depth, and breadth of all support, feature network, and box, class prediction networks are evenly scaled using a second compound scaling approach that is devised. A new class of object detectors called EfficientDet is created which is based on these improvements with EfficientNet outperforming earlier methods under a variety of resource limitations. On COCO testing dataset, the EfficientDet D7 network has attained 52.2AP having a total of 52 million parameters and having 325 billion FLOPs. Comparing the network to earlier detection models, it is 4 to 9 times lighter and utilizes 13 to 32 times less FLOPs.

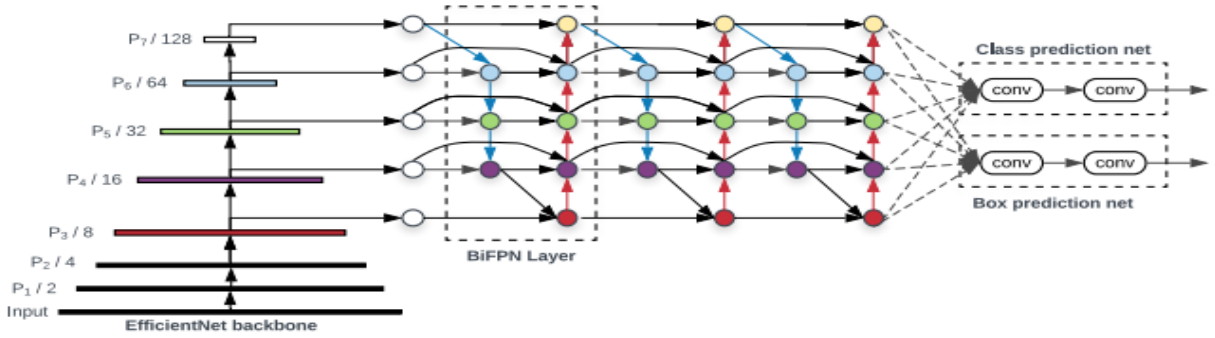



*Figure 20* *The EfficientDet Architecture Design by [51]*

[27] performed evaluation of EfficientNet model for object detection in context of indoor robot assistance navigation. Because of the growing number of applications and, in particular, the development of intelligent robots for a variety of tasks such as helping blind or visually impaired people and in healthcare systems, indoor object detection and recognition has emerged as one of computer vision's most captivating fields of study. The task of intelligent robot navigation still remains challenging because it includes different procedures such as vision, understand and recognition. This study examines and takes into account the implementations of low resource mobile robots in order to offer a vision-based detection system based on EfficientDet neural network that may address this problem. Weights pruning technique has been used to ensure a lightweight implementation of the suggested interior objects detection system and to design a system that could be used in mobile robot applications. A pruning strategy is used to contribute to an embedded execution of the suggested system, which significantly decreases the network size, complexity, and computing costs. The experimental findings of this approach ensure its robustness by achieving a score of 89% on testing data by using EfficientDet D2 network.

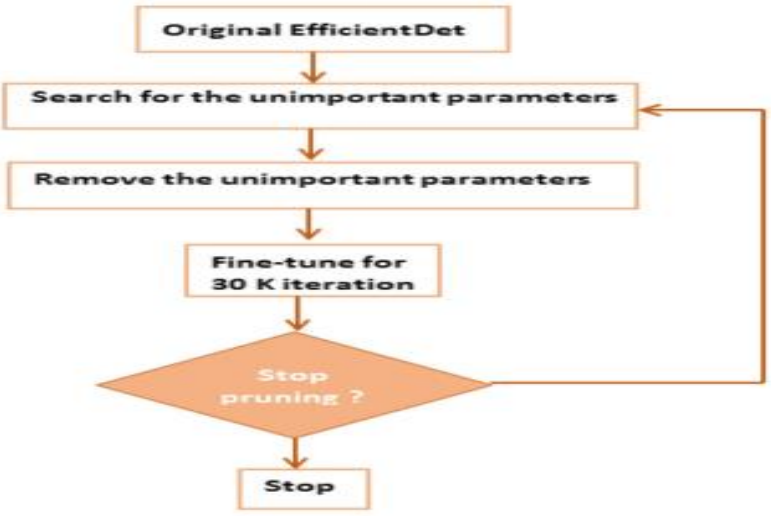



*Figure 21Method Used by [27]*

[28] Data augmentation has become a standard part of training elevated deep image classifiers, but its potential for object recognition has not been explored. Most modern object detectors improve from fine-tuning a pre-trained classifier, this study looks at how different data augmentations for classifiers are applied to object detection. The approach that is suggested in this study may dynamically choose the more potent adversarial pictures that come from the classification and localization detector branches, and it changes as the detector does to keep the augmentation policy up to date. Compared to Auto-Augment, a model-agnostic augmentation strategy searched based on one specific detector, this model-dependent augmentation is better able to generalize different object detectors. The suggested method enhances the performance of cutting-edge EfficientDet on the COCO object identification benchmark dataset by +1.1 mAP. Additionally, it improves detector resilience by 1.3 mAP against domain shift and by 3.8 mAP against natural abnormalities.

By emphasizing that each picture in a crowd count is represented by a dot and that each technique employs a gaussian methodism for likelihood estimate of each annotation, (B. Wang et al., 2020) implemented distribution matching mechanism on crowd counting. This study demonstrates how the performance of generalisation is impacted by applying Gaussians to annotations. Instead, the paper suggests using Distribution Matching for crowd counting (DM-Count) to count crowds. The relationship between the normalized predicted density map and the normalized ground truth density map is determined in DM-Count using Optimal Transport (OT). A Total Variation loss is also incorporated into this model in order to perform a stable OT computation. This study also shows that the density-matching network presented here has less generalisation error than

conventional gaussians. In terms of MAE and density-matching ratio on prestigious datasets like UCF-QNRF and NWPU, by yielding an MAE of 85.6 and MSE of 148.3, the experimental findings of this work have set benchmarks and reduced the error rate by 16 percent.

[29] highlighted that with recent advancements in deep learning and computer vision, many applications have been come in front, especially crowd counting due to its wide range of applications. In this work, a specific type of crowd counting known as counting based on aerial photographs taken at low altitudes by an unmanned aerial vehicle, or UAV, is provided. To discover ones suitable for on-board image processing employing edge computing devices while reducing the performance loss, a variety of neural network topologies are tested. This study also shows that the input picture resolution strongly affects the prediction quality and should be taken into account before using a more complicated neural network model to increase accuracy. This is based on a wide variety of trials on neural network designs. Moreover, this research also underlines that using more complex networks in this case could also result in overfitting since scarcity of training data may happen.

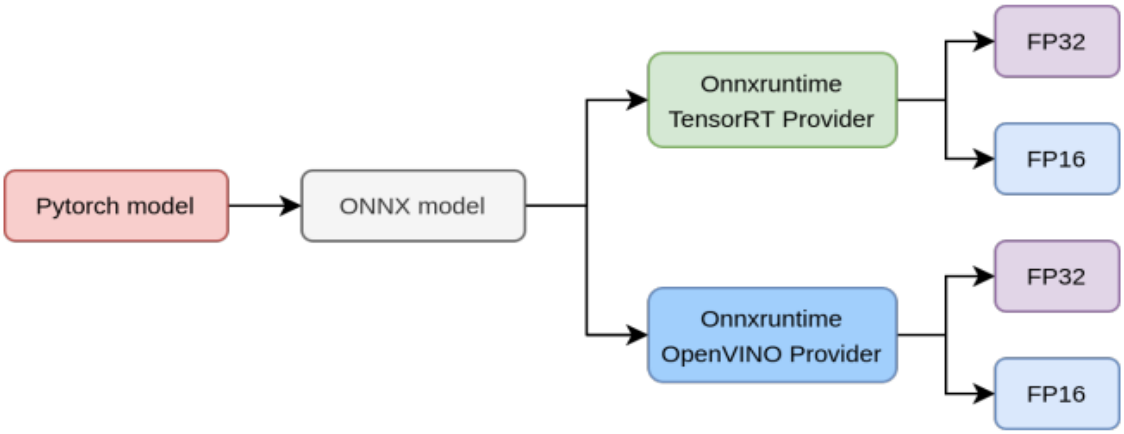



*Figure 22 The Crowd Counting Pipeline Given by [29]*

[35] used multi-level-based bottom-top and top-bottom approach for applying crowd counting which consisted of fusion of image features. Crowd counting has come up with many challenges which mainly happen in case of densely crowded scenes. Moreover, traditional approaches used for this task such as multi-scale feature fusion seem to be too simple and sometimes fail to yield targeted results especially in dense crowds. This mainly happens due to the lack of ability of these fusion models to accurate combine features and is considered one of the key problems faced in crowd counting. This research focuses on how to effectively exploit data existing at various

network tiers in order to get around these crowd counting issues. This study's network uses a multilevel bottom-top and top-bottom fusion (MBTTBF) technique to combine data at different levels from shallower to deeper layers and vice versa. Second, explicitly permit the flow of complementary features from neighbouring convolution operation along the fusion routes by scaling supportive feature extraction blocks (SCFB) utilising cross-scale residual functions. After conducting extensive experiments on three benchmark datasets such as ShanghaiTech, UCF_CROWD_50 and UCF_QNRF, the model yielded the best MAE and MSE score of 60.2 and 94.1 on ShanghaiTech dataset.

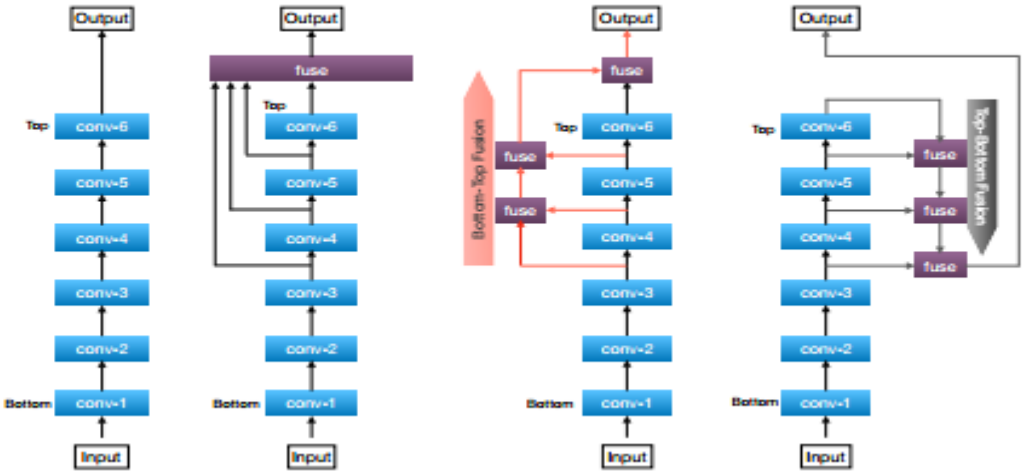



*Figure 23Neural Network Architecture Used by [35]*

(Liu et al., 2019) used recurrent attentive zooming for crowd counting by saying that crowd counting has come a significant application in computer vision sue to its applications in safety management. Traditional methos for crowd counting work by first creating a density map and the final crowd count is then calculated by counting the number of dots that appear in the density map. This paper has suggested a framework to address both of these issues, namely density estimation and localization, in order to solve the problem. A network developed by the research known as recurrent-attentive zooming is capable of recognizing confusing picture parts and zooming it for re-inspection. Here, a framework for reinforcement is also created, via which localization and counting are reinforced for improved performance. This paper has provided a strategy for addressing this issue that addresses both the localization and density estimation issues. The study led to the creation of a network known as recurrent-attentive zooming, which can recognise

confusing visual portions and zoom them for re-inspection. A reinforcement framework is also produced here due to which counting and localization reinforce each other for better performance. The tests of this experiment are made on datasets like UCF_QNRF and results show MAEs and MSEs of 65.3 and 108.8 on ShanghaiTech A and 119 and 198 on UCF_QNRF dataset.

