



Applications for Smart Airport

Deep learning-based population density counting in
Hong Kong International Airport.

Final Report

Supervisor:

Dr. T W Chim

Author:

Mak Chak Wing 3035564732

Abstract

Population density estimation, also known as crowd counting, is one of the vital tasks for the Hong Kong Airport Authority to allocate resources and enhance the operational efficiency of the Hong Kong International Airport. However, the current manual counting method is labor-intensive and may suffer from inaccuracy. Therefore, the project targets to explore the deep learning-based crowd counting method and queuing flow rate prediction. In the project, the most essential part is designing the deep learning network architecture for the crowd counting model and tracker model. The project also requires a distributed backend to serve numerous surveillance cameras and an informative frontend dashboard for Hong Kong Airport Authorities. To aggregate data for the project, the experiment is conducted in different Hong Kong vaccination centers because of the severe COVID-19 situation and compliance issue of airport CCTVs. Data collection was performed in different Hong Kong vaccination centers along with a survey report of the latest convolutional neural network in the crowd counting problem, focusing on density map estimation; An experiment about bounding box regression problem to find out the performance and accuracy of detection model is carried out; meanwhile, the overall system design of the infrastructure is completed. After the data collection and system design, the next step consists of the development of the web applications and the integration between well-trained deep learning models and the web application with different data visualization methods.

Acknowledgements

I would like to express my appreciation to the Department of Computer Science, and Dr. TW Chim for offering this hands-on real-life project with the Hong Kong Airport Authority and supporting our team throughout the project.

I would like to recognize Hong Kong Airport Authority's staff for providing the information of the airport, and the staff in Teksbotics, which is the vendor of the airport patrol robot.

Moreover, I would like to express my gratitude to my teammates throughout the collaboration.

Lastly, A special thanks to Ms. Mable Choi for assisting with the report preparation and presentation.

Table of Contents

Abstract.....	1
Acknowledgements.....	2
List of Figures.....	5
List of Tables.....	6
Abbreviations & Acronyms.....	6
1. Introduction.....	7
1.1 Background.....	7
1.1.1 Big Data.....	7
1.1.2 Motivation of Crowd Counting.....	7
1.1.3 Benefits of Crowd Counting.....	8
1.1.4 Current Approach in Population Density Counting.....	8
1.2 Proposed Approach.....	9
1.2.1 Deep Learning.....	9
1.2.2 Visualization.....	10
1.3 Project Objectives.....	10
1.4 Project contribution.....	11
1.5 Outline of Report.....	12
2. Methodology.....	13
2.1 Introduction.....	13
2.2 Human Localization Problem.....	13
2.2.1 Convolutional neural networks.....	13
2.2.2 Bounding Box Regression.....	14
2.2.3 Density Map Generation.....	15
2.3 MLOps.....	15
2.3.1 Data Management.....	15
2.3.2 Model Learning.....	16
2.3.3 Model Deployment.....	17
2.3.4 Processing.....	18
2.3.5 Additional Features.....	19
2.4 Software Implementation.....	19
2.5 Application Design.....	20
2.5.1 Frontend.....	20

2.5.2	Backend.....	20
2.6	Data visualization.....	21
2.6.1	D3.js	21
2.7	Summary	21
3.	Results and Discussion.....	22
3.1	Density Map Generation Model Survey.....	22
3.1.1	Dataset Selection	22
3.1.2	Data Preprocessing	23
3.1.3	Training Options.....	23
3.1.4	Evaluation Metrics.....	24
3.1.5	Model Architecture.....	24
3.1.6	Result.....	28
3.1.7	Model Evaluation Results.....	31
3.1.8	Summary.....	32
3.2	Deepstream Queuing Throughput Accuracy.....	33
3.3	Analysis of Human Size inside a Frame	34
3.3.1	Methodology.....	34
3.3.2	Result.....	36
3.3.3	Summary.....	41
3.4	System Design.....	42
3.4.1	MLOps.....	43
3.4.2	Backend.....	44
3.4.3	Frontend.....	45
3.5	Future plans.....	47
3.5.1	Analysis of detection model and tracker	47
3.5.2	Queue Identification	48
3.5.3	Queuing Flow Rate Estimation.....	48
3.6	Challenges and mitigation.....	49
4.	Conclusion.....	50
	References.....	51

