S	
E1	
E2	
For Office Use Only	

Masters Project Final Report

(MCS)

2022

Project Title	**An Approach for Crowd Counting and Crowd Density Estimation using Aerial Images**
Student Name	**M R L Perera**

Registration No. & Index No.	2019/MCS/066 19440669
Supervisor’s Name	S P W Wimalarathne

For Office Use ONLY

An Approach for Crowd Counting and Crowd Density Estimation using Aerial Images

A dissertation submitted for the Degree of Master of Computer Science

M R L Perera

University of Colombo School of Computing

2022

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: M R L Perera

Registration Number: 2019MCS066

Index Number: 19440669

Rlperera

Signature: Date: 20/11/2022

This is to certify that this thesis is based on the work of Mr./Ms Prasad Wimalarathne
Under my supervision, The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by: Prasad Wimalarathne

Supervisor Name: Prasad Wimalarathne

Signature: Date: 20/11/2022

Acknowledgement

I would like to express my sincere thanks to my supervisor Mr Prasad Wimalarathne who guided me to success throughout the project, without whom I couldn’t have accomplished my project.Special thanks to my parents and my friends who directly or indirectly helped me complete my project.

Abstract

When a large number of people gather in a single area, it could lead to a mass stampede which could result in major injuries and deaths. It is important to measure the number of people in a gathering therefore pre-security measures can be taken to avoid such stampedes from rising. High beam cameras or drones can be used to capture images in a large crowded area without any further human involvement. In the present, most of the crowd counting techniques use face and head detection on finding humans in a provided image. One major disadvantage of these approaches is occlusion occurring if objects are near in the images.

The goal of this thesis is to perform a bounding box regression algorithm on top of a convolutional neural network based model to detect and count the crowd for a provided input image.

The convolutional neural network is trained for head detection and the bounding box algorithm is used to draw bounding boxes on the areas which are detected by the network as heads. Box count is taken as the crowd count. The count is validated against the ground truth count provided by the testing dataset in Classification Accuracy and Mean Approximation Error. The model is implemented using Keras framework, an open source machine library in python.

The solution is trained using the dataset “Crowd Detection Model” and has shown a mean approximation error of 0.35 and a classification accuracy of 0.65.

Keywords - Convolutional Neural Network, Bounding Box, Non Max Suppression, Classification Accuracy, Mean Approximation Error.

Table of Contents

List of Figures

List of Tables

Introduction

At present, crowd gatherings occur frequently due to many events such as Sports, Strikes, Events organized by organizations such as walks and political meetings and religious events. Often, these gatherings cause tragic incidents such as stampedes which may result in severe injuries for the people and even loss of precious human lives.

Due to rapid increase of the population, in recent years, there were reports filed on losing lives due to stampedes occuring in a sudden gathering of a large amount of people in a limited urban area. During the recent past, there were multiple stampedes happening throughout the world. There were reports of multiple stampedes in Mecca, Saudi Arabia due to a large number of worshippers gathering suddenly for Hajj pilgrimage. Mina Stampede occurred in Mecca 2015 (“AP count,” 2015) considered to be a major stampede occurred due a crane collapse which resulted in over 2400 deaths.

Vaishno Devi Stampede (“Vaishno Devi Stampede,” n.d.) occurred in Jammu and Kashmir, India due to a massive crowd gathering in a hindu temple on celebrating new year which resulted in 12 deaths.Astroworld Stampede (“Astroworld Festival crowd crush,” n.d.) occurred in the United States during a musical festival which resulted in 10 deaths.Meron Stampede (New York Times, n.d.) occurred in Israel at the annual Meron pilgrimage which resulted in 45 deaths. Dar Es Salaam Stampede (“Dar es Salaam Stampede,” n.d.) occurred in Tanzania in a local stadium which resulted in 45 deaths. Kanjuruhan stadium disaster (Ratcliffe et al., 2022) occurred in Java, Indonesia in Kanjuruhan stadium where 131 people died and 300 got injured during a fight between supporters. Seoul halloween crowd crush (Kim and Kuhn, 2022) occurred in Seoul, South Korea on 29th October 2022 where more than 156 people died and more than 152 people were injured during a sudden crowd gathering in a halloween festival. Maligawatta Stampede (“3 dead and 9 injured in a stampede in Maligawatta; 6 arrested,” n.d.) occurred in Maligawatta, Sri Lanka due to an unforeseen gathering of crowd during a religious event which resulted in 3 deaths.

1.1 Problem Statement

Early crowd counting methods predominantly consist of detection (Mousas, 2017) and regression based (Dalal and Triggs, 2005) methods while the images used were supposed to be normal camera images. In present low cost drones with high quality cameras as well as low cost high beam cameras are popular throughout the community. These drones provide the capacity to capture high quality dense images. Most of the current detection based methods use face recognition as the primary technique to get the count. Since the drone images do not contain a clear face of a human these techniques are shown to be having limitations on drone related images. Drone images may contain images of crowds with other items. Regression based methods are shown to be having limitations on these types of images. In order to overcome these identified limitations, cluster based methods (Brostow, n.d.) were introduced. Cluster based methods are revealed to be having limitations on providing accurate count for drone related images since those methods ignore camouflaged images. Institution of Convolutional Neural Network related solutions occurred to provide solutions to discrete determined limitations.

1.2 Aim and Objectives

1.2.1 Aim

Crowd counting is the technique to estimate the number of people of a given image by extracting the features of the image. Crowd counting is used in many applications such as resource management, traffic control, security and disaster management. Currently, several research has been conducted in this area on using crowd counting methods to get the count and the density of the people using images of many sources such as street camera images and videos therefore, necessary security measurement could be made to avoid or reduce the number of damages if a threat is found in such a situation.

Currently, Drones or Unmanned Aerial Vehicles with high quality cameras are widely used to capture images in areas where it is hard to be performed by a human. Main aim is to develop a machine learning algorithm to count the crowd of a given image taken by a drone in a higher density crowd, therefore security measures can be taken to prevent incidents like stampedes from happening. Following objectives are identified to satisfy the identified aim.

1.2.2 Objectives

In order to achieve the desired aim, following objectives have been identified.

- Conduct a systematic literature Review on the current machine learning based crowd counting approaches.
- Identify datasets related to crowd counting.
- Identify the most suitable approach to implement the proposed solution by analyzing the data sets.
- Investigate and select the current designs to figure the most suitable technology to develop the proposed solution.
- Develop the proposing solution using the selected design.
- Verify the developed solution using the training dataset.
- Use cross validations to generalize the solution using the identified testing dataset.

1.3 Scope

This thesis aims to build a crowd counting model which has the ability to estimate the count of the crowd of a provided image. In order to achieve this, following activities are identified as itemized research activities of the pre identified objectives.

Since the proposed solution is based on using aerial related images, the first primary scope activity is to discover image datasets which are similar to aerial images with a ground truth count value. Investigate the current available technologies to figure out suitable technologies to implement the proposed solution. Observe a proper pre defined convolutional neural network implementation to use as the model in the proposed solution. Observe computer vision based libraries on extracting coordinates of the objects of the provided input images. Implementation of the bounding box algorithm to estimate the count of the crowd in the input image.Evaluation of the estimated count with the ground truth provided in the dataset. Grouping the crowd while counting and getting the count of each group is considered out of scope.

1.4 Thesis Structure

Chapter 2 of this thesis report provides a detailed literature review of the current crowd counting techniques and recently provided crowd counting solutions with identified limitations for the introduced problem.

Chapter 3 of this thesis describes the methodology of the proposed solution in detail with charts and the selected data sets.

Chapter 4 of this thesis describes the evolution of the proposed solution.

Chapter 5 of this thesis describes the conclusion and future work.

2 Literature Review

2.1 Introduction

Object detection is the process of detecting objects using selected features of a provided image. Object detection is the main concept used in crowd counting techniques for detection of human objects in a provided image. This chapter contains summarized background knowledge on the solutions based on the current crowd counting techniques. This section contains a comparison of the current crowd counting techniques based on their performance. Since this thesis contains estimating count using head detection, it contains more details on detection bases techniques.

2.2 Literature Review on Current Techniques

Traditionally, crowd counting is conducted by assigning a set of people to manually count the number of people of a given image. Limitations include consuming a large amount of time when the provided number of images are high etc have been identified. With the rise of computer science this process can be conducted using machine learning. At present, there are specific techniques identified in machine learning on crowd counting. Figure 1 contains the summarized taxonomy of the current available techniques on crowd counting.

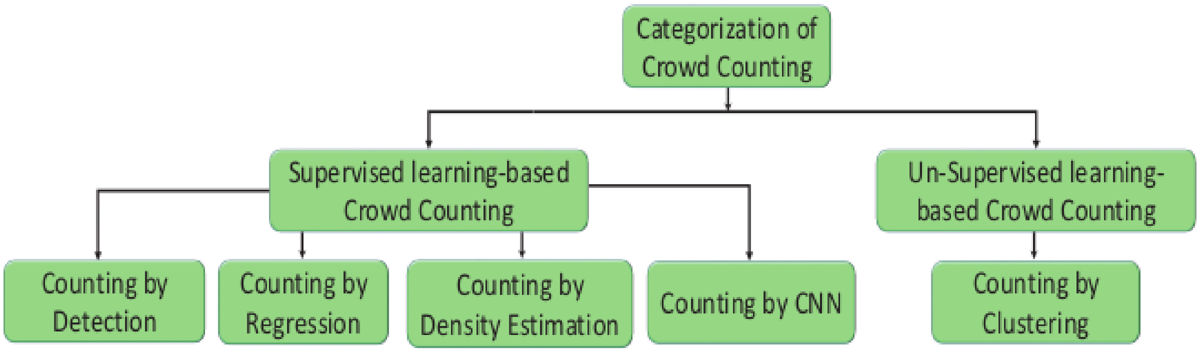

Categorization of Crowd Counting
Supervised learning-based Crowd Counting
Un-Supervised learning-based Crowd Counting
Counting by Detection
Counting by Regression
Counting by Density Estimation
Counting by CNN
Counting by Clustering

Figure 1. Taxonomy of Crowd counting techniques.