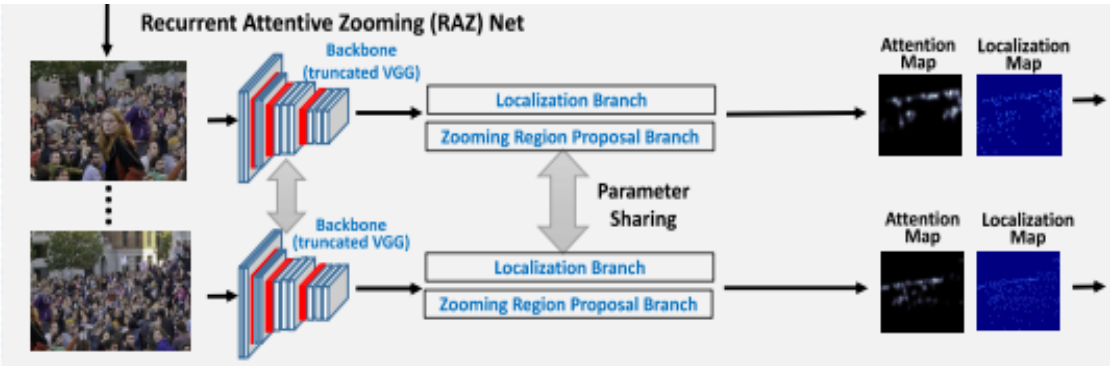



*Figure 24 Recurrent Attentive Zooming by [43]*

[32] presented a convolutional neural network (CNN)-based architecture that tackled the issue of density estimate in crowd counts. Two CNN models have been developed for producing high-quality density maps in light of the difficulty in producing such maps. The proposed network consists of two branches, the first of which uses CNN to build a low-resolution density map and the second of which uses CNN to create a high-resolution density map using the first branch's low feature representation of density maps. This method may also be extended such that each stage of a pipeline can make use of the model predictions from earlier stages. The performance of this model yielded an MAE of 260.9 and RMSE of 365.5 on UCF Crowd Counting Dataset.

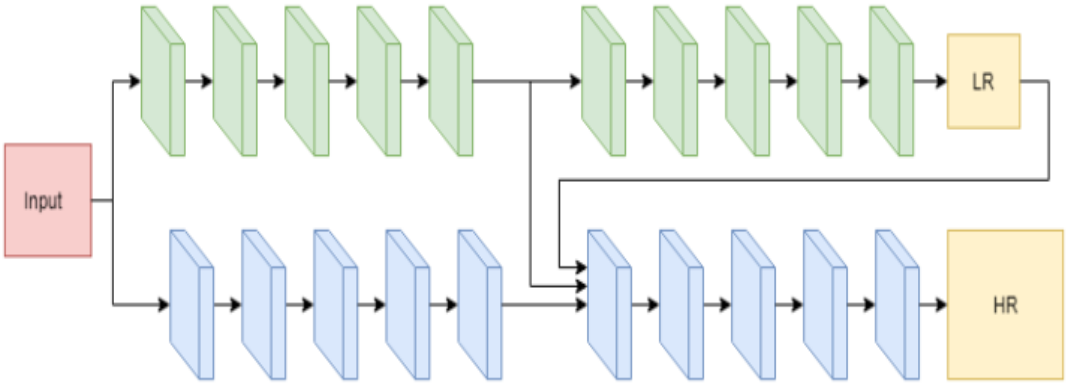



*Figure 25CNN Model for Crowd Counting by [32]*

In a research work contributed by [36], relational attention network was produced for crowd counting. Crowd counting has attracted a lot of interest from academics because of its many uses in a range of businesses. The downsides of this approach include things like variable backgrounds, limited resolution, and more. Density estimation methods are also a solution to this problem, but the pixel-wise regression method used there skips the relative importance of features and thus those pixel-wise predictions could be noisy and inaccurate. This research has introduced a Relational Attention Network (RA-Net) with a self-attention methodology for collecting pixel interdependence to alleviate this issue. The RA-Net improves the self-attention process by taking into consideration the interconnectedness of pixels over both short and long distances. Both of these applications are known as local and global self-attention. In addition, a relational module to combine local and self-attention to provide feature representations that are more meaningful and consolidated. Datasets like ShanghaiTech, UCF CC 50 and UCF QNRF are used to test the performance of Relational-Attention neural networks. These datasets have absolute and squared errors of 59.4 and 102.0 on ShanghaiTech, 239.8 and 319.4 on the second dataset, and 111, 90 on the third dataset, respectively.

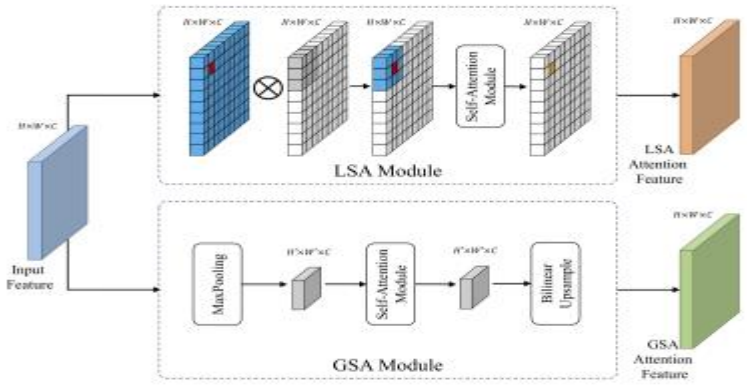



*Figure 26 Relational Attention Network by [36]*

[40]highlighted that imbalanced data destitution in crowd counting can lead to inaccurate measurements and can lead to over or under-estimation which needs to be tackles. This study provides a straightforward yet efficient locality-based learning paradigm to develop generic features by mitigating sample bias in order to address this difficult topic. In two ways, the suggested technique is locality-aware. A locality-aware data partition (LADP) strategy is started in the first phase to divide the training data into several categories using locality-sensitive hashing. As a result, LADP then creates a data chunk that is more evenly distributed. A novel data augmentation technique called locality-aware data augmentation (LADA), in which the picture regions are dynamically enhanced based on the loss, is developed for eliminating training skew and improving the coordination with LADP. The proposed method could be merged with many crowd counting applications for the sake of extending their performance. For making sure that model is capable enough to generalize on datasets, a wide range of experiments have also been conducted which have produced high-quality results.


*Figure 27 Data Augmentation principle for crowd count by[40]*

For making detection of pedestrians, [31]developed a modified version of a popular object detection model called as EfficientDet, to Fast EfficientDet. Pedestrian detecting and tracking has become crucial in today's world due to a wide variety of reasons such as surveillance management, criminal investigation, and so on. When faced with scenarios like multi-scale pedestrian monitoring, existing techniques' accuracy scores suffer from consistency issues. Fast EfficientDet, an updated variation of the multi-scale object identification EfficientDet model that extends the background functionality of the model and updates the depth-wise separable convolutional operations that have an impact on training, is provided as a solution to this problem. A new activation function known as the Mished activation function is introduced to expedite model training. The second stage proposes an enhanced feature pyramid-network Skip-BiFPN. An upgraded feature pyramid-network Skip-BiFPN is suggested in the second stage. A cross-layer data flow is created on the basis of this network to merge the semantics and geographic details of the item. The suggested network can better recognise items with significant size changes in complicated situations. The NMS post-processing procedure culminates with the introduction of the DIoU calculating technique. Talking about experimental results of this approach and after making a comparison of this network with traditional EfficientDet networks, it produced a score of 84.96% and also depicted a training speed to be increased by 15%. After going through this study, it can be concluded that for enhancing the speed of EfficientDet, the depth-wise separable convolutional needs to be updated.

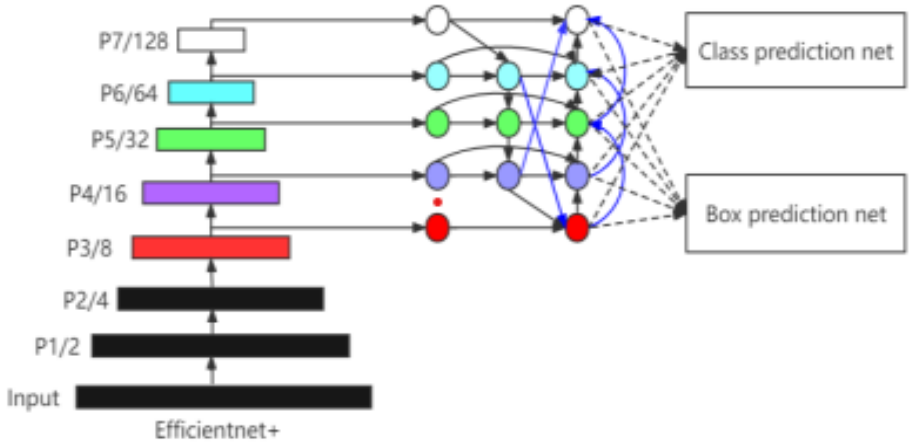



*Figure 28 Fast-EfficientDet Produced by [31]*

[38] employed an encoder-decoder based network for crowd counting and density estimation in images and videos. Many computer vision experts have found crowd counting to be fascinating, but it is still a difficult and difficult operation to complete. In order to produce an accurate density estimation map, the encoder-decoder network for crowd counting is presented in this study. The four primary steps of the recommended strategy are first creating a trellis architecture that accepts numerous decoding pathways; hierarchically aggregating picture characteristics at various levels of encoding; and implementing the suggested approach. In the next stage, dense skip connections are interspersed between paths to help with enough multi-scale feature fusions, which also aids the TED-network in processing the supervision data. The third stage involves applying a new combinatorial loss to ensure similarities in local coherence and spatial correlation amongst maps. The distributed application of this combinatorial loss on intermediate outputs allows the TED-network to do back-propagation while mitigating the vanishing gradient issue. On checking the capability of this network on four benchmark datasets, this model has showed an improvement in the MAE loss by 14%, which is claimed to be beating state-of-the-art.

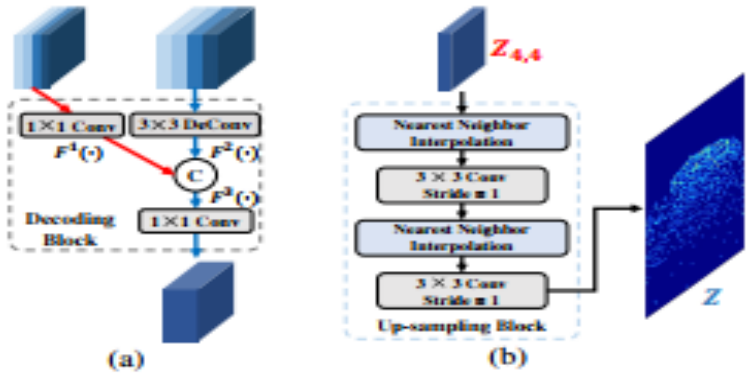



*Figure 29 Proposed TED-Net Model Used by [38]*

Computer Vision (CV) and deep learning (DL) are the branches of artificial intelligence (AI) that have been used in wide range of applications, like in terms of dealing with complex tasks such as crowd counting. Though these two fields of AI are separate from each other, they are used together in making machines intelligent enough to take verdict on images. Counting people in a crowd is a challenging task since it requires both visual understanding of an image to make a bounding box around that and then counting those bounding boxes to make the final crowd count in an image or video. There are also other fascinating applications of computer vision such as image classification, object detection and recognition, facial recognition, detecting emotions from faces and so on. Researchers prefer challenging areas in computer vision to on, and crowd counting is one if those areas where researchers can put their feet to start researching on a topic like crowd counting. Systems backboned with crowd counting can be used for various tasks such as for security surveillance, counting how many people stepped inside a shopping mall, for example and how many persons did go out, counting how many people crossed a particular street or how many people did cross the road at what time, all happen under the umbrella of crowd counting applications.