List of Figures

FIGURE 1.1 OVERVIEW OF DEEP NEURAL NETWORK [4]	9
FIGURE 2.1 NEURAL NETWORK STRUCTURE OF CNNs [5].....	13
FIGURE 2.2 OVERVIEW OF R-CNN [7]	14
FIGURE 2.3 DIFFERENCE BETWEEN DENSITY DESTINATION AND DENSITY GENERATION [9]	15
FIGURE 2.4 DEEPSTREAM BRIEF SUMMARY.....	18
FIGURE 2.5 APPLICATION FLOW CHART	19
FIGURE 2.6 EXAMPLE OF HEATMAP.....	21
FIGURE 3.1 NETWORK ARCHITECTURE OF MODIFIED VGG [15]	25
FIGURE 3.2 NETWORK ARCHITECTURE OF MODIFIED RESNET50 [16].....	26
FIGURE 3.3 NETWORK ARCHITECTURE OF MODIFIED MCNN [17]	27
FIGURE 3.4 NETWORK ARCHITECTURE OF MODIFIED ALEXNET [18]	27
FIGURE 3.5 RESULT OF CLASSIFICATION MODEL FOR THE SHANGHAI TECH PART B DATASET.....	31
FIGURE 3.6 RESULT OF CLASSIFICATION MODEL FOR THE UCF-QNRF DATASET	32
FIGURE 3.7 GALLERY OF INFERENCED IMAGES.....	35
FIGURE 3.8 PAIRPLOT OF THE RESULT OF THE ANALYSIS OF HUMAN SIZE INSIDE A SCALED DOWN IMAGE	37
FIGURE 3.9 SCATTER PLOT BETWEEN SCALING FACTOR AND THE PIXEL COUNT OF SMALLEST BOUNDING BOX.....	38
FIGURE 3.10 SCATTER PLOT BETWEEN SCALING FACTOR AND THE NUMBER OF BOUNDING BOXES	39
FIGURE 3.11 SCATTER PLOT BETWEEN NUMBER OF BOUNDING BOXES AND THE PIXEL COUNT OF SMALLEST BOUNDING BOX	40
FIGURE 3.12 OVERVIEW OF SYSTEM DESIGN	42
FIGURE 3.13 WEBSOCKET SEQUENCE DIAGRAM	44
FIGURE 3.14 DENSITY HEATMAP.....	45
FIGURE 3.15 DENSITY HEATMAP HOVER STATE.....	45
FIGURE 3.16 SCREENSHOT OF QUEUING FLOW RATE ESTIMATION APPLICATION.....	46
FIGURE 3.17 NEURAL NETWORK STRUCTURE OF RNNs [19].....	48

List of Tables

TABLE 2.1 DESCRIPTION OF DATA PREPROCESSING AND DATA ANALYSIS 16

TABLE 3.1 IMAGES SPECIFICATION OF THE DATASET 22

TABLE 3.2 DEEPSTREAM QUEUING OUTPUT ACCURACY 33

TABLE 3.3 RESULT OF THE ANALYSIS OF HUMAN SIZE INSIDE A SCALED DOWN IMAGE..... 36

Abbreviations & Acronyms

Abbreviations & Acronyms	Definition
AI	Artificial Intelligence
BBOX	Bounding Box
CNN	Convolutional neural network
ER	Entity Relation
FLOPS	Floating-point operations per second
GPU	Graphics Processing Unit
HKAA	Hong Kong Airport Authority
HKIA	Hong Kong International Airport
HKU	University of Hong Kong
KLT	Kanade-Lucas-Tomasi
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MCNN	Multi-Column Convolutional Neural Network
ML	Machine Learning
MSE	Mean Squared Error
MPEG	Moving Picture Experts Group
NMS	Non-maximum Suppression
R-CNN	Region-based Convolutional Neural Network
RNN	Recurrent Neural Network
RTSP	Real Time Stream Protocol

1. Introduction

1.1 Background

The Airport Authority Hong Kong (HKAA) has put a lot of focus on transforming Hong Kong International Airport (HKIA) into a smart airport as this brings huge benefits to minimizing the operational cost and providing a better travel journey for passengers. For example, using a patrol robot as a customer service center. This technology integration can show that Hong Kong is an advanced and vivid city. Due to Covid-19, passenger number has fluctuated a lot in HKIA, which creates a challenge for HKAA to monitor the passenger flow. Therefore, HKIA decides to make use of big data to create an all-in-one solution to monitor passenger flow and queuing throughput to provide more insight into management, marketing, and operational aspect [1].

1.1.1 Big Data

Big data intelligence is one of the five vital technologies to fulfill the smart airport vision and is the backbone of the monitoring passenger flow application, which manipulate the use of artificial intelligence and machine learning algorithm. These technologies require an enormous dataset to uncover the hidden pattern of passenger flow and generate more useful information. One of the significant examples is the self-check-in counter, in which passengers can proceed with the check-in without any staff. This can reduce labor costs and provide service 24/7.

Hong Kong Airport Authority is thrilled with technology and has released a series of technologies, such as advanced biometric credentials powered by facial recognition, multi-functional patrol robots, and the customer chatbot.

1.1.2 Motivation of Crowd Counting

As HKIA is one of the busiest airports, HKAA is required to handle enormous fluctuations in the number of travellers during different periods, such as the drop in passenger numbers during the COVID-19 period, the drastic increase of passengers during the summer holidays. HKAA must react quickly in terms of allocation of manpower and resources while facing a challenge, monitoring the visitor number manually, and preventing assigning excessive resources.

The HKAA has an obstacle in accurately tracking passenger flow and understanding the passenger shopping behavior inside HKIA, as the passenger traffic involved a lot of factors, such as economic situation, human psychology, and global accident. Real-time information gathering would be a far more accurate method of providing vital knowledge for optimizations to be performed.