Detection Based - Crowd counting by detection can be defined as a method to compute the abstraction of image information and local decisions at every point to know about features of a particular type at that point (Mousas, 2017). In Detection Based crowd counting, the target crowd is counted by detecting each person in the image. To detect humans, head-shoulder detection is used. To perform head-shoulder detection a head detector in a sliding window is used on the image. The count of targets in an image are automatically given as a byproduct of detection results. However to improve head-shoulder detection accuracy people should be separated in the given images. Therefore this method has limitations on counting images with a higher crowd densities.

Regression Based - The Regression Based crowd counting was introduced to overcome the limitations of the Detection based approach (Lian et al., 2019). This approach learns a mapping of the images to a scalar value of count using their global or local features. The global features include the texture, edge, and gradient features, and the local features include Scale-Invariant Feature Transform (SIFT), Local Binary Patterns (LBPs), Histograms of Oriented Gradients (HOGs) and Gray-Level Co-occurrence Matrices (GLCMs). These algorithms solve occlusion and background clutter issues with detection-based approaches.

Density Estimation Based - Density Estimation Based crowd counting preserves both the count and spatial distribution of the crowd (Tang et al., 2021) . An object density map is created for each image. In an object density map, the integral over any subregion is the number of objects within the corresponding region in the image. Density-based methods are generally better at handling cases where objects are severely occluded by bypassing the hard detection of every object, while maintaining some spatial information about the crowd. Density estimation based approach uses the standard features to extract low-level information.

CNN Based - Though detection, regression, clustering, and density-estimation-based crowd-counting techniques perform well to some extent by using handcrafted features, for crowd analysis, motion analysis, different types of CNN based approaches are proposed. Convolutional Neural Network based crowd counting uses a neural network which accepts the whole image as the parameter and creates an end-to-end regression method to return the count of the input image. There are three identified CNN based crowd counting methods, namely, network based (Alotaibi et al., 2019), image view based and training approach based.

Cluster Based - In Cluster based crowd counting, clusters are created using features of the humans of the images (Brostow, n.d.). When creating clusters, it is assumed that visual features and individual motion fields are uniform.

ML Approach	**Related Work**	**Accuracy Archived**	**Strengths**	**Limitations**
Detection Based	(Mousas, 2017) (Shao et al., 2018)	(Shao et al., 2018) Have shown a recall value of 0.9	Works well for detecting faces. Suitable for sparse crowded images.	Does not provide Accurate counts for crowded images as the target people are not clear. Not suitable for dense crowd images
Regression Based	("Yao et al. - Deep Spatial Regression Model for Image Crowd Coun.pdf," n.d.) (Lian et al., 2019)	(Lian et al., 2019) have shown a mean approximation error of 0.08 ("Yao et al. - Deep Spatial Regression Model for Image Crowd Coun.pdf," n.d.) have shown a mean approximation error of 0.09	Accuracy is high when working with different data in the same scenario. Suitable for both dense and sparse crowd images.	Inaccurate when working with different data in different scenarios.
Density	(Tang et al.,	(Tang et al., 2021)	Accuracy is high	Accuracy will be

Estimation Based	2021)	have shown a mean approximation error of 0.07	as the count is taken using both high and low resolution density maps. Suitable for both dense and sparse crowd images.	lower if the resolution of the images decreases.
CNN Based	(Gong and Bourennane, 2018) (Alotibi et al., 2019b) (Li et al., 2018) (“Jayaram - I certify that this thesis does not incorporate wi.pdf,” n.d.)	(Alotibi et al., 2019b) have shown a mean approximation error of 0.06	Can be used in general crowd counting scenarios with a decent accuracy. However accuracy can be further improved by tuning the layers of the implemented method. Suitable for both dense and sparse crowd images.	Occlusion may occur if objects are near in the images.
Cluster Based	(Brostow, n.d.) (Khan et al.,	(Khan et al., 2021) have shown a mean	Can be used in motion based data.	Ignorance of objects if they are camouflaged.

	2021)	approximation error of 0.08		Ignorance of parts of objects which deform or exhibit sustained articulations.

Table 1. Comparison of existing crowd counting techniques.

Table 1 contains a comparison of the related work performed using existing crowd counting techniques with respect to their accuracy, strengths and limitations.

(Lian et al., 2019) points out a regression guided detection method with a density map to count the crowd using head and non head classification for RGB-D images. One identified limitation of this approach with using drone images is that the heads of the crowd are visible, therefore feature extraction will be less accurate.

(Tang et al., 2021) points out a model using a density estimation to count the crowds. This approach suggests a parallel dilated convolutional model to scale up the input images and a multiresolution density map to get the count.Some identified limitations of this approach is that it need a bit large number of variable in the process which need a large amount of memory as well as resolution of the images should be higher enough passing deepen of the density maps to get a higher accuracy of count of which drone images may have limitations on performing. Overview of this solution is provided in Figure 2.

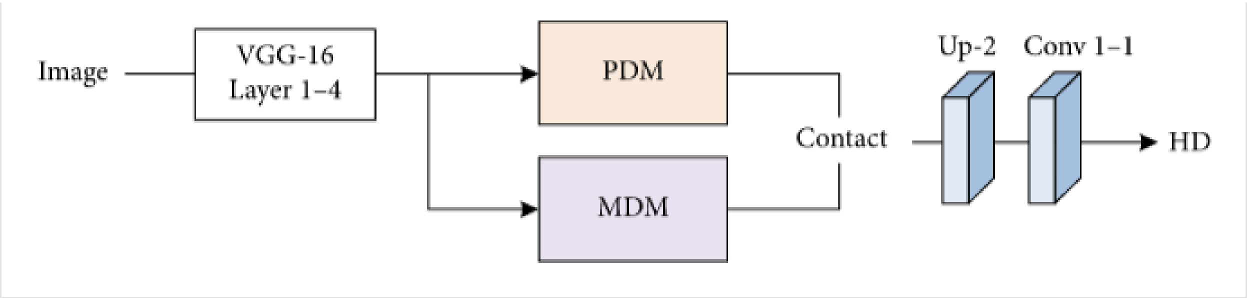


Figure 2. Overview of Tang's Solution

Shao (Shao et al., 2018) suggests an image annotation approach on detecting crowds in a provided image. In this approach, face detection is used and bounding boxes are drawn per each detected head. There are identified limitations as face detection is not guaranteed in aviation images where the head is considered as the ideal feature to count and the use case of this proposal is to get a higher accurate count of a dense crowd.

Alotaibi (Alotaibi et al., 2019) suggests a network based CNN deep learning method to count people of a given place. There are certain limitations identified including occlusion may occur if objects are near in the images.

Brostow (Brostow, n.d.) suggests a Baysian cluster algorithm to count the crowd. One major advantage of this approach is that the motion of the objects is considered. There are certain limitations identified including the ignorance of objects if they are camouflaged in the image and ignorance of parts of objects which deform or exhibit sustained articulations. Overview of this solution is provided in Figure 3.

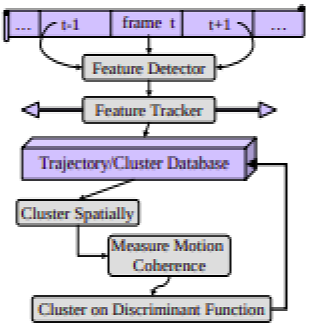


Figure 3. Overview of Brostow's Solution

Jayaram (“Jayaram - I certify that this thesis does not incorporate wi.pdf,” n.d.) suggests a linear regression method run in an artificial neural network to count crowds using head detection. Authors have used their own generated dataset to evaluate the method. There are certain limitations identified including the solution is sensible for the background of the input image and using an artificial neural network is very costly when it comes to computations. Overview of this solution is provided in Figure 4.

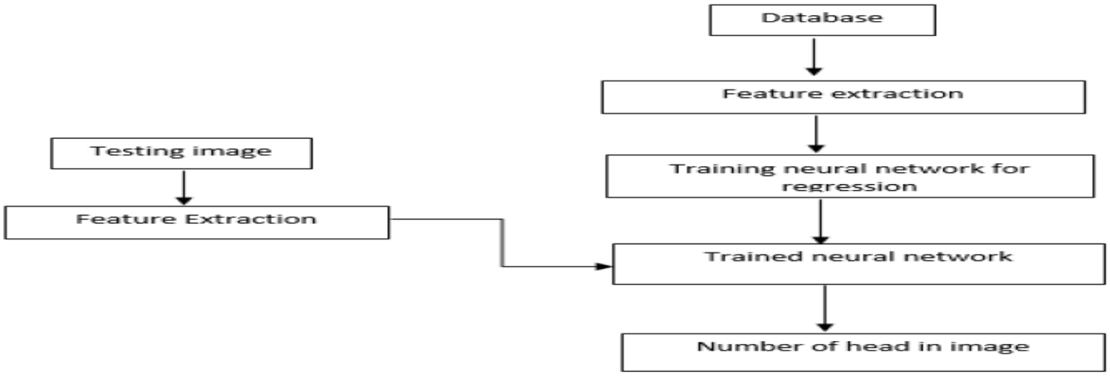


Figure 4. Overview of Manoj Jayaram’s Solution

2.3 Identification of the Research Gap

A review about previous work related to crowd counting techniques is provided through this section. However, very little research has been conducted on these identified crowd counting techniques using aerial related images. This section has revealed that the current crowd counting techniques have limitations when aerial related images are used as inputs which contain, ignorance of objects if they are camouflaged, occlusion occurring if objects are near in the images, inaccurate detection for a dense crowded image.