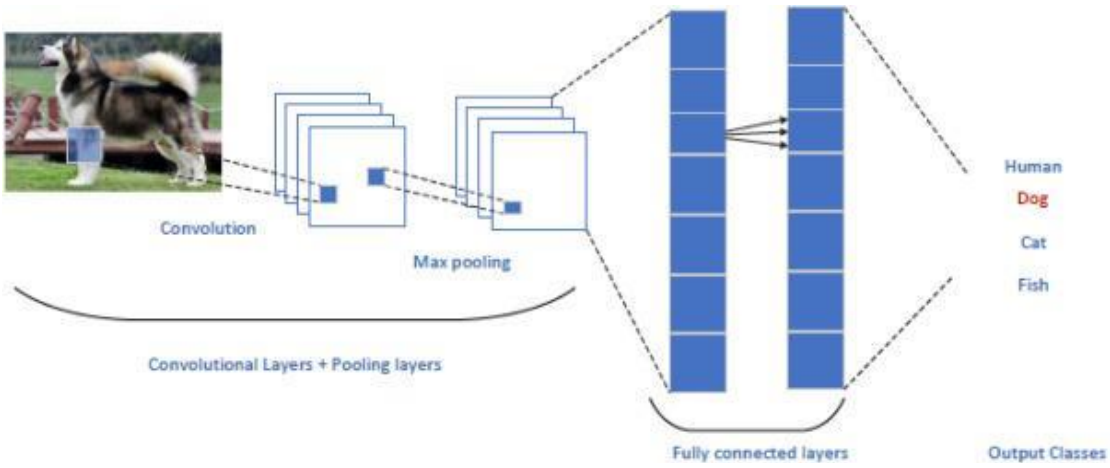



*Figure 30 The Paradigm of Deep Learning in Computer Vision*

[15] provided a comprehensive survey on CNN-based density estimation methods for crowd counting. The research has indicated that the task of counting the number of people in an image is a challenging task but it is widely used in may real-world applications such as public safety and planning systems. This domain can also be applied to other areas such as vehicle counting. Various researchers have worked in crowd counting and have done good literature work and they can aid in developing crowd counting systems. This paper has sorted out the best models for crowd counting by making a systematic survey of more than 200 papers to comprehend the models used for crowd counting while focusing on convolutional neural network (CNN) based model for density estimation. Top works held on crowd counting have been finalized and future developments in crowd counting are discussed. This research has also given a clue that optimized crowd counting solutions could also be applied to other fields. Finally, CNN-based density maps and model predictions are also elaborated by focusing on dataset like NWPU, and evaluation metrics are also provided.

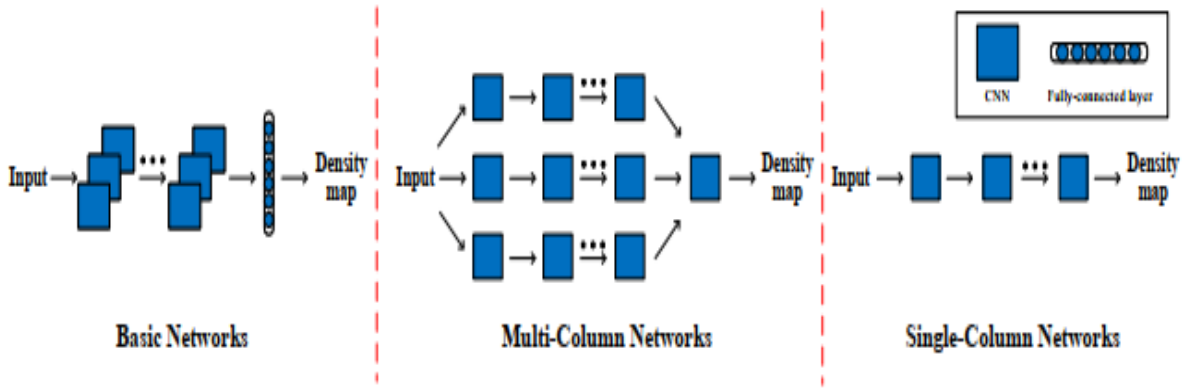



*Figure 31 Comparison of Various CNN Models by [15]*

[13] used a convolutional neural network (CNN) with a crowd attention module to do crowd counting. The study shows that despite certain advancements in CNNs, there are still some issues with these networks, such as the tendency to wrongly identify some individuals as objects, which ultimately results in a higher error rate. The study presents a brand-new architecture known as a crowd attention neural network that gives a CNN more attention. The CNN's attention module gives human heads more attention, making it easier to identify people in a crowd. Extensive tests are conducted to evaluate the performance of the neural network suggested in this study using challenging datasets. Extensive tests are evaluated and the results of the neural network suggested in this study using challenging datasets. This method is said to produce findings that outperform crowd counting methods that have previously been used.

An IoT-enabled smart system for counting pedestrians and ambient tutoring in a smart city was presented by[44]. The system that is being suggested in this study is made up of manufactured system nodes that count the number of people passing through a location and compute their direction of travel using a few environmental data. Things like temperature, humidity, carbon dioxide, etc. are taken into account while calculating ambient parameters for the system. A wide area network is used to connect the server with the data collected by sensors. This system's sophisticated algorithm has produced pedestrian counting results with noteworthy accuracy. A total of more than 70 nodes have been deployed, with a six-month expected operating time.

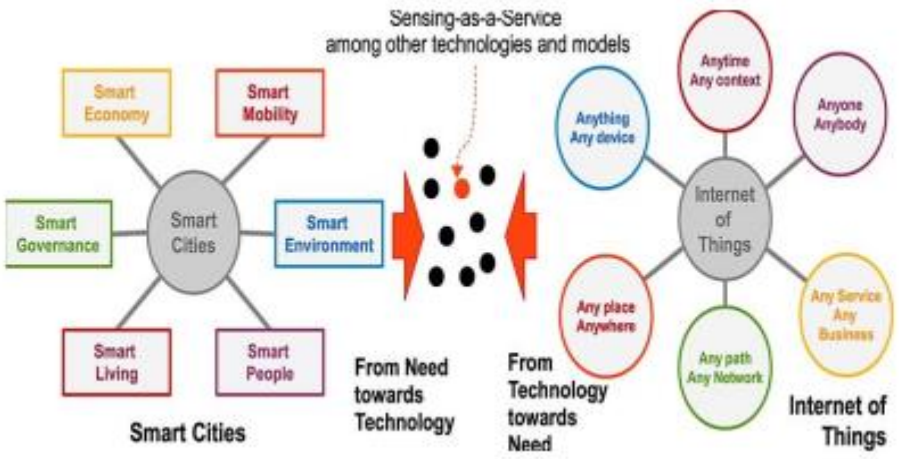



*Figure 32 A Pictorial Representation of IoT based System for Pedestrian Counting[44]*

For dealing with challenging problems like crowd counting, various datasets have been published which are used by researchers to undergo their experiments on crowd counting, analysis, density estimation, detection, and so on. A popular dataset, called as ShanghaiTech dataset, is one of the most famous and widely used datasets for crowd counting used by researchers. This dataset has a total of 1198 crowd annotated images and this dataset has been divided into 2 parts, i.e., ShanghaiTech A and ShanghaiTech B. Part A contains 482 images and B contains a total of 716 images. The train-test distributions among datasets show that in part A, 300 images are put in training set and 182 images are placed in test. B part has a train/test split of 400 and 316. Each person in both datasets is identified by a point and this there are a total of 330, 165 annotated people in the dataset.

[30] provided a comparative study of object tracking and detection algorithms for vehicles. This study has indicated that rapid advancements in deep learning-based systems have increased the scope of vehicle counting systems. To identify and track distinct kinds of cars in their region of interest (ROI), the researchers used several cutting-edge object identification and tracking algorithms in this study. Getting an exact count of cars is the goal of accurately identifying and tracking the vehicle in its ROI. To provide the ideal framework for vehicle counting, several combinations of object identification models and other tracking technologies are also used. Through its computationally dense training and feedback cycles, the model effectively extracts location of the vehicle and trajectories while addressing the problems associated with various

weather conditions, occlusion, and low-light environments. Vehicle counts were calculated from the combination of all models are validated and compared to ground truth values and experimental evidences show that convolutional models such as CenterNet and DeepSORT and YOLO V4 and DeepSORT provided the state-of-the-art results.

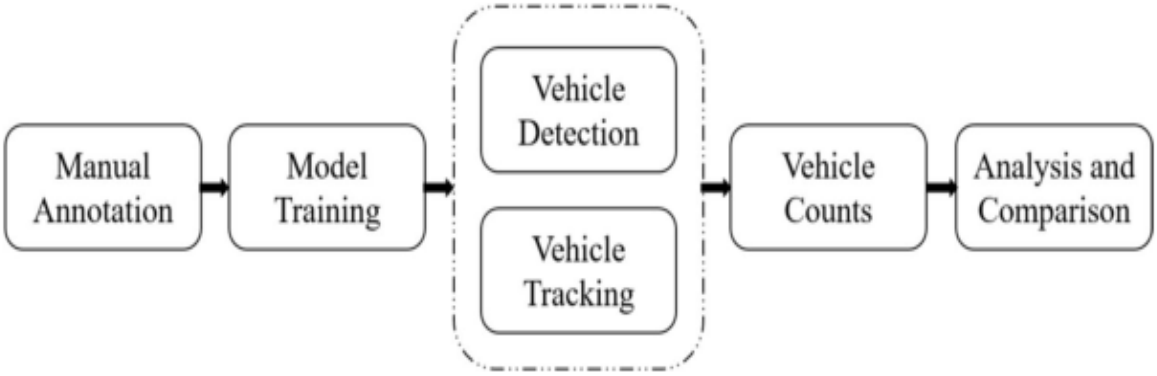



*Figure 33 Comparative Analysis Technique by [30]*

Artificial Neural Networks (ANNs) are a class of artificial intelligence that have been widely employed in various tasks based on their structure and domain. ANNs have outperformed classical machine learning algorithms due to their ability to work and perform well on big and complex datasets. The paradigm of artificial neural networks is very generalized, i.e., it contains a lot of neural networks such as feed-forward networks (FFNs), convolutional neural networks for working on images, recurrent neural networks (RNNs) to work on text data, deep belief networks (DBNs), generative adversarial networks (GANs) to generate realistic images, and so on. This research is also inspired by this factor since convolutional networks work exceptionally well on the task of crowd counting. Feed forward networks are most commonly used type of ANNs that are used for classification and regression. According to (Wu and Prasad, 2017), a feedforward neural network can also have a reclusive layer, or it'd have hidden layers.