1.1.3 Benefits of Crowd Counting

Real-time population monitoring can bring HKAA three benefits in terms of energy-saving; reduction of manpower and resources management. One of the direct improvements is the reduction in energy consumption is the air-conditioning system, which consumes 25% of overall energy usage in HKIA [2]. With the population density data, HKAA can reduce air-conditioning in some less dense areas to save energy instead of adjusting overall air-conditioning. Furthermore, HKAA can accurately estimate the shop value according to the pedestrian flow and population density.

Other benefits are the approximation of labor force and resource allocation. There are numerous queues inside HKIA, and this may become a bottleneck if the resource allocated is not sufficient. For example, the check-in counter, onboarding counter, and border control center. Monitoring the service demand and creating an alert system for HKAA is vital to smartly allocate resource and this may boost customer satisfaction and make sure HKIA maintain the busiest airport in the world. The alert system also reminds HKAA staff to allocate more labor force and set up a better queue environment for passengers.

1.1.4 Current Approach in Population Density Counting

Although population density counting brings benefits to the airport's operating efficiency, an automatic pipeline is lacking. Nowadays, HKAA adopted a manual counting strategy, in which staff is assigned to a designated area to count the number of people within a certain period. This method is labor-intensive and results in high inaccuracy due to human error. Also, the sample size is small as the manual collection can only sustain for a few hours. To conclude, manual counting has the biggest bottleneck when scaling up.

To solve the problems, HKAA and we proposed a deep learning-based and automated pipeline in population density counting to resolve the above problems for the smart airport vision.

1.2 Proposed Approach

In this project, a fully automated and scalable crowd counting solution will be developed with deep learning. Also, this project survey the limitation of deep learning-based crowding counting, such as comparing different methodologies and the accuracy of the deep learning model as HKIA has high roofs which is usually higher than 4m ,and human is rather small in CCTV. Moreover, visualization is another focus of this project as visualization is a direct medium to provide insight to HKAA’s staff.

1.2.1 Deep Learning

Deep learning is a subfamily of machine learning, which simulates the human neuron by using a feed-forward neural network to reveal the hidden information from the input data [3] .

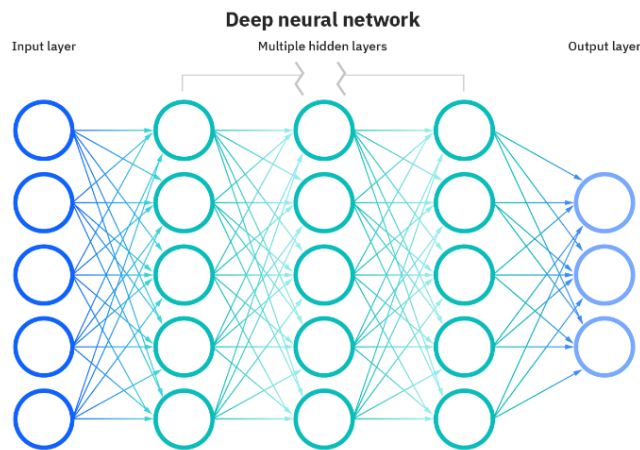


Figure 1.1 Overview of Deep Neural Network [4]

Figure 1.1 shows the network architecture of the deep learning model, which contains multiple layers of interconnected nodes; The visible layer of the deep learning model is the input and output layer. The input layer is the layer that translates the data into information and the output layer handles the prediction or the categorization of the data. The hidden layers between the input and output will capture the characteristics of the class, which helps the prediction. In our approach, deep learning is applied to people density counting and queuing flow rate estimation. Regarding people density counting, A well-trained deep learning model can localize most of the individuals. For the queuing flow rate estimation, a line-crossing algorithm is needed to act as a counter.

1.2.2 Visualization

Although this project emphasizes analysis of the performance of real-time crowd counting and queuing throughput system, visualization is also a critical part as it is the form of presentation of the output data. The use of visualization makes the efficiency of the deep learning model clear, while also acting as a prototype to demonstrate the possibilities of people movement detection solutions. By allowing users to engage with the prototype, each user can find patterns and trends that have business significance for their job responsibilities. The prototype presents the queueing information and the counter throughput through an interactive web portal. This solution should assist users in understanding the possibilities for business intelligence and may also propose new metrics to be counted or evaluated as a system enhancement.

1.3 Project Objectives

In this project, we aim to develop a population density counting and queuing flow rate estimation application in HKIA using the simulation in HKU.

The scope of this project will be divided into two parts. The first part is the web application for density data visualization which consists of backend and frontend, while the second part is the deep learning models for crowd counting. Below are our major tasks:

- Compare state-of-art deep learning models and Machine learning algorithms in terms of effectiveness in crowd counting
- Extract people density information from inference result
- Evaluate effectiveness of human tracking and queue counting
- Apply MLOps techniques in model optimization and data management to obtain results for evaluation
- Visualize density data in heatmap as well as queue statistics on web application
- System integration of MLOps component, backend and frontend as a prototype solution

1.4 Project contribution

Deep learning-based solutions are anticipated to be integrated into the smart city concept, such as multi-function smart lampposts used in assisting police to enforce the law in Hong Kong by artificial intelligence.

HKAA wants to integrate this deep learning-based solution into the surveillance system to create a smart airport image and provide more insights into operational efficiency. This also strengthens the image of HKIA as being technological and provides better traveling experiences.