The main research question analyzed in this thesis is to overcome the revealed limitations on the current methods reviewed in this chapter when aerial related images are used as input. To answer this question this thesis suggests a method which contains a combination of a regression method run on a convolutional neural network based model.

3. Methodology

3.1 Introduction

The proposed solution illustrated in Figure 5, is to perform a bounding box regression on top of a trained model based on a convolutional neural network. The proposed solution consists of two main sections. A VGG-16 based convolutional neural network and a bounding box algorithm. The proposed solution works as follows. A base prediction is made using the CNN to classify whether any human objects are present in the input image. Then, the locations of the objects in the image are taken. If any object matches the base prediction made, the coordinates of that object are extracted and passed to the bounding box algorithm and boxes are drawn in these locations. Additionally, a non max suppression is performed to filter any redundant box drawn.

3.2 Proposed Solution

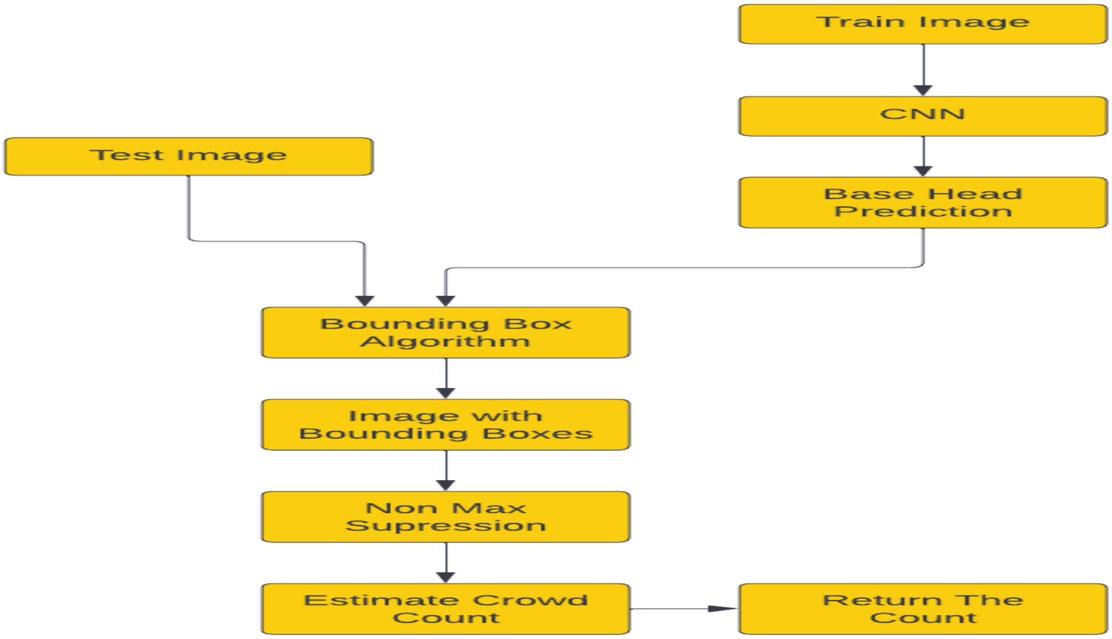


Figure 5. Overall model of the proposed solution

3.2.1 Datasets

Since the motivation is to build a solution to use aerial based images to get the count, relevant data sets should be used to train and test the proposed solution. Table 2 summarizes the details of the chosen datasets with their previous usages.

DataSet	Description	Usages
Crowd Detection Model (“Crowd Detection Model,”n.d.)	Created by Francisco Mena. has over 2000 composed of 640*480 pixel RGB images taken from a publicly accessible webcam. Contains a csv file which contains each image labeled with the count of the crowd. Used this data set as the ground truth. Includes images of low density crowds and high density crowds	Ekaterina Dranitsyna on her research on crowd counting using face detection.
Human Detection Dataset (“Human Detection Dataset,” n.d.)	Contains more than 921 images taken from high mountain CCTV cameras. Includes images under two categories, images of lower density crowds and high density crowds.	Constantine Werner on his research to count crowds using face detection
High Density Crowd (“High Density Crowd,” n.d.)	Over 250 images taken during several marathons held in the USA using high beam cameras set in buildings in the path of the marathons. Used in the training process	Ali and Mubarak Shah in their thesis (Ali and Shah, 2008), tracking individuals in high density crowd scenes.

Table 2. DataSet Selection Summary

3.3 Convolutional Neural Network

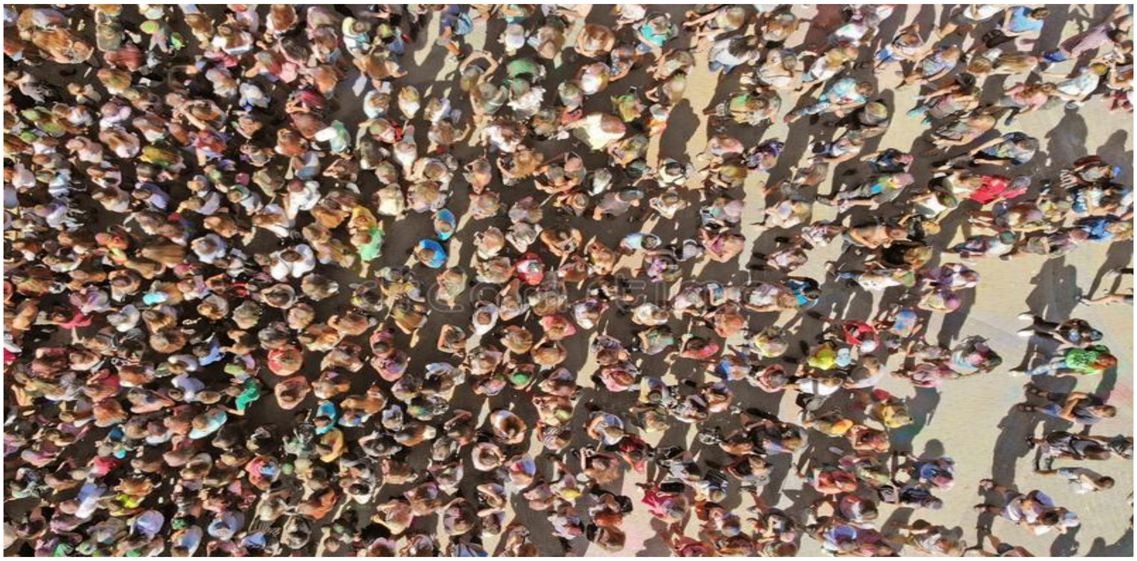


Figure 6. Sample drone image

Figure 6 is a sample drone image with a dense crowd. In this image a considerable number of people are captured. In some areas of the image a large number of people are located which results in people in those areas being captured close to each other and even merge with others. In traditional convolutional network based models this is considered to be a known limitation whereas this limits the accuracy of the count (Ilyas et al., 2020) .

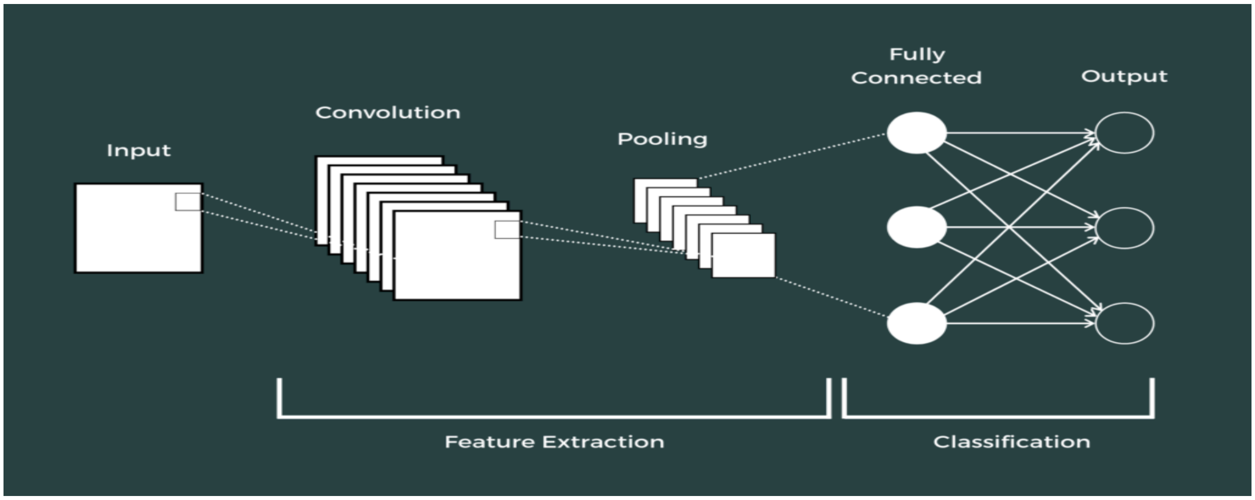


Figure 7. Structure of the Convolutional Neural Network

VGG-16 model is a pretrained CNN model mainly developed for object detection and classification with more than 92% accuracy. It contains an input layer, five convolutional layers and a dense output layer. The VGG-16 model can be customized by changing the input shape and by adding more layers to the output (Learning, 2021).