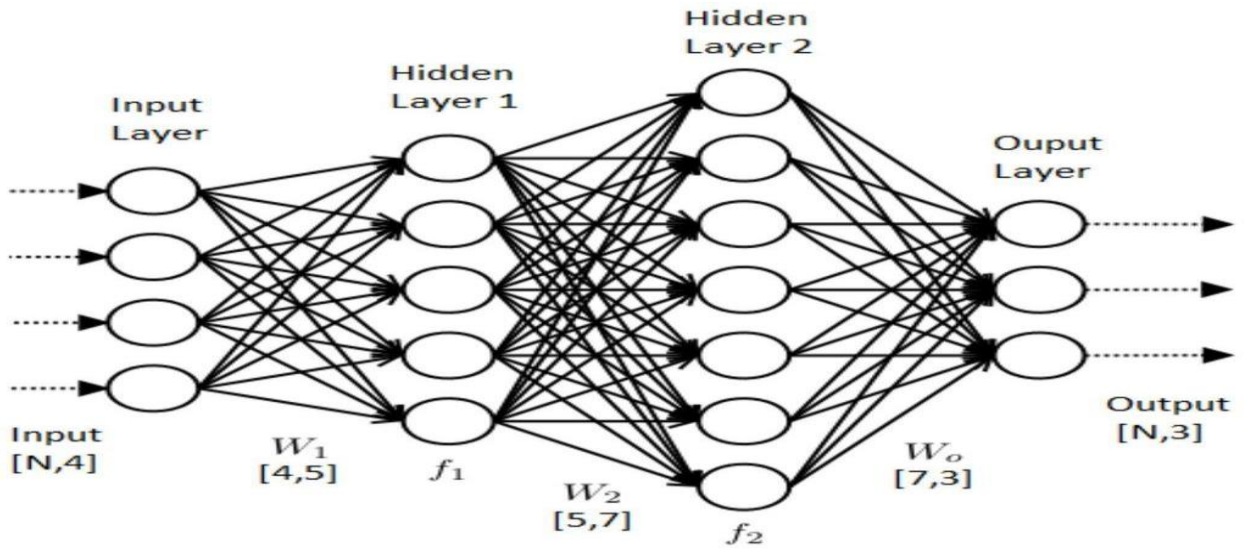



*Figure 34 The Base Architecture of an Artificial Neural Network (ANN)*

[45] developed a crowd counting system using single-image via multi-column convolutional network (MCNN). The task of MCNN network is to map the input image to it target density map. The input picture can be any size or resolution using the specified MCNN in this study. The characteristics learnt by each column CNN are adaptable to differences in individuals and their head size owing to perspective effect or picture resolution by using filters with receptive fields of varied sizes. Additionally, geometry-adaptive kernels, which do not require knowledge of the input image's perspective map, are used to precisely generate the real density map. Existing crowd counting datasets lack the ability to cover all challenging things, therefore, the researchers have made a new dataset which is composed of 1198 images with 330,00 annotated heads. Experiments were conducted on previous as well as the new dataset, the ability of proposed MCNN is tested and it is found it the suggested model outperformed all other existing models at that time. It was also found that this model, once applied to a dataset, could also be applied to another dataset.

| *Paper* | *Year* | *Headline* | *MSE/MAE (%)* | *Journal* |
|---|---|---|---|---|
| *Miao et al., (2019)* | 2019 | ST-CNN Spatial Temporal CNN for Crowd Counting in Videos | 13.79 MSE 9.25 MAE | Elsevier |
| *Li et al., (2021)* | 2021 | Approaches on Crowd Counting & Density Estimation – A Review. The research outlines CNNs to be the best for this task | - | Springer |
| *Hossain et al., (2019)* | 2019 | Crowd Counting Using Scale-aware Attention Networks | 28.41 MSE 16.86 MAE | IEEE (Conference) |
| *Zheng et al., (2020)* | 2020 | DSP-Net (Deep Scale Purifier) for Dense Crowd Counting | 14.0 RMSE 8.9 MAE | Elsevier |
| *Wang et al., (2020)* | 2020 | Spatial Context Learning Network for Congested Crowd Counting | 102.94 MSE 67.89 MAE | Elsevier |

**Table 1:** Comparison Table of Previous Researches in Crowd Counting

# Chapter-3:Proposed Research Methodology

The proposed research methodology for undergoing an intelligent crowd counting system will be discussed here. The suggested intelligent approach in this section is divided into three phases, i.e., the input phase, the middle or central phase, and the functional phase. The functional phase will be the phase in which intelligent model will be applied.

The input phase comprises of data loading process in memory, which hints the starting of our approach. The dataset is consisted of images or videos and the input data is used for training process. Before inputting data to model, it must be transformed to numbers, which is done in pre-processing step.

To convert data to numbers, pre-processing step is used. The preprocessing phase comprised of all those procedures used to make our data ready to be input to a model. This includes data transformation, scaling, augmentation, etc. After completing the preprocessing step, the input data is ready to be forwarded to a neural network for intelligent detection.

The functional step comprises of inputting data to a neural network to perform intelligent detection and counting. For calculating the skill of model of how well it has learned from training data, loss function will be used and a finite number of training steps will be initiated. For achieving state-of-the-art results on unseen dataset, we'll be using a pre-trained model, for example EfficientDet, which is extensively employed for object detection in images and videos. We are opting to make an intelligent model for object detection that could be able to detect people in an image or video and count those people. Optimization algorithms & proper loss function will also be used to monitor the training process.

Below is a picture that describes a high-level representation of deep learning model (CNN) and its workflow, which can be also applicable to our research.

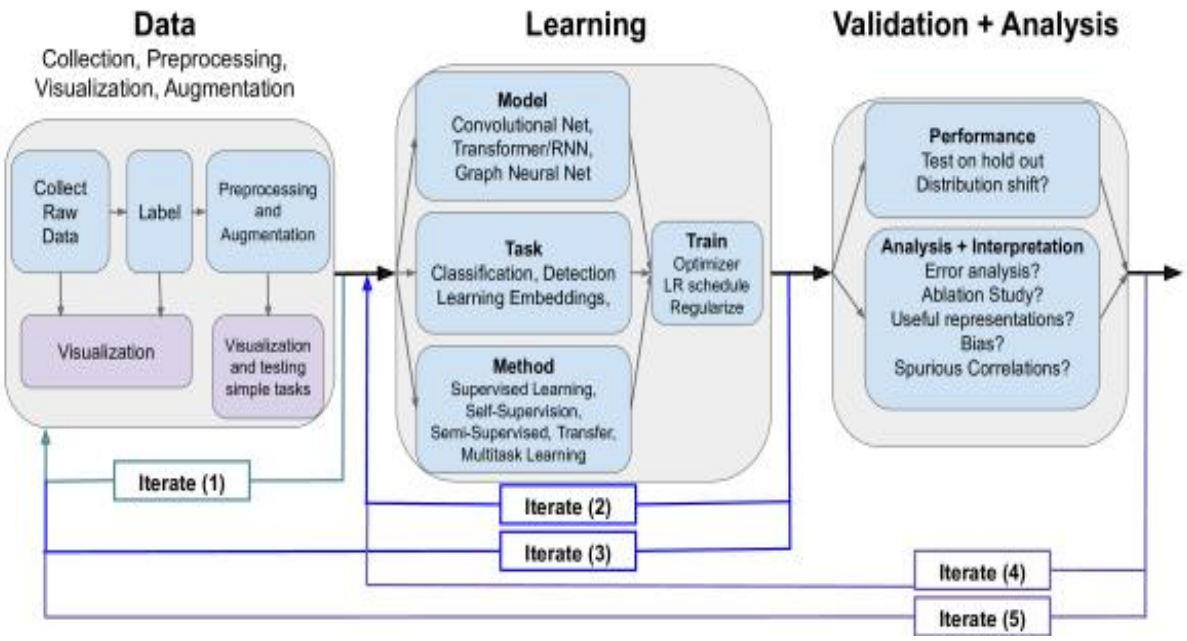



*Figure 35 Proposed Research Methodology*

Using neural networks over classical machine learning methods have numerous advantages. The key fact behind these is the layered structure of neural networks in which they pass information from one layer to another & choose best weights that optimize the problem. Convolutional neural networks (CNNs) are widely employed for image-related tasks & especially using a pre-trained model, which yields more accurate results. For image classification or object detection, these the information is transferred via fully connected layers.

For undergoing this concept of crowd counting, we'll be applying the concept of transfer learning (TL), in which a trained model (CNN) used for one task can be used for another task for the process of image classification, feature extraction, etc. The proposed model for this research is EfficientDet, a pre-trained model that is widely used for object detection. We can apply this to our problem for detecting objects in an image and then drawing a bounding box around them to count the number of people. Given below is an example of EfficientDet Model.

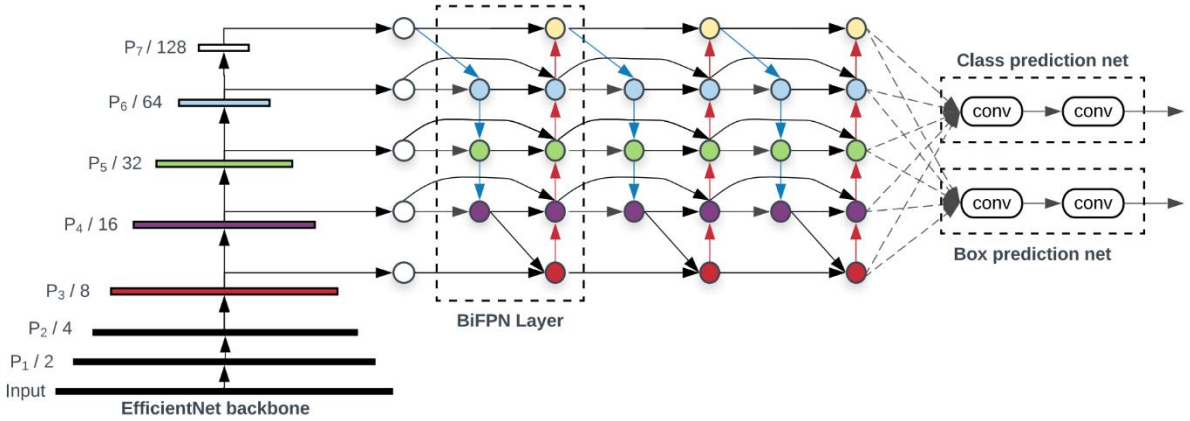



Figure 3: **EfficientDet architecture –** It employs EfficientNet [36] as the backbone network, BiFPN as the feature network, and shared class/box prediction network. Both BiFPN layers and class/box net layers are repeated multiple times based on different resource constraints as shown in Table 1.

*Figure 36 The EfficientDet Architecture*

The procedure described in above paragraphs described the proposed methodology for our research with an aim to count people in a crowd using deep learning CNNs.

# Chapter-4:Project Evaluation

The experimental findings & performance of the proposed model used to perform object detection for crowd counting will be discussed in this section. The performance of the deep learning CNN, namely as EfficientDet, Resnet will be calculated in this section which depends on the skill of model to detect objects with less amount of error. Moreover, Yolov5 will be implemented for object detection and counting.

### 4.1:Transfer Learning:

Transfer learning is a machine learning technique where knowledge gleaned from a model used in one task can be applied to another activity as a starting point.

Machine learning algorithms make predictions and generate new output values using past data as their input. They are primarily made to perform solitary duties. A task from which information is passed to a target task is known as a source task. When knowledge is transferred from a source task to a target task, learning is enhanced.

Transfer learning involves using the knowledge gained and quick advancement from a source task to enhance learning and growth for a new target work. Utilizing the properties and traits of the source task, which will be applied and mapped onto the target work, is how knowledge is transferred.