Furthermore, this project also may contribute to the deep learning research area and accelerate deep learning adaptation as this is a real-life application of deep learning which has yet to be actively discovered due to a lack of resources.

1.5 Outline of Report

This report is arranged into four chapters.

Chapter one covers the big data intelligence in HKIA, the motivation and benefits of population density counting, and the proposed approach; it also mentions the project objectives and their significance.

Chapter two explains the project methodology. It outlines the deep learning problem in terms of crowd counting, and machine learning workflow. Also, software implementation and application design will be covered. Lastly, the data visualization strategy will be explained as it is vital for HKAA in optimizing operational efficiency.

Chapter three shows the result of different experiments, such as the survey of state-of-art models on density map estimation, the report of DeepStream SDK Queueing throughput accuracy ,and the analysis of human size inside a frame. Moreover, a scalable and robust system architecture will be discussed. Lastly, it also mentions the future plan and the challenges that we have encountered.

Chapter four summarizes the overall report. It describes the major accomplishments and the future projection of the project. Regarding major achievements, a testing prototype, which contains web application and deep learning application, is developed to show the strengths and weaknesses of this project with the use of DeepStream SDK, Redis, React frontend framework.

2. Methodology

2.1 Introduction

To successfully implement the overall monitoring tool built by big data technology, a detailed breakdown is critical. This project involved three parts, which are the deep learning problem, machine learning operational strategy (MLOps), and the software implementation detail.

This chapter is divided into three sections. The first section is the deep learning problem and potential solution. The second section focus on MLOps and deployment details. The last section describes the visualization techniques and the implementation details.

2.2 Human Localization Problem

Human localization can be performed with a deep learning model. Most of the method is based on CNN, which served as a features extractor. Bounding box regression is the traditional method to achieve localization and density map generation is the more advanced approach to counting smaller human heads.

2.2.1 Convolutional neural networks

Convolutional neural networks (CNNs) are designed to solve image classification and object localization problems.

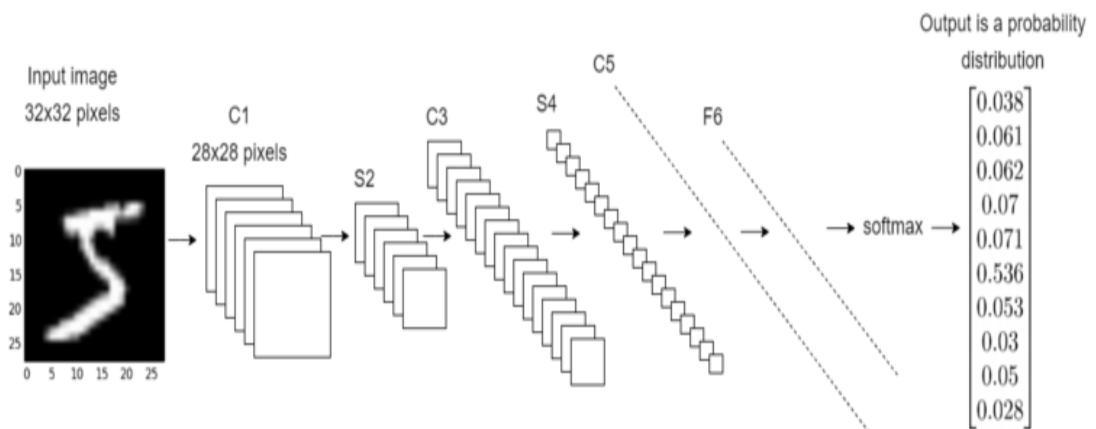


Figure 2.1 Neural Network Structure of CNNs [5]

According to Figure 2.1, The architecture of CNNs is a multi-layer feed-forward neural network that consists of a sequence of convolutional layers and pooling layers. The convolutional layers mirror the structure of the human visual cortex, which consists of multiple layers to distinguish incoming images and identify the object [6]. The pooling layer aims to down-sample the image that helps the convolutional layers to learn the hidden pattern. Before CNNs output the prediction result, the SoftMax layer converts the result from previous to 0 - 1 by fitting the data into a probability distribution.

2.2.2 Bounding Box Regression

Bounding box regression is the most traditional method of object detection in the deep learning field. In the bounding box regression problem, the input is the images that contain objects that need to be detected and the output is the array of bounding boxes. As CNN can classify the images that output the class and the confidence level, which is not variable. The problem is how to achieve output that has variable size.

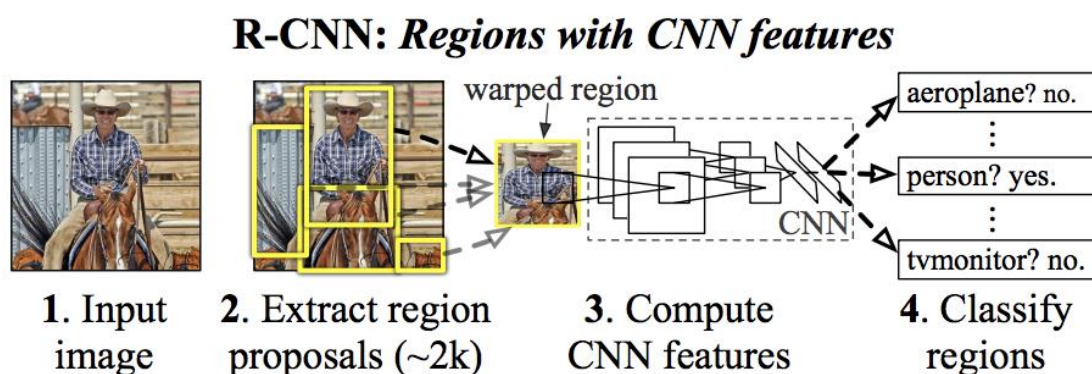


Figure 2.2 Overview of R-CNN [7]