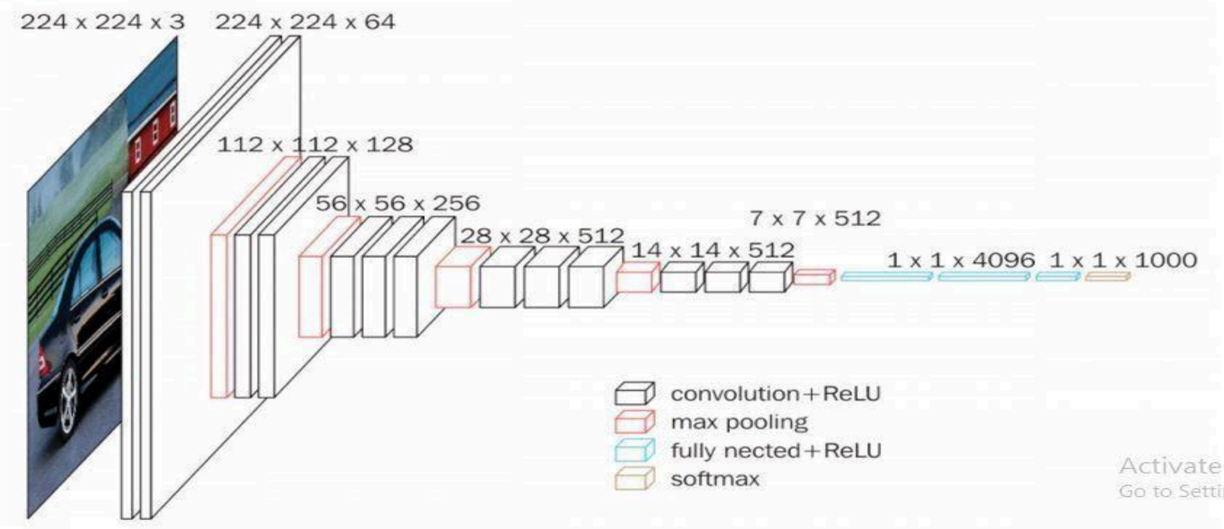


Figure 8. VGG-16 Architecture

The structure of the implemented VGG-16 based Convolutional Neural Network of the proposed solution is given in Figure 8. The VGG-16 based Convolutional Neural Network consists of 3 main layers. First layer is the input layer, which will take the input images and create a list of features called the feature map. This feature map will be the input for the next layer. The input layer will be trained to perform the following tasks. First task is to detect human objects in the provided image. A provided image may have many objects rather than humans which are not relevant to this approach. Therefore, the input layer should be trained to detect human objects from the others. Furthermore, neurons in the input layer should be trained to detect and separate the merged objects of the images before the processed data is passed to build the feature map.

Then, the detected human objects should be used to extract the feature for the feature map.When an image is passed to the input layer, the kernel in the layer converts this to a 2D numerical matrix. This will be used as the input to the hidden layer.

This matrix is passed to the neurons of the hidden layer. These neurons contain another matrix parameter called pattern filters. These pattern filters contain details on extracting the relevant features(eg:- head of the human) from the provided matrix.

Inside the neurons the convolutional operation is performed by performing the matrix multiplication between each pattern filter and each block of equal size as the filter in the converted image matrix. When the convolutional operation is performed, if the block in the image matrix is similar or identical to the feature matrix the return value of the operation will be maximized. Else values will be much lower than the maximum expected. This set of values is known as a feature map. The neural network will create a feature map for each filter.
Since the main feature used for this solution is the heads, pooling operation can be performed for the returned feature maps to filter and pass the maps which are classified as heads.

The output layer returns the prediction executed by the CNN. The original output layer of the VGG-16 is modified and implemented as an output layer containing four dense layers.

The training dataset is passed to the implemented CNN and the CNN generates a base head detection and this is retrieved by the bounding box algorithm to estimate the crowd count. This process is run multiple times for the training dataset before passing to the bounding box algorithm to train the machine.

During the training process, coordinates of the locations of the heads are passed to the CNN in order to perform the head detection training.

Figure 9 illustrates the details of the implemented CNN.

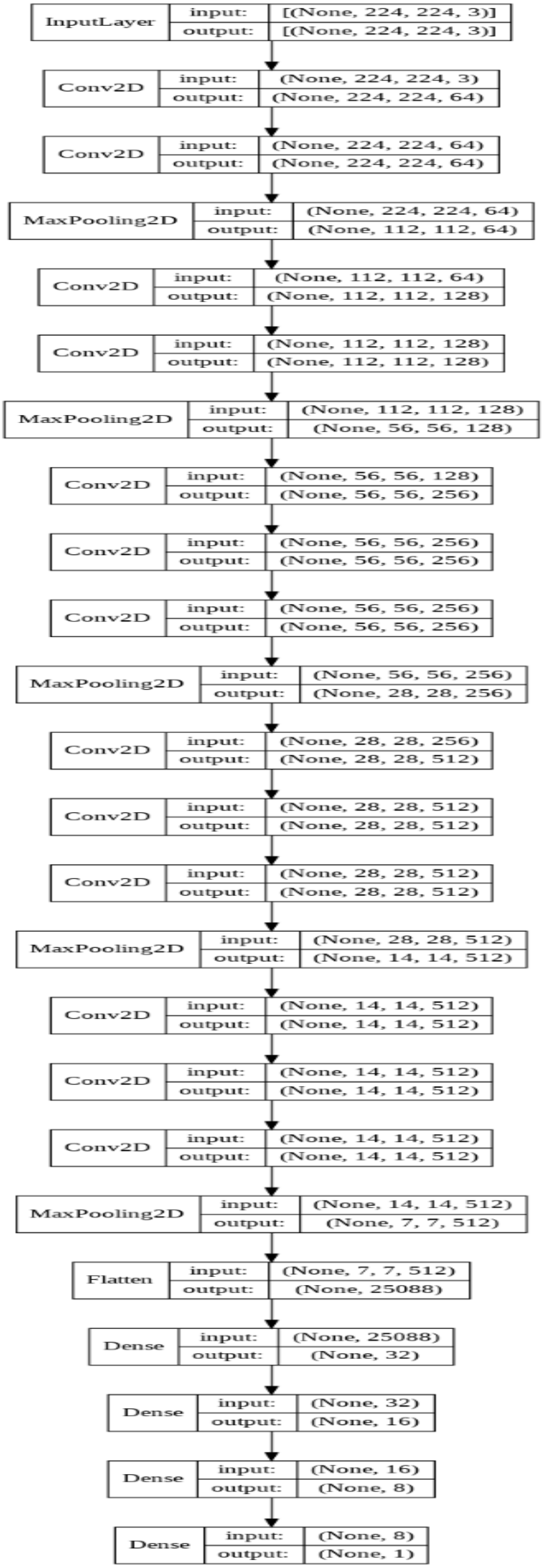

InputLayer | input: [(None, 224, 224, 3)] | output: [(None, 224, 224, 3)]
Conv2D | input: (None, 224, 224, 3) | output: (None, 224, 224, 64)
Conv2D | input: (None, 224, 224, 64) | output: (None, 224, 224, 64)
MaxPooling2D | input: (None, 224, 224, 64) | output: (None, 112, 112, 64)
Conv2D | input: (None, 112, 112, 64) | output: (None, 112, 112, 128)
Conv2D | input: (None, 112, 112, 128) | output: (None, 112, 112, 128)
MaxPooling2D | input: (None, 112, 112, 128) | output: (None, 56, 56, 128)
Conv2D | input: (None, 56, 56, 128) | output: (None, 56, 56, 256)
Conv2D | input: (None, 56, 56, 256) | output: (None, 56, 56, 256)
Conv2D | input: (None, 56, 56, 256) | output: (None, 56, 56, 256)
MaxPooling2D | input: (None, 56, 56, 256) | output: (None, 28, 28, 256)
Conv2D | input: (None, 28, 28, 256) | output: (None, 28, 28, 512)
Conv2D | input: (None, 28, 28, 512) | output: (None, 28, 28, 512)
Conv2D | input: (None, 28, 28, 512) | output: (None, 28, 28, 512)
MaxPooling2D | input: (None, 28, 28, 512) | output: (None, 14, 14, 512)
Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)
Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)
Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)
MaxPooling2D | input: (None, 14, 14, 512) | output: (None, 7, 7, 512)
Flatten | input: (None, 7, 7, 512) | output: (None, 25088)
Dense | input: (None, 25088) | output: (None, 32)
Dense | input: (None, 32) | output: (None, 16)
Dense | input: (None, 16) | output: (None, 8)
Dense | input: (None, 8) | output: (None, 1)