### 4.2:Problem Statement:

Crowd counting is a method for counting or estimating how many individuals are present in an image. Estimating the number of persons or objects in a single image accurately is a difficult but important issue that has been used in numerous applications, including public safety and urban planning. Due of its unique importance to social security and development, crowd counting is particularly prevalent in the numerous objects counting jobs. In the context of big data and the internet of things, the problem of object counting becomes intractable when you have videos tracking and you need to count the objects in every frame. The challenge of objects counting is challenging because you have to count all the specific objects on an image. An automated procedure, such as a machine learning algorithm, that takes an image as input and outputs the number of certain interesting objects in the image, can solve this problem (discrete value). For this,

a variety of methods may be used. The most common and straightforward is to use the quantity of elements in the image as a label and change it into a classification issue. Other ones employ a completely convolutional design, where the final output of the convolution can take into account the quantity of objects in that region before being included.

### 4.3:Dataset Description:

The dataset is made up of RGB images from video frame inputs that count the number of pedestrians (the object in the image) in each frame. A webcam in a mall captured three channels of 480x640-pixel photographs of the same location, but each channel showed a different number of people, creating a challenge with crowd counts. There are total 2000 images in dataset. We provide the image data in a binary NumPy file (.npy format) that can be loaded quickly using the NumPy load function.

The JPG photos are also available in a folder if you prefer to load each image individually. There is one label or target for each image that indicates the number of people on the frame JPG image. These are integer numbers for the counting; for instance, there are 29 objects in the image above (people).

Dataset is taken from this link: `https://www.kaggle.com/datasets/fmena14/crowd-counting`

### 4.4: Algorithms Used:

As our problem is concerned with detection and counting, so we have used the following algorithms:

- **ResNet Customized**
- **EfficientNetB0**

#### 4.4.1:Algorithm Description (Res-Net):

Residual Networks, or Res-Nets, train residual functions with reference to the layer inputs as opposed to learning unreferenced functions. Instead, then assuming that every few stacked layers will exactly match a specified underlying mapping, residual nets enable these layers to fit a residual mapping. By stacking residual blocks on top of one another, they can create networks; for instance, a ResNet-50 uses fifty layers of these blocks. Formally, by naming the required underlying mapping as and, we allow the stacked nonlinear layers to fit another mapping of. The original mapping is changed to. According to empirical study, these networks are easier to optimize and can increase accuracy over a much wider range of depths.

It was developed by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in 2015

**4.4.2:Algorithm Description (EfficientNetB0):**

Using a compound coefficient, the convolutional neural network construction and scaling method Efficient Net uniformly scales all depth, breadth, and resolution dimensions. Unlike traditional practice, which scales these variables freely, the Efficient Net scaling method uniformly adjusts network width, depth, and resolution using a set of preset scaling coefficients. To use additional CPU resources, for instance, we might simply increase the network depth, breadth, and picture size.

and are constant coefficients found on the original small model through a little grid search. To scale the network's width, depth, and resolution evenly, Efficient-Net employs a compound coefficient. The network needs more channels and layers to broaden its receptive field in order to capture more fine-grained patterns on the larger image, which is justified by the assumption that if the input image is larger, the compound scaling strategy makes sense. In addition to squeeze-and-excitation blocks, the foundational EfficientNet-B0 network is built upon the MobileNetV2 inverted bottleneck residual blocks.

It was implemented by AutoML MNAS.

**4.4.3:Algorithm Description (YoloV5):**

One of the most fundamental and challenging problems in computer vision is Object Detection, which is seen as the most important factor in the most recent research. It has been regarded as a historical snapshot of computer vision during the last two decades. The goal of the study is to assess the performance of YOLOV5 using multiple dataset types and data sizes, as well as varied calculation speeds and object detection efficiencies. This study studies and examines some measurements for assessing the object detection algorithm YOLOV5.

The you just look once (YOLO) calculation makes full advantage of the element guides by connecting the component maps in various scales to the image's bounding box. The COCO datasets were used to prepare the YOLO. The COCO dataset also includes people, tricycles, bicycles, automobiles, jet aircraft, helicopters, stop signs, fire hydrants, and animals including dogs, birds,

cows, horses, and sheep. It also includes kitchen and eating utensils such wine glasses, cups, forks, knives, spoons, and so forth. The darknet group created the photos.

Glen Jocher Developed Yolov5 and its open source available on github.

### 4.4.4:Implementation Steps:

The model is implemented in this work utilising the Python programming language and the data science, machine learning, and deep learning packages listed below:

1. numpy ( library for manipulating with arrays)
2. pandas (library for analyzing the data)
3. os (library for interacting with an operating system)
4. seaborn (library for data visualization)
5. matplotlib (library for data visualization)
6. pyplot (library for data visualization and plotting graphs)
7. shutil (for dividing the dataset into training and testing sets)
8. train_test_split (for data splitting)
9. tensorflow (for deep learning model building)
10. cv2 (for image processing and computer vision tasks)
11. keras (for deep learning model building)
12. tqdm (for adding progress bar)
13. imgaug (for image augmentation)
14. pil (for image processing and manipulation)
15. deep learning libraries for model building and evaluation.

### 4.4.5:Complete Workflow:

The whole process used to create this deep learning model is detailed in the workflow explanation that follows.

1. Importing the useful libraries
2. Getting the labels

3. Loading the images in vector format
4. Setting the features and target
5. Model building
6. Applying callbacks
7. Model compilation and fitting
8. Model evaluation
9. Graphs (Loss Vs Error for ResNet)
10. Graphs (Loss Vs Error for EfficientNetB0)
11. ResNet model prediction
12. EfficientNetB0 model prediction

The complete explanation of this task has been mentioned above, and in the below section we have described each component of this task step-by-step.

- **Importing necessary libraries:**

In this step, we have imported the mandatory libraries for detection and counting tasks. Below is the attached code screenshot.

```
[ ] import numpy as np
    import pandas as pd
    import seaborn as sns
    import matplotlib.pyplot as plt
    import shutil
    from sklearn.model_selection import train_test_split
    import tensorflow as tf
    from sklearn.model_selection import train_test_split
    import numpy as np
    import pandas as pd


    import os
    import tensorflow as tf
    import cv2
    from tensorflow.keras import layers
    from tensorflow.keras.layers import Input, Add, Dense, Dropout, Activation, ZeroPadding2D, BatchNormalization, Flatten, Conv2D, AveragePooling2D, MaxPooling2D, GlobalMaxPoolin
    from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler, EarlyStopping, ReduceLROnPlateau
    from tensorflow.keras.optimizers import Adam,SGD
    from tensorflow.keras.preprocessing.image import ImageDataGenerator
    from tensorflow.keras.models import Sequential, load_model, Model
    from tensorflow.keras.callbacks import LearningRateScheduler
    from tensorflow.keras.preprocessing import image
    from tensorflow.keras.utils import to_categorical
    from tensorflow.keras import metrics
    from tensorflow.keras.preprocessing import image
    from tensorflow.keras.applications.imagenet_utils import preprocess_input
    from tensorflow.keras.initializers import glorot_uniform
    from tqdm import tqdm
    import imgaug as ia
    from imgaug import augmenters as iaa
    from PIL import Image
    import keras.backend as K
    K.set_image_data_format('channels_last')
    K.set_learning_phase(1)
```

- **Getting the labels**

In this step, we have defined the labels given in our dataset. The label table shows the count against each personID. Below is the attached code screenshot.

```
# getting the labels corresponding to the image
label_df = pd.read_csv('/content/drive/My Drive/crowd-counting/labels.csv')
label_df.columns = ['id' , 'people']
label_df.head()
```

- **Loading the images in vector format:**

In this step, we have loaded the .npy formatted image files present in our dataset and displayed the shape of all images. Below is the attached code screenshot.

```
[ ]  # loading the images in vector format
    img = np.load('/content/drive/My Drive/crowd-counting/images.npy')
    #img = img.reshape(img.shape[0], img.shape[1], img.shape[2], img.shape[3],1)
    img.shape
```

- **Setting the features and target:**

In this step, we have splatted our dataset into training, and testing sets. We have kept the split ratio to 0.2. It means 80% data will be used for training purposes whereas 20% data will be used for testing purposes (for making predictions on testing data). We have shown shapes for training and testing. Below is the attached code screenshot.

```
[ ]  # setting features and target value

    x_train, x_test, y_train, y_test = train_test_split(img, labels, test_size=0.1)
    print(x_train.shape[0])
    print(x_test.shape[0])
```

- **Model building:**

In this step, we have defined the architecture of our model by first importing it from TensorFlow applications and afterwards downloading and adding some parameters to it. We have set the weights to default **(ImageNet)**, include top to **False** as our image size is different from default image size which is **(224x224x3)**. In our case, the image size is (480x640x3). We have set the image size as input shape and pooling as avg. Below is the attached code screenshot.

```
[ ]  resnet_model = ResNet50(
        weights='imagenet',
        include_top=False,
        input_shape=(480, 640, 3),
        pooling='avg',
    )
```

- **Fine tuning for ResNet 50 customized:**

In this step, we have fine tuned the model by freezing the model layers which perform poor and training it with the layers which perform extraordinary. Below is the attached screenshot.

In [9]:

```
x = resnet_model.output
x = Dense(1024, activation='relu')(x)
predictions = Dense(1, activation='linear')(x)
```

In [10]:

```
model = Model(inputs=resnet_model.input, outputs=predictions)
```

In [11]:

```
k = -7
for layer in model.layers[:k]:
    layer.trainable = False
print('Trainable:')
for layer in model.layers[k:]:
    print(layer.name)
    layer.trainable = True
```

- **Fine tuning for efficientnetb0:**

```
In [ ]:
x = efficientnetb0_model.output
x = Dense(1024, activation='relu')(x)
predictions = Dense(1, activation='softmax')(x)
```

```
In [ ]:
model = Model(inputs=efficientnetb0_model.input, outputs=predictions)
```

```
In [ ]:
k = -7
for layer in model.layers[:k]:
    layer.trainable = False
print('Trainable:')
for layer in model.layers[k:]:
    print(layer.name)
    layer.trainable = True
```

- **Applying Callbacks:**

In this step, we have applied the callbacks for our model. Callbacks are basically used for performing actions at various stages of the training. We have used 3 callbacks for our model (early stopping, reduce lr on plateau, learning rate scheduler). Early Stopping stops the training of a model if it starts to overtrain. It eventually stops the training of a model if the evaluation metric stopped improving. Reduce LRO n Plateau reduces the learning rate if the evaluation metric stopped improving. The Learning Rate Scheduler callback obtains the new learning rate value with the current epoch and current learning rate from the schedule function that we establish beforehand before training, and then applies the updated learning rate to the optimizer. Below is the attached code screenshot.

```
[ ] def scheduler(epoch, lr):
      if epoch < 10:
        return lr
      else:
        return lr * tf.math.exp(-0.1)
```

```
[ ] reduce_lr = tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5, min_lr=0.001)
    early_stopping = tf.keras.callbacks.EarlyStopping(monitor='loss', patience=3)
    learning_rate_scheduler = tf.keras.callbacks.LearningRateScheduler(scheduler)
```

- **Model Compilation And Fitting:**

In this step, we have compiled our model using ‘Adam’ as our optimizer, ‘Huber loss’ as our loss function and ‘mean absolute error’ as our evaluation metric. We have finally fitted our model on 20, 50 and 100 epochs. Below is the attached code screenshot.

```
[ ]  history = model.fit(x_train,
                y_train,
                validation_data=(x_test, y_test),
                epochs=20,
                batch_size=8,
              )
```