To tackle the above problem, Ross Girshick et al suggested the selective search algorithm, that aims to extract 2000 regions from the images as the region proposal [8]. Instead of attempting to categorize a large number of areas, the model only needs to deal with 2000 regions. This approach may not work with crowded images.

2.2.3 Density Map Generation

To estimate the number in the highly congested scenario, adaptive density map generation is the solution.

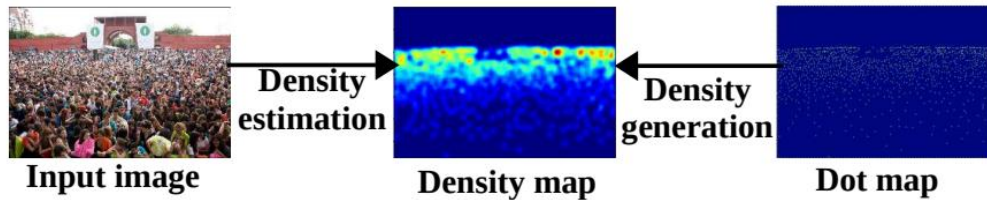


Figure 2.3 Difference between density destination and density generation [9]

In Figure 2.3, given an input picture, a model is trained to predict the density map, which is then summed to yield the projected count. This process is called density estimation. On the other hand, density generation means that the ground-truth crowd density map is generated from the ground-truth dot annotations of people.

Density generation and density estimation are critical to build a deep learning-based model, which follows the encoder-decoder architecture. For density generation, it is used to generate a dataset that maps between the input image and density. As the dataset will be learned by the deep learning model, the quality of generation has a high correlation with the performance of the deep learning model.

2.3 MLOps

The bounding box regression problem and the density map generated both can be solved with the deep neural network powered by deep learning. Deep learning is a supervised learning method that requires a mapping between input and output to learning the existing features. The deep neural network learns to extract the latent feature inside the frame and optimize itself in the model training step. A robust deep learning deployment framework is needed to monitor the model' status and accuracy. Three vital steps are needed: 1. Data Management 2. Model Learning 3. Model Deployment [10].

2.3.1 Data Management

In terms of model performance, data quality and the model network architecture are the two main prerequisites. Regarding data, data is the learning material for the deep learning model therefore data management and data analysis are required before model training. This ensures that the data is not

corrupted, and the data contains the features and embedding that can be learned by the deep neural network. The embedding refers to the correlation between high dimensional vectors and the low-dimensional vector, which simply means that there exists a mapping between input and prediction. Below are the two steps in data management.

Data Preprocessing	Data Analysis
Data preprocessing aims to create the mapping from input to output. In our project, the frame from the CCTV is the input and the bounding boxes or density map is the output. Image transformation is needed to standardize the frame from different camera	Data analysis aims to confirm the relationship between the input and output. Also, it makes sure the dataset is not corrupted or has the wrong file format. This step also double-checks the dataset before model training.

Table 2.1 Description of data preprocessing and data analysis

2.3.2 Model Learning

Model Learning is a critical technique in deep learning for computer neural networks to capture hidden characteristics inside a photograph. Latent characteristics are those that people cannot see directly but can be retrieved using computer techniques. A smart model can extract critical details from training data to 'understand' the link between input and output. Furthermore, the model should have domain adaptations, which means that it can still discriminate between unknown datasets. Model selection and hyperparameter selection is the critical steps in model learning.

2.3.2.1 Model Selection

In terms of Model Selection, this stage aims to find the best model to replicate the one-to-one mapping inside the training set. There are a large number of state-of-art deep learning models available. In this project, we will be surveying some famous state-of-art modes, including but not limited to VGG, RestNet32, RestNet50 ,and PeopleNet. the training accuracy and validation accuracy in a statistical manner will be evaluated.

2.3.2.2 Hyperparameter Selection

The term "hyperparameter selection" refers to the process of picking the parameters associated with the training stage. The hyperparameter influences the efficacy of the training process as well as the model's correctness. Because deep learning is still a black box method, there is no standard technique for tweaking hyperparameters rather than testing. For example, with a varied learning rate, the model's

accuracy in spotting humans varies. In optimizations, the learning rate is a hyperparameter that allows the model to cover local minima. The selection of hyperparameters is essential and aids in the development of a deep learning model.