Figure 9. Implementation of the proposed CNN model

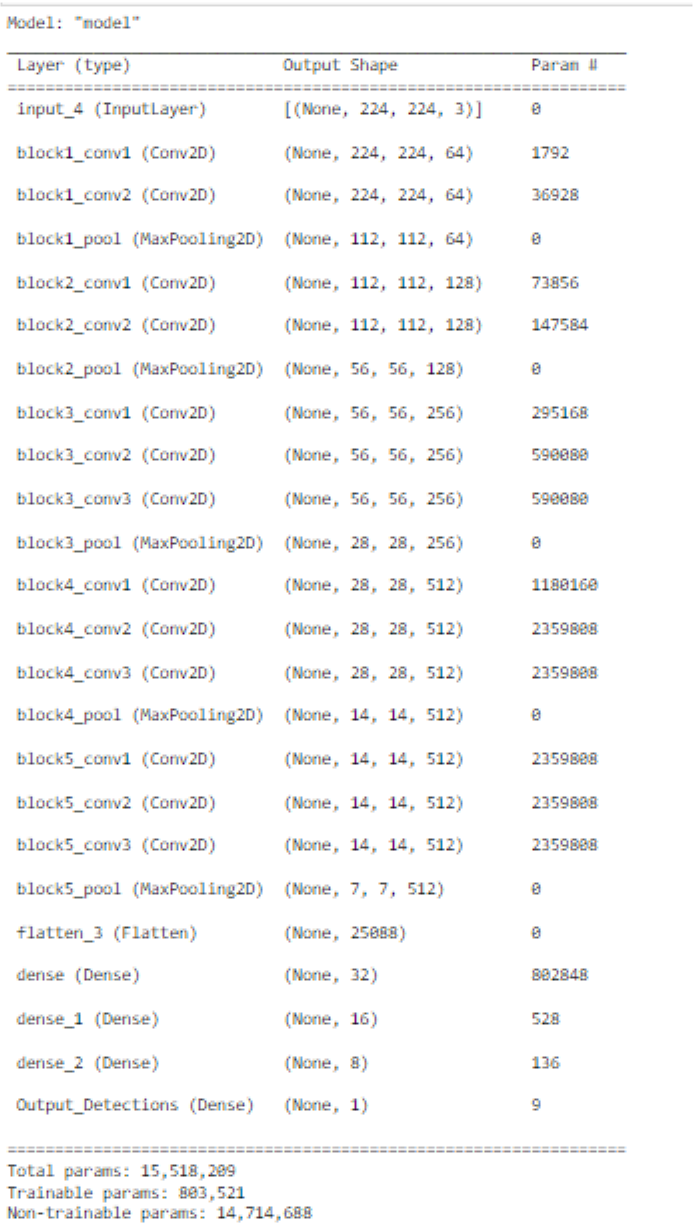
In between the convolutional and the output layers, pooling layers, dense layers and flatten layers are added in the implementation.

The main objective of the pooling layers is to remove unwanted filters from the neurons of the convolutional layers in order to avoid overfit the network and furthermore, to summarize the features from the generated feature maps. In this implementation, max pooling is used in order to maximize the presence of the features in the feature map.

The objective of the flatten layer is to convert multidimensional output of the convolutional layer to a single dimensional vector, which then is passed to the dense layer.

The objective of the dense layer is to classify the input image using the output returned from the convolutional layer. Two dense layers are added in the implementation, one using a Relu activation function and the other one using a sigmoid function. Relu function is used in order not to activate all the neurons at the same time. Sigmoid function is used for binary classifications of the features.

Figure 10 shows the descriptive view of the implemented CNN.



```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_4 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

 flatten_3 (Flatten)         (None, 25088)             0

 dense (Dense)               (None, 32)                802848

 dense_1 (Dense)             (None, 16)                528

 dense_2 (Dense)             (None, 8)                 136

 Output_Detections (Dense)   (None, 1)                 9

=================================================================
Total params: 15,518,209
Trainable params: 803,521
Non-trainable params: 14,714,688
```

Figure 10. Descriptive view of the CNN

3.4 Bounding Box Algorithm

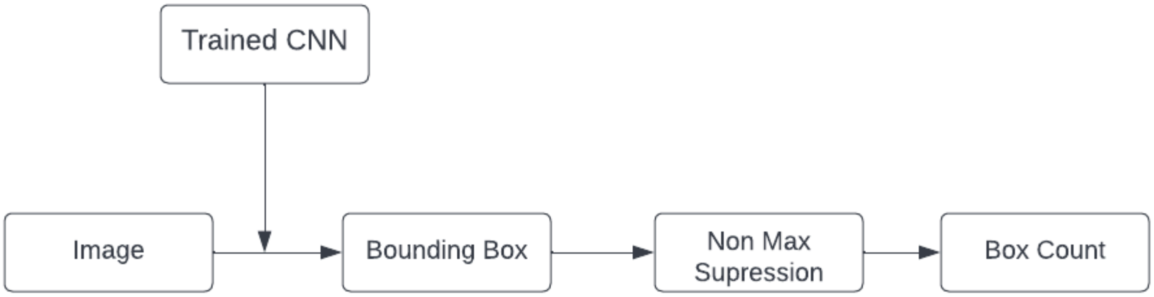


Figure 11. Structure of the Bounding Box Algorithm

The second section is the bounding box algorithm illustrated in Figure 11. Despite the fact that a CNN provides a high accuracy in object detection, there are limitations including occlusion occurring when objects are near in the image (Ilyas et al., 2020), (Gao et al., 2020). Therefore using a CNN alone to estimate the count of an aerial image with dense crowds will not provide an accurate count. To overcome this issue, the bounding box algorithm is introduced .

The implemented bounding box algorithm consists of two components, a bounding box algorithm and non max suppression function. The images in the testing dataset are used for this.

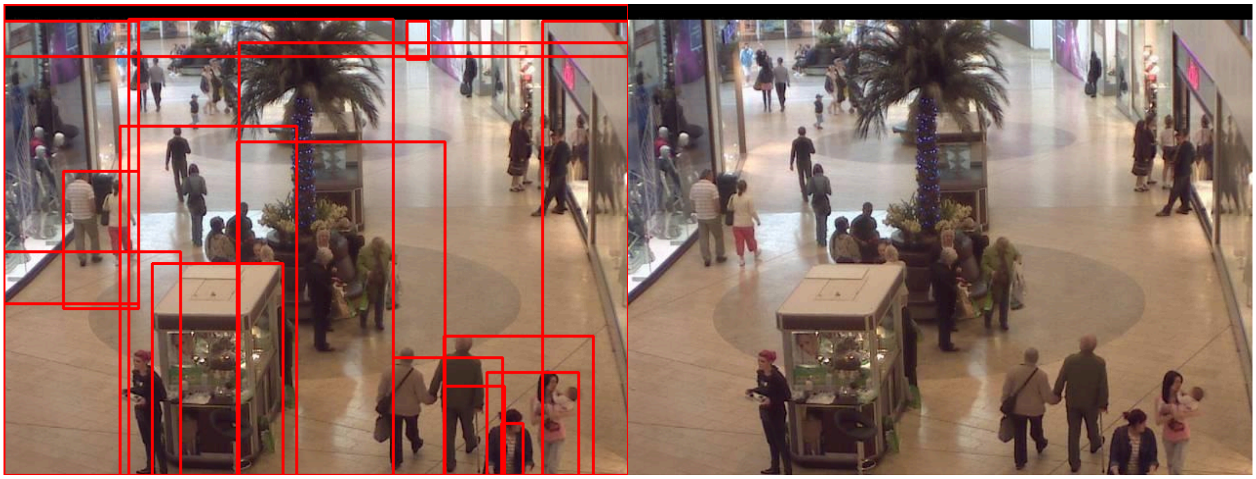
The bounding box algorithm is performed as follows. The algorithm accepts three parameters, the path of the image, the trained CNN and a reference variable which is initialized to zero, to store the predicted count. Once the path is passed, the corresponding image is loaded. Then the loaded image is passed to the trained CNN and using the CNN's trained knowledge on head detection using trained dataset, a base head classification is generated.

Using selective search segmentation, a search is performed to detect objects and the coordinates of the detected objects is returned. Bounding boxes are drawn in the regions of the coordinates which match with the base prediction classified as heads. Then, these selected coordinates are stored as an array of tuples and are passed to the non-max suppression function.

The main usage of the non-max suppression function is to remove redundant boxes drawn by the bounding box algorithm for the same head. When an object detection method for instance a bounding box method is used, multiple boxes can be drawn around the same object. When estimating the count, for this scenario, the same head could be counted more than one. Therefore, the non-max suppression function is introduced to overcome this (Bhat, n.d.).

The non-max suppression function accepts two parameters, the array of bounding boxes and the threshold value. The threshold value is defined as the maximum percentage of overlapping area any of the two bounding boxes are allowed to share. This function calculates the overlapping area of each bounding box and removes the boxes which have a value more than the provided threshold.

After the non-max suppression function returns the filtered set of bounding boxes, for each box, the reference variable value is increased by one.