- **Model evaluation:**

In this step, we have evaluated our model on training and testing sets. We have shown the mean absolute error and loss on training and testing sets. Below is the attached code screenshot.

```
[ ]  # model error on training dataset
     score = model.evaluate(x_train,
                                  y_train,
                                  verbose = 0)
     print("\nTrain error: %.1f%%" % (100.0 * score[1]))
```

```
[ ]  # model error on test dataset
     score = model.evaluate(x_test,
                                 y_test,
                                 verbose = 0)
     print("\nTest error: %.1f%%" % (100.0 * score[1]))
```

```
[ ]  eval_score = model.evaluate(x_test, y_test)
     print("Test loss:", eval_score[0])
     print("Test error:", eval_score[1])
```

- **Model Prediction:**

Below are the results of ResNet customized on our crowd counting dataset.

```
Training set --
    ground truth:  55981
  evaluate count:  [55940]
Testing set --
    ground truth:  6334
   predict count:  [6294]
```

### 4.5:YOLOv5 Implementation

#### 4.5.1:Format for YOLO labels

The majority of annotation platforms enable export in the YOLO labelling format, which provides one text file for each image's annotations. Each object in the image has its own bounding-box (BBox) annotation in its respective text file. The annotations fall between 0 and 1, normalised to the size of the image. The following format is used to display them:

< object-class-ID> <X center> <Y center> <Box width> <Box height>

If there are multiple objects in the image, then yolo annotations look like.

```
0 0.217187 0.170833 0.034375 0.116667
0 0.379687 0.48125 0.05 0.125
0 0.347656 0.501042 0.0546875 0.102083
0 0.304688 0.229167 0.01875 0.075
0 0.920313 0.305208 0.034375 0.14375
0 0.245312 0.169792 0.028125 0.114583
0 0.507812 0.626042 0.05 0.227083
0 0.595312 0.595833 0.059375 0.204167
0 0.745313 0.147917 0.028125 0.1
0 0.64375 0.858333 0.1 0.258333
0 0.30625 0.428125 0.04375 0.18125
0 0.866406 0.307292 0.0328125 0.147917
0 0.883594 0.888542 0.0796875 0.222917
```

### 4.5.2:Image Annotation

Image annotation is the process of assigning labels to digital photographs. Typically, this involves human input, however occasionally computers may also aid. To provide the computer vision model with information about the items present in the image, labels are predetermined by a machine learning engineer and are selected.

### 4.5.3:LabelImg (Annotation Tool)( https://github.com/heartexlabs/labelImg)

LabelImg is a graphical image annotation tool. We installed labelImg using python anaconda prompt. We did manual annotation to improve the object detection and prediction.

In the above image we draw manual annotation and green boundary boxes show that one and we are looking for people we selected label as person.

### 4.5.4:Complete Workflow For YOLOV5:

1). Here we have set up the model, and we have downloaded the resources from GitHub and uploaded them into the local drive. After that, we have installed dependencies and checked PyTorch and GPU.

```
%cd /content/drive/MyDrive/crowd_count_dataset/yolov5
%pip install -qr requirements.txt  # install

import torch
import utils
display = utils.notebook_init()  # checks
```

```
YOLOv5 🚀 v6.2-14-gd40cd0d Python-3.7.13 torch-1.12.1+cu113 CUDA:0 (Tesla T4, 15110MiB)
Setup complete ✅ (2 CPUs, 12.7 GB RAM, 37.4/78.2 GB disk)
```

2). In this step,Detect.py runs yolov5 inference on a given source and results are saved into runs/detect/exp11

```
!python detect.py --weights yolov5l.pt --img 640 --conf 0.25 --source /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/

detect: weights=['yolov5l.pt'], source=/content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/, data=data/coco128.yaml, imgsz
YOLOv5 🚀 v6.2-14-gd40cd0d Python-3.7.13 torch-1.12.1+cu113 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
YOLOv5l summary: 367 layers, 46533693 parameters, 0 gradients
image 1/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000001.jpg: 480x640 21 persons, 44.1ms
image 2/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000002.jpg: 480x640 25 persons, 43.8ms
image 3/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000003.jpg: 480x640 25 persons, 43.8ms
image 4/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000004.jpg: 480x640 24 persons, 44.0ms
image 5/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000005.jpg: 480x640 25 persons, 43.8ms
image 6/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000006.jpg: 480x640 26 persons, 43.8ms
image 7/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000007.jpg: 480x640 18 persons, 43.8ms
image 8/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000008.jpg: 480x640 27 persons, 43.8ms
image 9/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000009.jpg: 480x640 16 persons, 43.8ms
image 10/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000010.jpg: 480x640 15 persons, 43.8ms
image 11/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000011.jpg: 480x640 13 persons, 43.8ms
image 12/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000012.jpg: 480x640 15 persons, 43.8ms
image 13/2000 /content/drive/MyDrive/crowd_count_dataset/yolo_dataset/images/seq_000013.jpg: 480x640 16 persons, 43.8ms
```

```
Results saved to runs/detect/exp11
2000 labels saved to runs/detect/exp11/labels
```

3). Next to, we load labels using pandas library from our dataset.

```
import pandas as pd
label_df = pd.read_csv('/content/drive/MyDrive/crowd_count_dataset/labels.csv')
label_df.columns = ['id' , 'people']
label_df.head()
label_df.tail()
```

| | id | people |
|---|---|---|
| **1995** | 1996 | 27 |
| **1996** | 1997 | 27 |
| **1997** | 1998 | 25 |
| **1998** | 1999 | 26 |
| **1999** | 2000 | 26 |

4)In this step, we have used dictionary too store result of the object detected and inside that used file function of python. As well, we implemented orderedict because the output is unordered and random so we using orderdict and sorted function so we can get proper output.

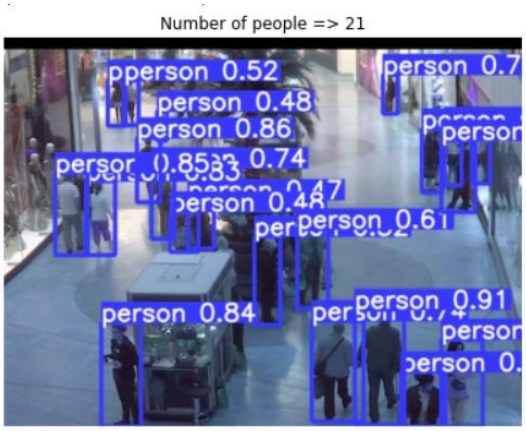
```
[ ]  predicted_lines = {}
     for i in result_label:
       with open(result_dir+i, 'r') as fp:
         lines = len(fp.readlines())
         predicted_lines[int(i.split('_')[1][:-4])] = lines
         # predicted_lines.append(lines)
         # print('Total Number of lines:', lines)
```

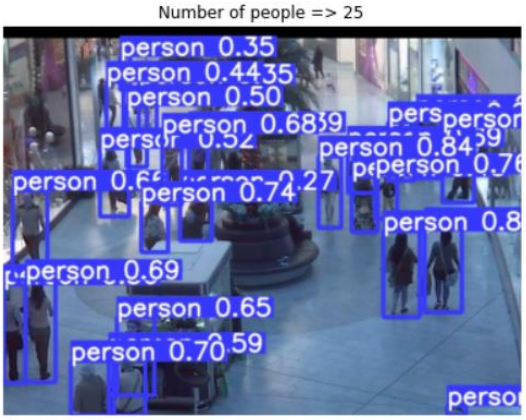
```
     import pandas as pd
     from collections import OrderedDict
     dict1 = OrderedDict(sorted(predicted_lines.items()))
     df = pd.DataFrame(list(dict1.items()), columns = ['id', 'count'])
```

5). After that we can compare the actual vs predicted count .

| people | predicted_count |
|---|---|
| 35 | 21 |
| 41 | 25 |
| 41 | 25 |
| 44 | 24 |
| 41 | 25 |
| ... | ... |
| 27 | 11 |
| 27 | 16 |
| 25 | 13 |

6) In below image we have plotted the object detected image using matplotlib library with labels of number of people.




7) In below image we have trained our dataset and we have given the path in data.yaml using epoch of 3 and batch size of 16

```
!python train.py --img 640 --batch 16 --epochs 3 --data data.yaml --weights yolov5s.pt
```

```
train: weights=yolov5s.pt, cfg=, data=data.yaml, hyp=data/hyps/hyp.scratch-low.yaml, epochs=3, batch
```

8) Next to it, we test our dataset with latest trained best.pt and detect the updated images with bounding boxes.

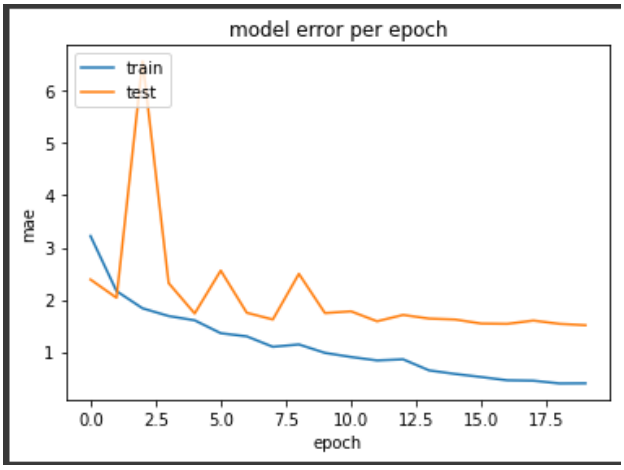
```
[ ] !python detect.py --weights runs/train/exp6/weights/best.pt --img 640 --conf 0.25 --source /content/drive/MyDrive/crowd_count_dataset/yolo_dataset
```

9) Below image shows the updated boxes with the counting persons detected by model.

```
Text(0.5, 1.0, 'Fourth')
```

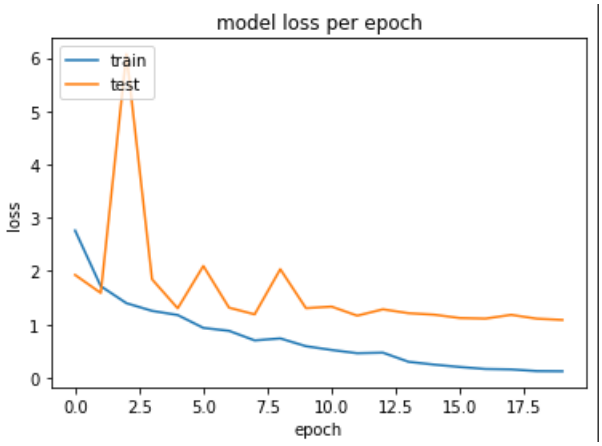
# Chapter-5:Discussion

- **Loss Vs Error for ResNet:**




**It can be inferred from the graph:**

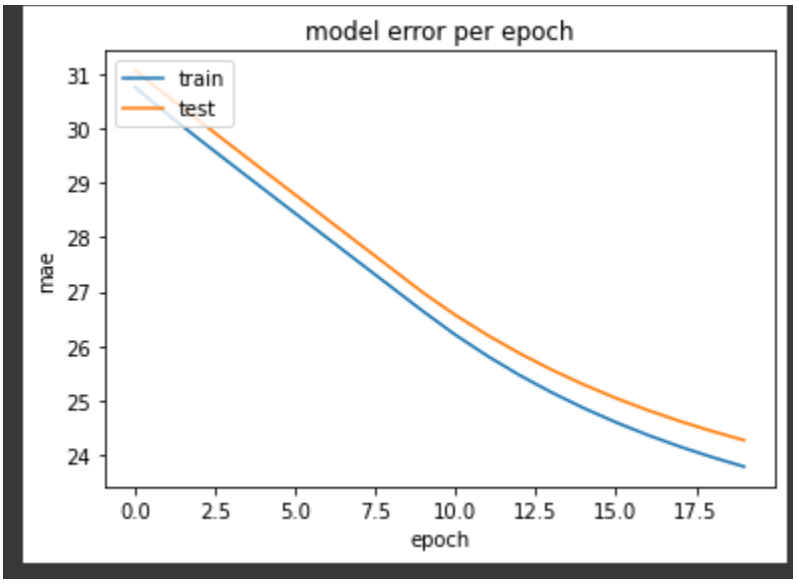
- The blue line shows the training error whereas the orange line shows the testing error.
- The training error starts from 3.22 and eventually reached end with ups and downs at 0.39.
- The testing error starts from 2.38 and eventually reached end with ups and downs following training error at 1.51.
- The x-axis shows the total no of epoch whereas the y-axis shows the mean absolute error.
- There is a scale of 2.5 on the x-axis and 1 on the y-axis.