2.3.3 Model Deployment

The underlying reason for having an MLOps cycle is to create a production-grade model deployment platform that served the well-trained deep learning model, which can accurately predict without downtime. Before selecting the most appropriate pipeline and infrastructure, understanding the business requirement and the working environment is vital

Regarding the business requirement, the most important measure is the cost. In HKIA, there are a gigantic amount of surveillance cameras that monitor the whole airport. Predicting the frame inside CCTV requires an enormous amount of infrastructure to distribute the workload and labor force to maintain the system. Therefore, our solution needs to be lightweight that can operate at a low cost. The second consideration is adaptability, the airport is a highly regulated environment that is subject to changes rapidly. Our solution needs to be isolated from the current system without affecting the current CCTV system and subject to business changes easily.

2.3.3.1 DeepStream SDK

NVIDIA's DeepStream SDK provides a comprehensive streaming analytics toolbox for AI-based multi-sensor processing, video, audio, and picture recognition. It encapsulated some low-level implementation details and provided a fast-processing video analytics pipeline. Deepstream SDK provides easy integration to another system without affection also it provided a certain number of plugins that enabled inferences and tacking more easily. This reduced the integration time and the development process. Using this SDK also ensure flexibility and scalability as inferencing with batching is provided and it is proven that it can handle numerous real-time stream by a different protocol.

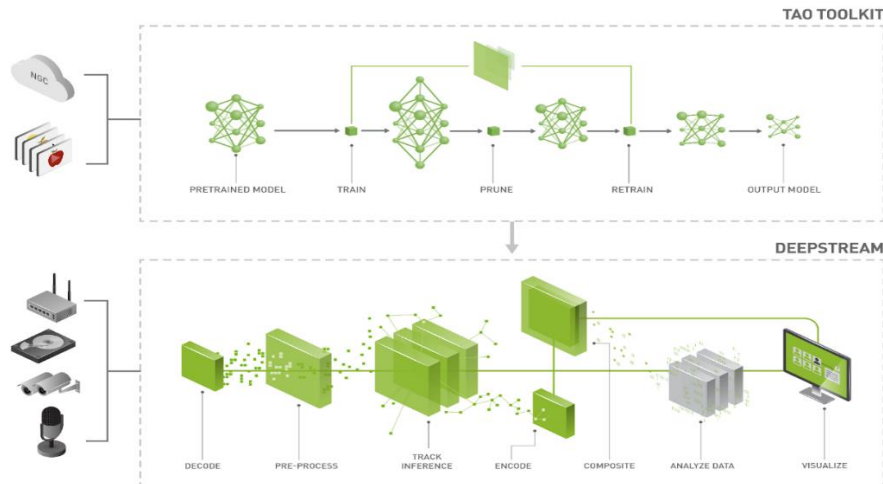


Figure 2.4 DeepStream Brief Summary

In Deepstream SDK, there are two critical components which are Tao Toolkit and the Deepstream app.

Tao Toolkit aims to accelerate the development of AI models, which provide model flexibility, apply transfer learning techniques and optimize the model for faster inference. The Tao toolkit also provides model retraining so that model can be updated easily to adapt.

Deepstream app is developed based on GstStreams, which is a popular open-source video framework. It can easily config to receive an enormous amount of video sources and perform inference. Also, the model deployment is handled automatically with optimization options. Moreover, Deepstream also provides an analytic module that provides features like region-of-interest filtering, and counter threshold alert.

2.3.4 Processing

With Deepstream SDK, the Deepstream app first retrieves the information from the RTSP stream and the hardware perform decoding on the video. Then Deepstream divides the video into an image and preprocesses the image for inferences, such as image interpolation, transformation, or padding. During the inference phase, the images are batched dynamically and perform inferences by the gstreamer plugin. This can further minimize the network overhead and improve the inference throughput. After getting the prediction, clustering and tracking algorithm are applied to find the most appropriate bounding boxes and track bounding boxes respectively.

2.3.5 Additional Features

The first of the SDK's extra features that might be relevant in our situation is hardware buffer sharing. This implies that if we utilize the offered plugins, there is no copying of data in memory throughout the whole process of input to output; instead, just memory references are passed, allowing for very high efficiency and minimizing the computational cost of adding new plugins or capabilities.

Second, the SDK can be simply deployed in a container and managed using K8s, the industry-standard scalable deployment tool. The third method is to update the model through the internet. This implies that the inference model can be changed on the hardware with minimal downtime, giving us a lot of flexibility in the future when we want to try out alternative models for different regions without having to reprovision all the gear.. There is also a lot of leeway in the kind of models that can be operated using this SDK. The SDK may also host models currently owned by Nvidia, such as the PeopleNet, or models with predefined architecture but varied parameters, or even totally bespoke models. If any criteria or capabilities are required, this architecture may still be used by changing a model and a few variables.

2.4 Software Implementation

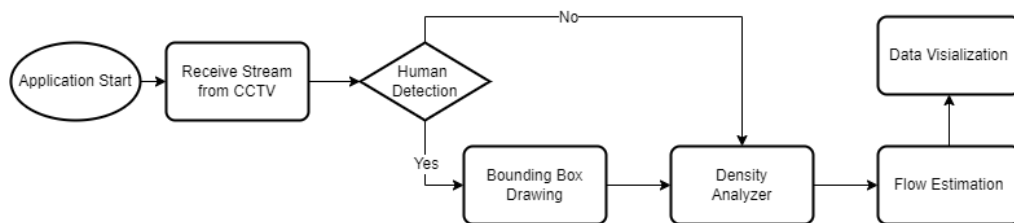


Figure 2.5 Application Flow Chart

In Figure 2.5, our application firstly receives a stream from CCTV and sends it to the deep learning model for the prediction of bounding boxes, which have four points that cover the whole human. After the computation of the center point of the human, the density analyzer will generate a heatmap that clearly shows the density of the area and visualize it on the web portal. If the area involved a queue as stated by the HKAA, our application will perform flow estimation and calculate the throughput of the queue and visualize the result online.