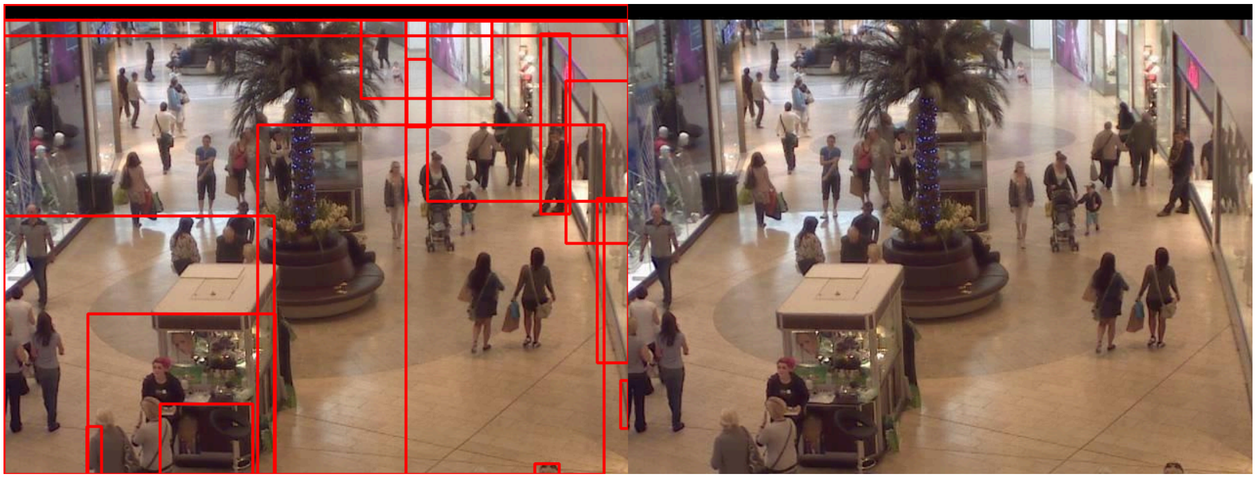


Figure 12. Input and Output Images through the model

Figure 12 contains two images in which, the right one is passed to the model and the and the left one which is after the model processed the image.

3.5 Implementation

The proposed solution is implemented using the following tools.

Keras

Keras is an open source neural network library written in Python programming language, provided by Tensorflow. The predefined VGG-16 based model is loaded from Keras and the output layer is modified to contain three Relu layers and one Sigmoid layer on which the output of the input image is returned. Keras also provides the functionality to save a trained model and load a saved trained model. Keras provides additional sub-libraries such as pre-processing. The preprocessing library contains functions such as ImageDataGenerator which the proposed solution used to generate a train-test split for the custom dataset provided and image_to_array which the proposed solution used to convert a provided image to a number array with a given shape (Team, n.d.) .

Scikit Learn

Scikit Learn is an open source library on machine learning written in Python programming language. It provides various classification, regression and clustering algorithms. Accuracy Score and Classification Matrix of this solution is generated using classification matrices provided by Scikit Learn ("scikit-learn: machine learning in Python — scikit-learn 1.2.1 documentation," n.d.).

OpenCV

OpenCV is an open source library used for real time computer vision operations. The proposed solution used OpenCV within the bounding box algorithm. First the image located in the provided input path of the algorithm is loaded through OpenCV. Then, Selective search segmentation algorithm provided by OpenCV is performed to the loaded image. The Selective search segmentation algorithm detects and returns the coordinates of all the objects in the loaded image. Additionally, the OpenCV rectangle function is used to draw the bounding boxes in the loaded image from the coordinates classified as heads. The OpenCV imshow() function is used to display the image with the bounding boxes (Bradski and Kaehler, 2011).

4. Evaluation and Results

4.1 Introduction

Evaluation will be performed under two categories. First category of the evaluation describes the evaluation of the training process whereas the second category describes the results and findings of the implemented model with a comparison of the ground truth count provided by the datasets.

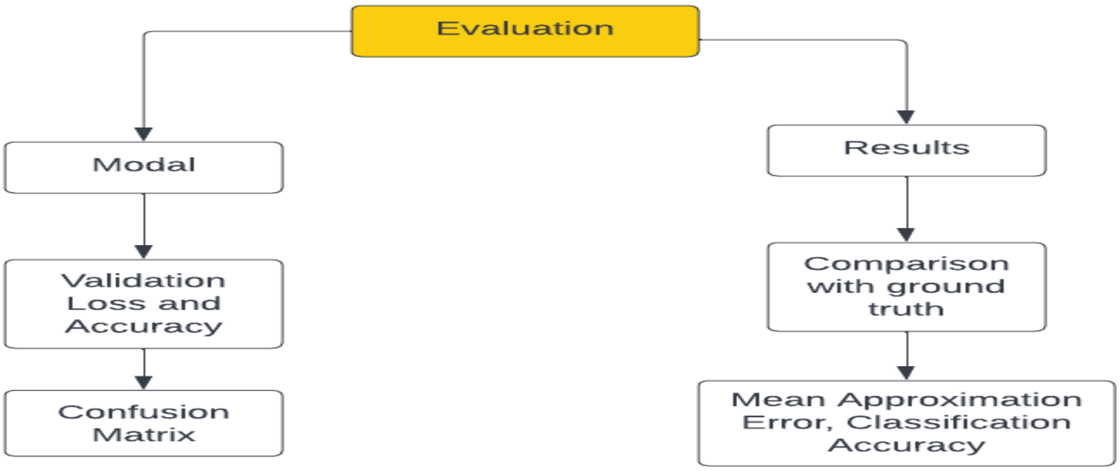
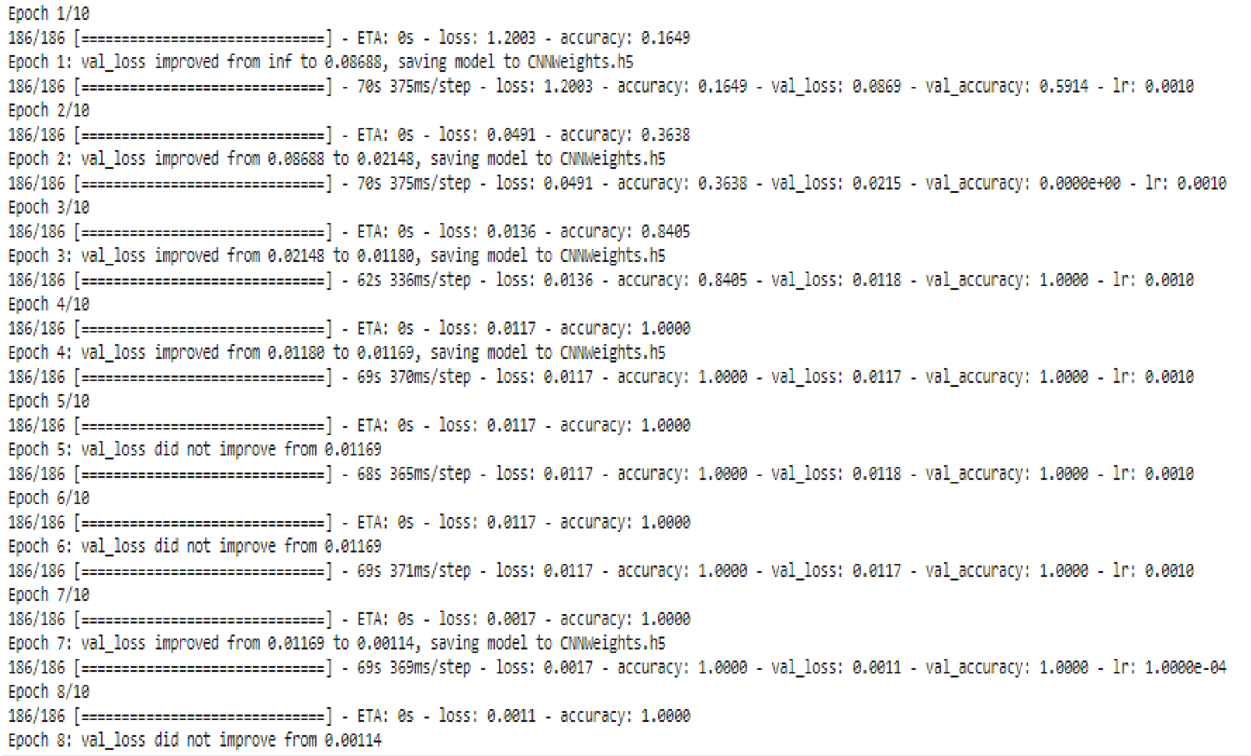

Evaluation
Modal
Validation Loss and Accuracy
Confusion Matrix
Results
Comparison with ground truth
Mean Approximation Error, Classification Accuracy

Figure 13. Overview of the evaluation

Figure 13 illustrates the overview of the planned evaluation.

4.2 Evaluating the Training Phase



```
Epoch 1/10
186/186 [==============================] - ETA: 0s - loss: 1.2003 - accuracy: 0.1649
Epoch 1: val_loss improved from inf to 0.08688, saving model to CNNWeights.h5
186/186 [==============================] - 70s 375ms/step - loss: 1.2003 - accuracy: 0.1649 - val_loss: 0.0869 - val_accuracy: 0.5914 - lr: 0.0010
Epoch 2/10
186/186 [==============================] - ETA: 0s - loss: 0.0491 - accuracy: 0.3638
Epoch 2: val_loss improved from 0.08688 to 0.02148, saving model to CNNWeights.h5
186/186 [==============================] - 70s 375ms/step - loss: 0.0491 - accuracy: 0.3638 - val_loss: 0.0215 - val_accuracy: 0.0000e+00 - lr: 0.0010
Epoch 3/10
186/186 [==============================] - ETA: 0s - loss: 0.0136 - accuracy: 0.8405
Epoch 3: val_loss improved from 0.02148 to 0.01180, saving model to CNNWeights.h5
186/186 [==============================] - 62s 336ms/step - loss: 0.0136 - accuracy: 0.8405 - val_loss: 0.0118 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 4/10
186/186 [==============================] - ETA: 0s - loss: 0.0117 - accuracy: 1.0000
Epoch 4: val_loss improved from 0.01180 to 0.01169, saving model to CNNWeights.h5
186/186 [==============================] - 69s 370ms/step - loss: 0.0117 - accuracy: 1.0000 - val_loss: 0.0117 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 5/10
186/186 [==============================] - ETA: 0s - loss: 0.0117 - accuracy: 1.0000
Epoch 5: val_loss did not improve from 0.01169
186/186 [==============================] - 68s 365ms/step - loss: 0.0117 - accuracy: 1.0000 - val_loss: 0.0118 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 6/10
186/186 [==============================] - ETA: 0s - loss: 0.0117 - accuracy: 1.0000
Epoch 6: val_loss did not improve from 0.01169
186/186 [==============================] - 69s 371ms/step - loss: 0.0117 - accuracy: 1.0000 - val_loss: 0.0117 - val_accuracy: 1.0000 - lr: 0.0010
Epoch 7/10
186/186 [==============================] - ETA: 0s - loss: 0.0017 - accuracy: 1.0000
Epoch 7: val_loss improved from 0.01169 to 0.00114, saving model to CNNWeights.h5
186/186 [==============================] - 69s 369ms/step - loss: 0.0017 - accuracy: 1.0000 - val_loss: 0.0011 - val_accuracy: 1.0000 - lr: 1.0000e-04
Epoch 8/10
186/186 [==============================] - ETA: 0s - loss: 0.0011 - accuracy: 1.0000
Epoch 8: val_loss did not improve from 0.00114
```