**It can be inferred from the graph:**

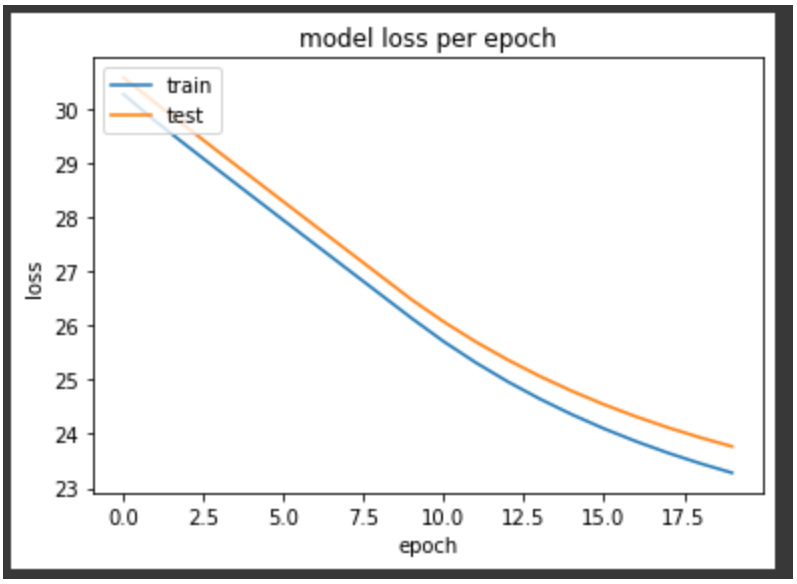
- The blue line shows the training loss whereas the orange line shows the testing loss.
- The training loss starts from 2.76 and eventually reaching end with ups and downs at 0.12.
- The testing loss starts from 1.92 and eventually reaching end with ups and downs following training loss at 1.08.
- There is a scale of 2.5 on x-axis and 1 on the y-axis.

- **Loss Vs Error Graphs (EfficientNetB0):**




**It can be inferred from the graph:**

- The blue line shows the training error whereas the orange line shows the testing error.
- The training error starts from 30.76 and eventually reaching end at 23.78.
- The testing error starts from 31.06 and eventually reaching end at 24.26.
- The x-axis shows the no. of epochs whereas the y-axis shows the mean absolute error.
- There is a scale of 2.5 at x-axis and 1 at y-axis.




**It can be inferred from the graph:**

- The blue line shows the training loss whereas the orange line shows the testing loss.
- The training loss starts from 30.26 and eventually reached end at 23.28.
- The testing loss starts from 30.56 and eventually reached end at 23.76.
- The x-axis shows the no. of epochs whereas the y-axis shows the model loss.
- There is a scale of 2.5 on x-axis and 1 on the y-axis.

**Comparison Of Mean Absolute Error By No. Of Epochs (ResNet):**

| **No.of epochs** | **Model** | **MAE** | **VMAE** |
|---|---|---|---|
| 20 | ResNet | 0.3981 | 1.5111 |
| 50 | ResNet | 0.2664 | 1.6556 |
| 100 | ResNet | 0.3576 | 1.2172 |

**Comparison of Mean Absolute Error by No. Of Epochs (EfficientNetB0):**

| No. of epochs | Model | MAE | VMAE |
|---|---|---|---|
| 20 | EfficientNetB0 | 23.7807 | 24.2670 |
| 50 | EfficientNetB0 | 0.9148 | 1.4084 |
| 100 | EfficientNetB0 | 22.2360 | 23.2999 |

**Comparison of Actual vs Predicted crowd count of (pretrained yolov5)**

| people | predicted_count |
|---|---|
| 35 | 21 |
| 41 | 25 |
| 41 | 25 |
| 44 | 24 |
| 41 | 25 |

In the above image we have actual people in image with 35 but our pretrained model predicted and detected people is 21 and same thing for next upcoming images.

**Comparison of Actual vs Predicted crowd count of (yolov5 with epochs and batch size)**

| people | predicted_count |
|---|---|
| 35 | 24 |
| 41 | 28 |
| 41 | 25 |
| 44 | 25 |
| 41 | 26 |

In this upper side image illustrates that, the updated result with epochs and batch size is better than the normal one and predicted count is increased by 3 to 4 person and this happened to all images. So with this our result is good as compare to previous result.

# Chapter-6: Summary Conclusion

### 6.1 : Conclusion

In the area of computer vision & deep learning, crowd counting is a crucial and difficult application in which items, notably as humans, are counted in a given picture or video with the use of object detection algorithms like EfficientDet and Yolov5. This application is crucial in a number of areas, including traffic management and catastrophe management. In the realm of object identification, deep learning and artificial intelligence have recently displayed exceptional benchmark performance. The objective of this project is to create an intelligent object recognition model that counts individuals in a video or picture using CNN deep learning techniques. Keeping in view the nature of dataset CNN algorithm will be used, & accurate results by using the proposed CNN model is projected.[46]

### 6.2: Limitation

Accurate counts are difficult when the input is a dense image. This is because most targets are heavily hidden in a very crowded scene. Less data is one the major limitation because applying a model that was trained on one incidence may not compulsory apply in the second incidence. Deep learning systems are prone to spoofing because of their substantial reliance on accurate and plentiful data.

### 6.3: Future work

The yolov5 algorithm the persons whose full body is seen in image, in crowded scenes its not perfect giving the results which could be improved in the future. In future the training can be implemented on the face of person for better handling in crowded inputs. In the Efficientnetb0 and Resnet improvement with object detection part could be extended. Sampling the poorly performing photos and retraining the CNN model on the dataset using the sampled images may improve the results. Applying updated yolo might improve the results.

# Chapter-7:References


1. Tan, M., Pang, R. and Le, Q.V., 2020. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10781-10790)
2. Zhu, X., Lyu, S., Wang, X. and Zhao, Q., 2021. TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2778-2788).
3. Gao, J., Wang, Q. and Yuan, Y., 2019. SCAR: Spatial-/channel-wise attention regression networks for crowd counting. *Neurocomputing*, *363*, pp.1-8.
4. Bhangale, U., Patil, S., Vishwanath, V., Thakker, P., Bansode, A. and Navandhar, D., 2020. Near Real-time Crowd Counting using Deep Learning Approach. *Procedia Computer Science*, *171*, pp.770-779
5. Miao, Y., Han, J., Gao, Y. and Zhang, B., 2019. ST-CNN: Spatial-Temporal Convolutional Neural Network for crowd counting in videos. *Pattern Recognition Letters*, *125*, pp.113-118.
6. Hossain, M., Hosseinzadeh, M., Chanda, O. and Wang, Y., 2019, January. Crowd counting using scale-aware attention networks. In *2019 IEEE winter conference on applications of computer vision (WACV)* (pp. 1280-1288). IEEE.
7. Zeng, X., Wu, Y., Hu, S., Wang, R. and Ye, Y., 2020. DSPNet: Deep scale purifier network for dense crowd counting. *Expert Systems with Applications*, *141*, p.112977.
8. Wang, S., Lu, Y., Zhou, T., Di, H., Lu, L. and Zhang, L., 2020. SCLNet: Spatial context learning network for congested crowd counting. *Neurocomputing*, *404*, pp.227-239.
9. Xu, C., Liang, D., Xu, Y., Bai, S., Zhan, W., Bai, X. and Tomizuka, M., 2022. Autoscale: Learning to scale for crowd counting. *International Journal of Computer Vision*, *130*(2), pp.405-434.
10. Sreenu, G. and Durai, S., 2019. Intelligent video surveillance: a review through deep learning techniques for crowd analysis. *Journal of Big Data*, *6*(1), pp.1-27.
11. Bhuiyan, M.R., Abdullah, J., Hashim, N. and Al Farid, F., 2022. Video analytics using deep learning for crowd analysis: a review. *Multimedia Tools and Applications*, pp.1-28.

12. Elbishlawi, S., Abdelpakey, M.H., Eltantawy, A., Shehata, M.S. and Mohamed, M.M., 2020. Deep learning-based crowd scene analysis survey. *Journal of Imaging*, *6*(9), p.95.
13. Liu, S., Zhao, Y., Xue, F., Chen, B. and Chen, X., 2019. DeepCount: Crowd counting with WiFi via deep learning. *arXiv preprint arXiv:1903.05316*.
14. Liu, L., Wang, H., Li, G., Ouyang, W. and Lin, L., 2018. Crowd counting using deep recurrent spatial-aware network. *arXiv preprint arXiv:1807.00601*.
15. Wang, Q., Lin, W., Gao, J. and Li, X., 2020. Density-aware curriculum learning for crowd counting. *IEEE Transactions on Cybernetics*.
16. Hossain, M., Hosseinzadeh, M., Chanda, O. and Wang, Y., 2019, January. Crowd counting using scale-aware attention networks. In *2019 IEEE winter conference on applications of computer vision (WACV)* (pp. 1280-1288). IEEE.
17. Cheng, Z.Q., Li, J.X., Dai, Q., Wu, X. and Hauptmann, A.G., 2019. Learning spatial awareness to improve crowd counting. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6152-6161).
18. Jiang, X., Zhang, L., Xu, M., Zhang, T., Lv, P., Zhou, B., Yang, X. and Pang, Y., 2020. Attention scaling for crowd counting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 4706-4715).
19. Meng, Y., Zhang, H., Zhao, Y., Yang, X., Qian, X., Huang, X. and Zheng, Y., 2021. Spatial uncertainty-aware semi-supervised crowd counting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 15549-15559).
20. Miao, Y., Lin, Z., Ding, G. and Han, J., 2020, April. Shallow feature based dense attention network for crowd counting. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 34, No. 07, pp. 11765-11772).
21. Yan, Z., Yuan, Y., Zuo, W., Tan, X., Wang, Y., Wen, S. and Ding, E., 2019. Perspective-guided convolution networks for crowd counting. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 952-961).
22. Sindagi, V.A., Yasarla, R. and Patel, V.M., 2019. Pushing the frontiers of unconstrained crowd counting: new dataset and benchmark method. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 1221-1231).