2.5 Application Design

2.5.1 Frontend

React with Typescript is selected to handle this complex real-life web application. React is ranked second in the top 10 website development frameworks [11], the best open-source web application frame that got 176,000 stars on GitHub and is widely supported by the community. The reason for adopting Typescript is type safety, which means that the variable is not able to change its type after declaration. This brings benefits in the standardization of coding style and allows developers to develop a bug-free application.

2.5.2 Backend

In terms of reliability and speed, Golang is chosen to deliver this low-latency project as Golang is designed to be a parallel computing programming language, offering Communication Sequential Processes (CSP) paradigm internally [12]. The CSP enables convenient parallel processing to facilitate concurrent execution, which means that processes are running simultaneously. Golang backend is developed to handle multiple concurrent streaming requests by the cameras.

2.6 Data visualization

Numerical representation and time series line graph will be used to display generic metadata, for example, the number of people in HKIA and the predicted increase in density. On the other hand, the heatmap is the building block of visualizing density information in specific areas.

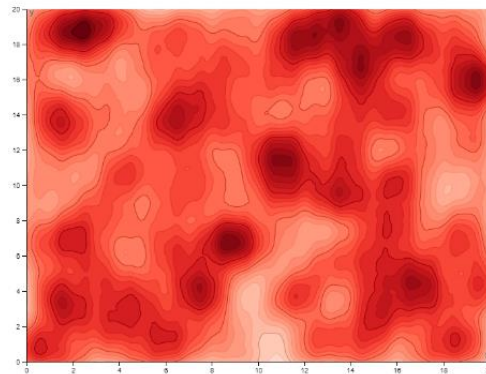


Figure 2.6 Example of Heatmap

A heatmap is a common tool to visualize density by highlighting denser areas and less dense areas. It is commonly used to transform 2d cartesian coordinates into heatmap diagrams. In Figure 2.6, density is shown in a gradient format. The darker the color the higher the density while the lighter the color the lower the density. The contour plot gives HKAA staff a general idea of which region is crowded.

2.6.1 D3.js

D3.js is a JavaScript package that allows you to manipulate pages based on data. Using HTML, SVG, and CSS, D3 allows you to bring data to life. D3's emphasis on web standards offers you complete access to contemporary browser features without tying you to a proprietary framework, combining sophisticated visualization components with a data-driven approach to DOM manipulation.

2.7 Summary

The above section has covered the two deep learning solution to the crowd counting problem. The two solutions are the bounding boxes regression and the density map estimation method. Also, we have introduced the entire system pipeline that is concerned with the machine learning component. The capabilities of DeepStream and their advantages in our system are then thoroughly examined. Lastly, the visualization strategies are explain in details with D3.js.

3. Results and Discussion

3.1 Density Map Generation Model Survey

A survey report on density map creation is undertaken to determine which model is most suited for real-time crowd counts. The models have been evaluated on the HKU CS GPU Farm 2. The GPU farm is outfitted with NVIDIA GeForce RTX 2080 Ti GPU cards and a processor capable of delivering up to 44 Tera-FLOPS FP64 computing power.

Models are trained with just 30 epochs due to computational power and time constraints. Most models will converge to their global minima. In addition, the models are trained using Google's Pytorch framework. Pytorch uses eager mode computing, which means the neural graph is created dynamically, which makes debugging easier.

3.1.1 Dataset Selection

In this report, two popular datasets are used to evaluate the models. There are UCF-QNRF and the ShanghaiTech Part B dataset. The UCF-QNRF is a large crowd counting dataset, that has a very high number of people count and contains a variety of scenes. While the ShanghaiTech Part B dataset, contains images that are taken from the busy streets and the people are less relatively sparse. The images specifications are as fellow.

Dataset	Number of Images	Resolution ($H \times W$)	Count Statistics			
			Total	Min	Ave	Max
UCF-QNRF	1,535	2013 X 2902	1,251,642	49	815	12,865
ShanghaiTech Part B	716	768 X 1024	88,488	9	123	578

Table 3.1 Images specification of the dataset

According to Table 3.1 Images specification of the dataset, the resolution and count statistics differ significantly between UCF-QNRF(QNRF) and ShanghaiTech Part B(SHTB). In terms of the overall population, UCF-QNRF outnumbers ShanghaiTech Part B by 14 times. This suggests that UCF-QNRF recorded more persons in a single photograph, whereas ShanghaiTech Part B had a narrower field of vision.