Figure 14. Training Summary

The implemented CNN is trained on the training image set with more than 900 images for 10 epochs. At the end of the first epoch, CNN shows 0.1649 accuracy and a loss of 1.2 with 0.5914 validation accuracy and a validation loss of 0.0869. At the end of the rest of the epochs, accuracy and the validation accuracy value increases above the previous epoch indicating the CNN is training. From the fifth epoch, accuracy and the validation accuracy approaches to 1.0 and the loss value decreases in every epoch. Figure 14 contains the training summary of the implemented model.

In order to control overfitting during the training, EarlyStopping, ReduceLROnPlateau and ModelCheckpoint callback functions are used. EarlyStopping function validates whether the loss value is improved during the epochs. If loss value is not improved, it is assumed that the training is complete and the training is finished. The ReduceLROnPlateau function monitors the validation loss value and if it is not improving during the epochs, the learning rate is decreased by this function. The ModelCheckpoint function is used to save the weights generated by the neurons in the CNN. Once these weights are saved, they can be loaded to the CNN directly to perform the training.

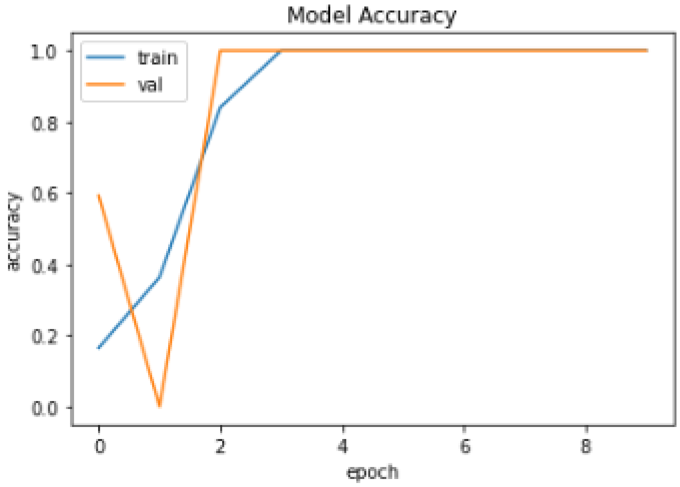


Figure 15. CNN model Accuracy

Figure 15 shows the plotted training and validation accuracies with the number of epochs. At the end of the first epoch, 0.1649 accuracy and 0.5914 validation accuracy were shown. The values increased with the number of epochs and after the 4th epoch, the maximum accuracy reached.

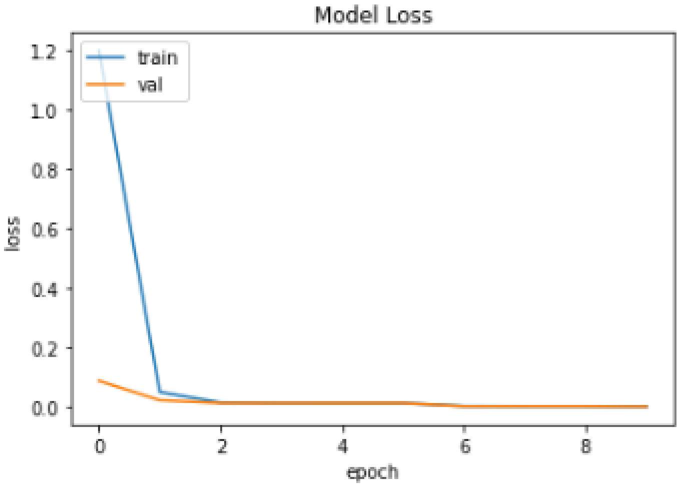


Figure 16. CNN model Loss

Figure 16 shows CNN's loss variance with the number of epochs. At the end of the first epoch, 1.2 loss and 0.082 validation loss were shown. The values are reduced with the number of epochs.

The Confusion Matrix presented in Figure 17, will be generated using predictions generated by the datasets and the datasets used to train the neural network. Using the training dataset, the feature matrix will be adjusted To lower the false negatives and false positives. Additionally, the confusion matrix is used to measure the accuracy and sensitivity.

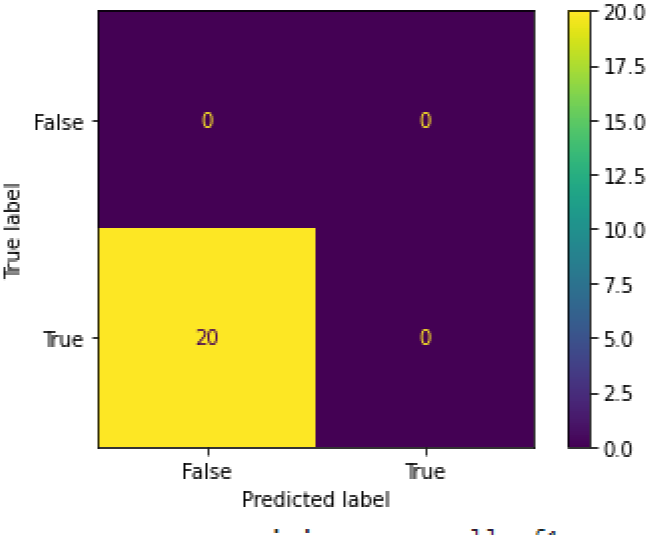
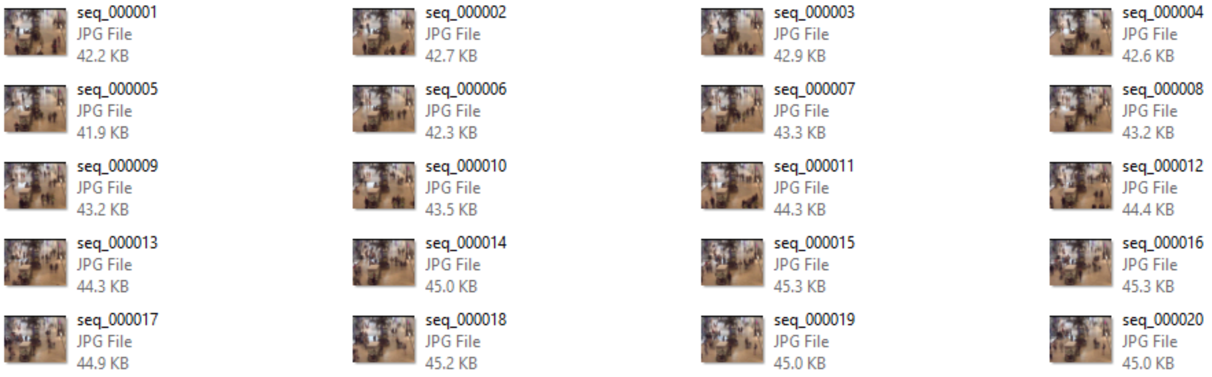


Figure 17. Confusion Matrix of the trained model

4.3 Evaluating Results

For the evaluation, a portion of 20 images presented in Figure 18, taken from the testing dataset was used.



seq_000001 JPG File 42.2 KB	seq_000002 JPG File 42.7 KB	seq_000003 JPG File 42.9 KB	seq_000004 JPG File 42.6 KB
seq_000005 JPG File 41.9 KB	seq_000006 JPG File 42.3 KB	seq_000007 JPG File 43.3 KB	seq_000008 JPG File 43.2 KB
seq_000009 JPG File 43.2 KB	seq_000010 JPG File 43.5 KB	seq_000011 JPG File 44.3 KB	seq_000012 JPG File 44.4 KB
seq_000013 JPG File 44.3 KB	seq_000014 JPG File 45.0 KB	seq_000015 JPG File 45.3 KB	seq_000016 JPG File 45.3 KB
seq_000017 JPG File 44.9 KB	seq_000018 JPG File 45.2 KB	seq_000019 JPG File 45.0 KB	seq_000020 JPG File 45.0 KB

Figure 18. Testing Images Portion

Table 3 is a summary of the expected and the actual counts of the selected portion.

Id	Expected Count	Actual Count
1	**35**	**23**
2	**41**	**25**
3	**41**	**16**
4	**44**	**17**
5	**41**	**18**
6	**41**	**23**
7	**35**	**14**
8	**36**	**12**
9	**27**	**16**
10	**24**	**10**
11	**16**	**10**
12	**22**	**16**
13	**23**	**16**
14	**25**	**12**
15	**15**	**9**
16	**16**	**18**
17	**15**	**12**
18	**25**	**16**
19	**31**	**23**
20	**25**	**15**

Table 3. Comparison of expected and actual count

Table 4 presents the calculated classification accuracy and mean approximation for each image in the test portion.