23. Zhou, Y., Yang, J., Li, H., Cao, T. and Kung, S.Y., 2020. Adversarial learning for multiscale crowd counting under complex scenes. *IEEE transactions on cybernetics*, *51*(11), pp.5423-5432.
24. Brunetti, A., Buongiorno, D., Trotta, G.F. and Bevilacqua, V., 2018. Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. *Neurocomputing*, *300*, pp.17-33.
25. Tan, M., Pang, R. and Le, Q.V., 2020. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10781-10790).
26. Tan, M., Pang, R. and Le, Q.V., 2020. Efficientdet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10781-10790).
27. Afif, M., Ayachi, R., Said, Y. and Atri, M., 2022. An evaluation of EfficientDet for object detection used for indoor robots assistance navigation. Journal of Real-Time Image Processing, 19(3), pp.651-661.
28. Chen, X., Yan, H., Li, T., Xu, J. and Zhu, F., 2021. Adversarial scale-adaptive neural network for crowd counting. Neurocomputing, 450, pp.14-24.
29. Ptak, B., Pieczyński, D., Piechocki, M. and Kraft, M., 2022. On-Board Crowd Counting and Density Estimation Using Low Altitude Unmanned Aerial Vehicles—Looking beyond Beating the Benchmark. Remote Sensing, 14(10), p.2288.
30. Mandal, V. and Adu-Gyamfi, Y., 2020. Object detection and tracking algorithms for vehicle counting: a comparative analysis. Journal of big data analytics in transportation, 2(3), pp.251-261.
31. Cao, M.Y. and Zhao, J., 2022. Fast EfficientDet: An Efficient Pedestrian Detection Network. Engineering Letters, 30(2).
32. Ranjan, V., Le, H. and Hoai, M., 2018. Iterative crowd counting. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 270-285).
33. Zhang, A., Yue, L., Shen, J., Zhu, F., Zhen, X., Cao, X. and Shao, L., 2019. Attentional neural fields for crowd counting. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 5714-5723)

34. Yan, Z., Yuan, Y., Zuo, W., Tan, X., Wang, Y., Wen, S. and Ding, E., 2019. Perspective-guided convolution networks for crowd counting. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 952-961).
35. Sindagi, V.A. and Patel, V.M., 2019. Multi-level bottom-top and top-bottom feature fusion for crowd counting. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 1002-1012).
36. Zhang, A., Shen, J., Xiao, Z., Zhu, F., Zhen, X., Cao, X. and Shao, L., 2019. Relational attention network for crowd counting. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 6788-6797).
37. Wang, B., Liu, H., Samaras, D. and Nguyen, M.H., 2020. Distribution matching for crowd counting. Advances in neural information processing systems, 33, pp.1595-1607.
38. Jiang, X., Xiao, Z., Zhang, B., Zhen, X., Cao, X., Doermann, D. and Shao, L., 2019. Crowd counting and density estimation by trellis encoder-decoder networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 6133-6142).
39. Chen, X., Bin, Y., Sang, N. and Gao, C., 2019, January. Scale pyramid network for crowd counting. In 2019 IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 1941-1950). IEEE.
40. Zhou, J.T., Zhang, L., Du, J., Peng, X., Fang, Z., Xiao, Z. and Zhu, H., 2021. Locality-aware crowd counting. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(7), pp.3602-3613.
41. Oh, M.H., Olsen, P. and Ramamurthy, K.N., 2020, April. Crowd counting with decomposed uncertainty. In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 07, pp. 11799-11806).
42. Jiang, X., Zhang, L., Zhang, T., Lv, P., Zhou, B., Pang, Y., Xu, M. and Xu, C., 2020. Density-aware multi-task learning for crowd counting. IEEE Transactions on Multimedia, 23, pp.443-453.
43. Liu, C., Weng, X. and Mu, Y., 2019. Recurrent attentive zooming for joint crowd counting and precise localization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1217-1226).

44. Alahi, M.E.E., Akhter, F., Nag, A., Afsarimanesh, N. and Mukhopadhyay, S., 2022. Internet of Things (IoT)-enabled pedestrian counting in a smart city. In *Proceedings of International Conference on Computational Intelligence and Computing* (pp. 89-104). Springer, Singapore.

45. Zhang, Y., Zhou, D., Chen, S., Gao, S. and Ma, Y., 2016. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 589-597).

46. https://learn.solent.ac.uk/mod/assign/view.php?id=2351371 self-project submitted to solent university- project proposal.

47. Elgendy, M., 2020. *Deep learning for vision systems*. Simon and Schuster.

48. Karamouzas, I., Sohre, N., Hu, R. and Guy, S.J., 2018. Crowd space: a predictive crowd analysis technique. *ACM Transactions on Graphics (TOG)*, *37*(6), pp.1-14.

49. Li, B., Huang, H., Zhang, A., Liu, P. and Liu, C., 2021. Approaches on crowd counting and density estimation: A review. *Pattern Analysis and Applications*, *24*(3), pp.853-874.

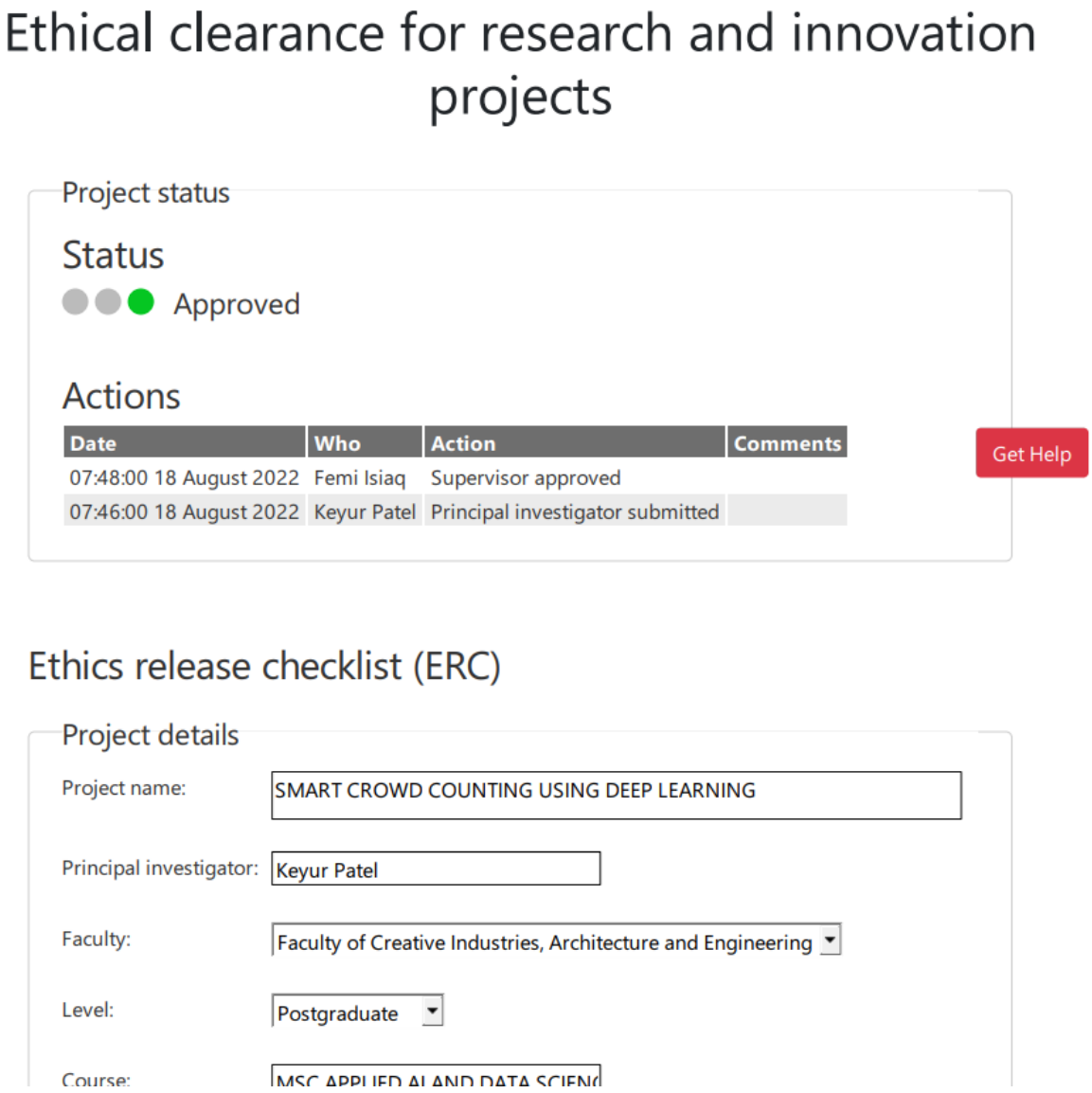
50. Shi, Z., Zhang, L., Liu, Y., Cao, X., Ye, Y., Cheng, M.M. and Zheng, G., 2018. Crowd counting with deep negative correlation learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5382-5390).

51. Tan, M., Pang, R. and Le, Q.V., 2020. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10781-10790).

52. Wan, J. and Chan, A., 2019. Adaptive density map generation for crowd counting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 1130-1139).

# Chapter-8: Appendices

**Appendix A.**
Ethical approval from supervisor.



Ethical clearance for research and innovation projects

Project status

Status

Approved

Actions

| Date | Who | Action | Comments |
|---|---|---|---|
| 07:48:00 18 August 2022 | Femi Isiaq | Supervisor approved | |
| 07:46:00 18 August 2022 | Keyur Patel | Principal investigator submitted | |

Get Help

Ethics release checklist (ERC)

Project details

Project name: SMART CROWD COUNTING USING DEEP LEARNING

Principal investigator: Keyur Patel

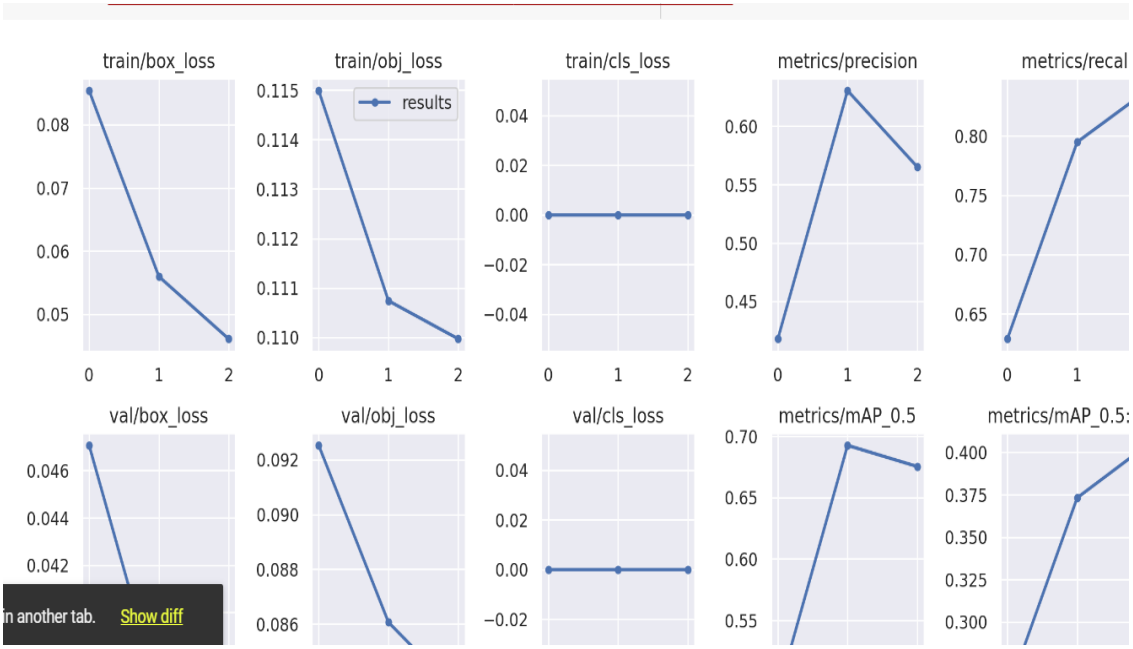
Faculty: Faculty of Creative Industries, Architecture and Engineering

Level: Postgraduate

Course: MSC APPLIED AI AND DATA SCIEN(

## Appendix B.

Results of yolov5




## Appendix C.

The ground truth of yolov5 training data