Id	Classification Accuracy	Mean Approximation Error
1	**0.6571**	**0.3429**
2	**0.6097**	**0.3903**
3	**0.3903**	**0.6097**
4	**0.3463**	**0.6537**
5	**0.439**	**0.561**
6	**0.5609**	**0.4391**
7	**0.4**	**0.6**
8	**0.333**	**0.667**
9	**0.5925**	**0.4075**
10	**0.4167**	**0.5833**
11	**0.625**	**0.375**
12	**0.7272**	**0.2828**
13	**0.6956**	**0.3044**
14	**0.48**	**0.52**
15	**0.6**	**0.4**
16	**1.125**	**-0.875**
17	**0.8**	**0.2**
18	**0.64**	**0.36**
19	**0.7419**	**0.2581**
20	**0.6**	**0.4**

Table 4. Count Evaluation

Since both the total count as well as count of each image is provided in the datasets, Classification Accuracy will be calculated for the count of the each image as well as for the total Count to measure the accuracy of the count. Using training dataset, the feature matrix which is passed to the hidden layer will be adjusted to get the accuracy above 90%.

$$Classification\ Accuracy\ =\ No\ of\ correct\ predictions\ \div\ Total\ Predictions \quad (1)$$

Mean Approximation Error is also used to compare the similarity of the actual count with the expected count provided by the dataset. If the MAE is closer to 0, then it is said that the actual count is closer to the expected count.

$$Mean\ Approximation\ Error\ =Actual\ Count\ -\ Expected\ Count\ \div\ Expected\ Count\ (2)$$

The above table represents the calculated classification accuracy and the mean approximation errors of the selected portion.

4.3 Comparison with existing work

Jayaram ("Jayaram - I certify that this thesis does not incorporate wi.pdf," n.d.) have shown a mean approximation error of 0.152 for a CNN based solution for sparse crowd images taken from high beam cameras. Alotaibi (Alotibi et al., 2019b) have shown a mean approximation error of 0.06 for a VGG-16 CNN based solution for sparse crowd images. Tang (Tang et al., 2021) have shown a mean approximation error of 0.07 a Density map estimation based solution on a dense image dataset of 1500 images. Lian (Lian et al., 2019) have shown a mean approximation error of 0.08 for a regression based solution for sparse crowd images. (Shao et al., 2018) have shown a recall value of 0.9. Khan (Khan et al., 2021) have shown a mean approximation error of 0.08 for a cluster based crowd counting solution on sparse crowd images. These existing solutions are tested on non aerial images whereas the proposed solution shows a mean approximation error of 0.35 and a classification accuracy of 0.65 for dense crowd aerial images where the head detection is used..

5 Conclusion

5.1 Conclusion

This thesis proposed a method which includes a bounding box regression method running on top of a VGG-16 based CNN model to detect crowd and acquire the count from each image passed. The CNN model is trained to detect the human heads from the provided aerial input image. Bounding box regression method is used to draw bounding boxes of the locations which are classified as human heads by the trained model. The purpose of this thesis is to provide a model which improves the identified limitations of the current techniques for aerial related images. The proposed solution is evaluated with a portion of 20 images and has shown a mean approximation error of 0.35 and a classification accuracy of 0.65 for dense crowd aerial images.

5.2 Future Work

The current method uses a VGG-16 based convolutional neural network as the detection model (More than 5 hours to train a 1200 amount of images in 5 epochs) . By using a larger convolutional kernel and by reducing pre-implemented fully connected layers, training time will be reduced.Bounding box algorithm implemented in the current method depends too much on the base prediction made by the model. If the base prediction is incorrect, the returned count is also incorrect. By improving the feature extraction in the training process and performing the detection process of the convolutional neural network to the locations of the objects listed from the image relying on the base prediction will be reduced. Bounding box algorithm implemented in the current method shows the possibility of having a single box drawn capturing multiple humans. In such cases, the prediction count is incorrect. By improving the detection process as well as improving the threshold of the non max suppression algorithm, multiple objects captured from one drawn box will be fixed.

References

3 dead and 9 injured in a stampede in Maligawatte; 6 arrested [WWW Document], n.d. URL https://www.newsfirst.lk/2020/05/21/3-dead-and-4-injured-in-a-stampede-in-maligawatte/ (accessed 10.4.21).

Ali, S., Shah, M., 2008. Floor Fields for Tracking in High Density Crowd Scenes, in: Forsyth, D., Torr, P., Zisserman, A. (Eds.), Computer Vision – ECCV 2008, Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1–14. https://doi.org/10.1007/978-3-540-88688-4_1

Alotibi, M.H., Jarraya, S.K., Ali, M.S., Moria, K., 2019a. CNN-Based Crowd Counting Through IoT: Application For Saudi Public Places. Procedia Comput. Sci., 16th Learning and Technology Conference 2019Artificial Intelligence and Machine Learning: Embedding the Intelligence 163, 134–144. https://doi.org/10.1016/j.procs.2019.12.095

AP count: Over 2,400 killed in Saudi hajj stampede, crush [WWW Document], 2015. . AP NEWS. URL https://apnews.com/article/mecca-dubai-united-arab-emirates-archive-saudi-arabia-3a42a7733a8b476889bb4b7b3be3560e (accessed 4.23.22).

Astroworld Festival crowd crush, n.d.

Bhat, A., n.d. Aerial Object Detection using Learnable Bounding Boxes 95.

Bradski, G.R., Kaehler, A., 2011. Learning OpenCV: computer vision with the OpenCV library, 1. ed., [Nachdr.]. ed, Software that sees. O'Reilly, Beijing.

Brostow, G., n.d. Unsupervised Bayesian Detection of Independent Motion in Crowds.

Dar es Salaam Stampede, n.d.

Gao, G., Gao, J., Liu, Q., Wang, Q., Wang, Y., 2020. CNN-based Density Estimation and Crowd

Counting: A Survey.

Gong, S., Bourennane, E.-B., 2018. Implementation of real time reconfigurable embedded architecture for people counting in a crowd area, in: Chikhi, S., Amine, A., Chaoui, A., Saidouni, D.E. (Eds.), 5th International Symposium on Modelling and Implementation of Complex Systems (MISC 2018), Modelling and Implementation of Complex Systems. Springer, Laghouat, Algeria, pp. 219–229. https://doi.org/10.1007/978-3-030-05481-6_17

High Density Crowd [WWW Document], n.d. URL https://www.kaggle.com/datasets/danaelisanicolas/high-density-crowd-counting (accessed 11.1.22).

Human Detection Dataset [WWW Document], n.d. URL https://www.kaggle.com/datasets/constantinwerner/human-detection-dataset (accessed 2.18.23).

Ilyas, N., Shahzad, A., Kim, K., 2020. Convolutional-Neural Network-Based Image Crowd Counting: Review, Categorization, Analysis, and Performance Evaluation. Sensors 20, 43. https://doi.org/10.3390/s20010043

Jayaram - I certify that this thesis does not incorporate wi.pdf, n.d.

Khan, K., Khan, R.U., Albattah, W., Nayab, D., Qamar, A.M., Habib, S., Islam, M., 2021. Crowd Counting Using End-to-End Semantic Image Segmentation. Electronics 10, 1293. https://doi.org/10.3390/electronics10111293

Kim, J., Kuhn, A., 2022. Over 150 dead after Halloween crowd surge in Seoul. NPR.

Learning, G., 2021. Everything you need to know about VGG16. Medium. URL https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918 (accessed 11.5.22).

Li, Z., Zhou, Y., Xiao, S., He, C., Huang, Z., Li, H., 2018. Auto-Conditioned Recurrent Networks for Extended Complex Human Motion Synthesis. ArXiv170705363 Cs.

Lian, D., Li, J., Zheng, J., Luo, W., Gao, S., 2019. Density Map Regression Guided Detection Network for RGB-D Crowd Counting and Localization, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Long Beach, CA, USA, pp. 1821–1830. https://doi.org/10.1109/CVPR.2019.00192

Mousas, C., 2017. Full-Body Locomotion Reconstruction of Virtual Characters Using a Single Inertial Measurement Unit. Sensors 17, 2589. https://doi.org/10.3390/s17112589

New York Times, n.d. Meron Stampede.

Ratcliffe, R., Connett, D., Fulton, A., 2022. 125 dead after crowd crush at Indonesian football match. The Guardian.

scikit-learn: machine learning in Python — scikit-learn 1.2.1 documentation [WWW Document], n.d. URL https://scikit-learn.org/stable/ (accessed 2.18.23).

Shao, S., Zhao, Z., Li, B., Xiao, T., Yu, G., Zhang, X., Sun, J., 2018. CrowdHuman: A Benchmark for Detecting Human in a Crowd.

Tang, J., Zhou, M., Li, P., Zhang, M., Jiang, M., 2021. Crowd Counting Based on Multiresolution Density Map and Parallel Dilated Convolution. Sci. Program. 2021, e8831458. https://doi.org/10.1155/2021/8831458

Team, K., n.d. Keras documentation: Keras API reference [WWW Document]. URL https://keras.io/api/ (accessed 2.18.23).

Vaishno Devi Stampede, n.d.

Yao et al. - Deep Spatial Regression Model for Image Crowd Coun.pdf, n.d.

Human Detection Dataset [WWW Document], n.d. URL