3.1.2 Data Preprocessing

There are two primary motivations for data preparation. The first reason is to reduce VRAM utilization, like the NVIDIA GeForce RTX 2080 Ti only has 10 Gb of memory. Models and pictures are loaded into VRAM for calculation throughout the training phase. Images are scaled to conserve memory since QNRF has a rather big resolution size. The second purpose is to accommodate a more advanced CNN model. Because certain models require a downsampling layer, the picture resolution must be multiplied by a factor of 16. As a result, the QNRF photos are scaled to 1024 X 1024 while maintaining the aspect ratio, but the size of SHTB remains the same (768 X 1024).

The ground truth density map is developed after data preparation for training purposes. The estimated density map was created for both datasets using the kernel size 15 and the Gaussian kernel function. The purpose is to convert the people's centroid point into a radial basis probability distribution with a total area equal to one.

3.1.3 Training Options

Label Normalization is a strategy for improving model accuracy. According to the C3F Experiment, the Magnification factor is a useful parameter for optimizing the model. Following data preprocessing, a magnification factor is added to the ground truth density map. The formula is as follows:

$$New\ Density\ Map = DensityMap * MF \quad where\ MF\ is\ Z$$

Equation 1 Magnification Factor Equation

Because each model and dataset has its unique distribution, the magnification factor has a large influence on the error rate for the Resnet50 model in the C3F Experiment. If the difference is substantial, the model will find it difficult to converge to the dataset [13].

3.1.4 Evaluation Metrics

This survey adopted two evaluation metrics to validate the models' accuracy. The two metrics are mean absolute error (MAE) and mean squared error (MSE).

The MAE equations are as follow:

$$MAE = \frac{1}{n} \sum_{i=1}^n |f_i - y_i|$$

The MSE Equation is as follow:

$$MSE = \sum_{i=1}^n (f_i - y_i)^2$$

Where f_i is the predicted value, y_i is the ground truth.

3.1.5 Model Architecture

There are five models to be examined in this survey. VGG, ResNet50, ResNet101, MCNN, and AlexNet are among them. They are all popular state-of-the-art CNN models with good feature extraction capabilities and outstanding results in many competitions.

3.1.5.1 VGG16

VGG is the abbreviation for Oxford University's Visual Geometry Group [14], and 16 indicates that the neural network has 16 layers, 13 of which are convolutional layers and three of which are fully linked layers. The pretrained model has been trained on millions of ImageNet pictures with 1,000 classes, which covers practically everything humans see. VGG16 is a general image recognition solution that achieves 92 percent accuracy.

The first ten convolutional layers of VGG are retrieved and used as the encoder in our survey. The decoder consists of simply two convolution layers. The following is a graph of a neural network:

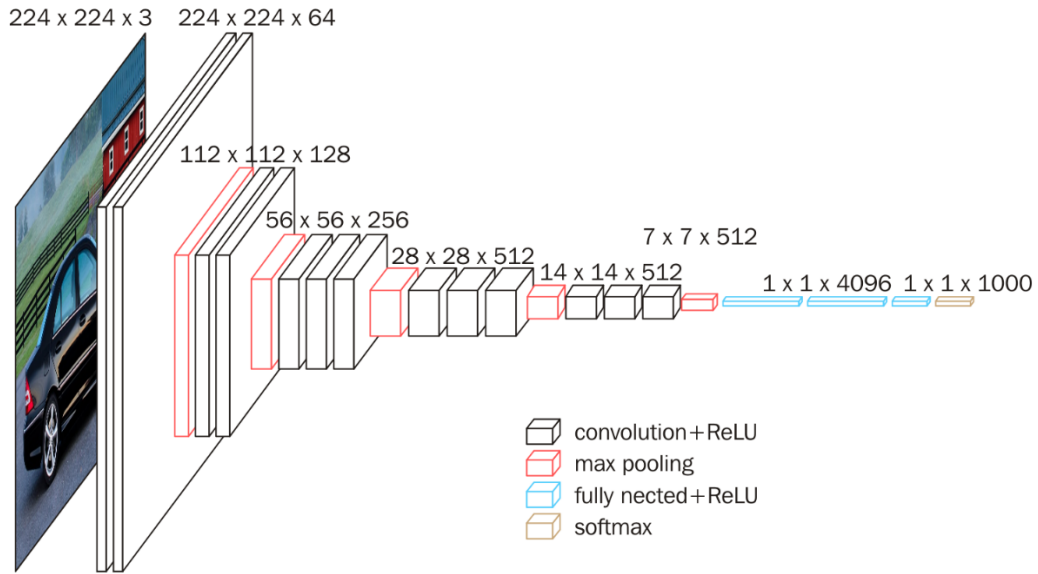


Figure 3.1 Network Architecture of modified VGG [15]

3.1.5.2 ResNet50/101

ResNet is a residual neural network [7] made out of pyramidal cells from the cerebral cortex. ResNet proposes the skip connections strategy to overcome the problem of vanishing and bursting gradients in VGG models. This technique skips the training from a few layers and straight leads to the output. In the picture recognition challenge, the pretrained ResNet model scored 96.7 percent [7]. The number following ResNet denotes the number of layers within the neural network.

In our study, the encoder is the ResNet Model, which changed the stride parameter on the layer 3 convolutional layers from 2 to 1. The decoder is like VGG in that it has two convolution layers.

ResNet50's neural network graph is shown below:

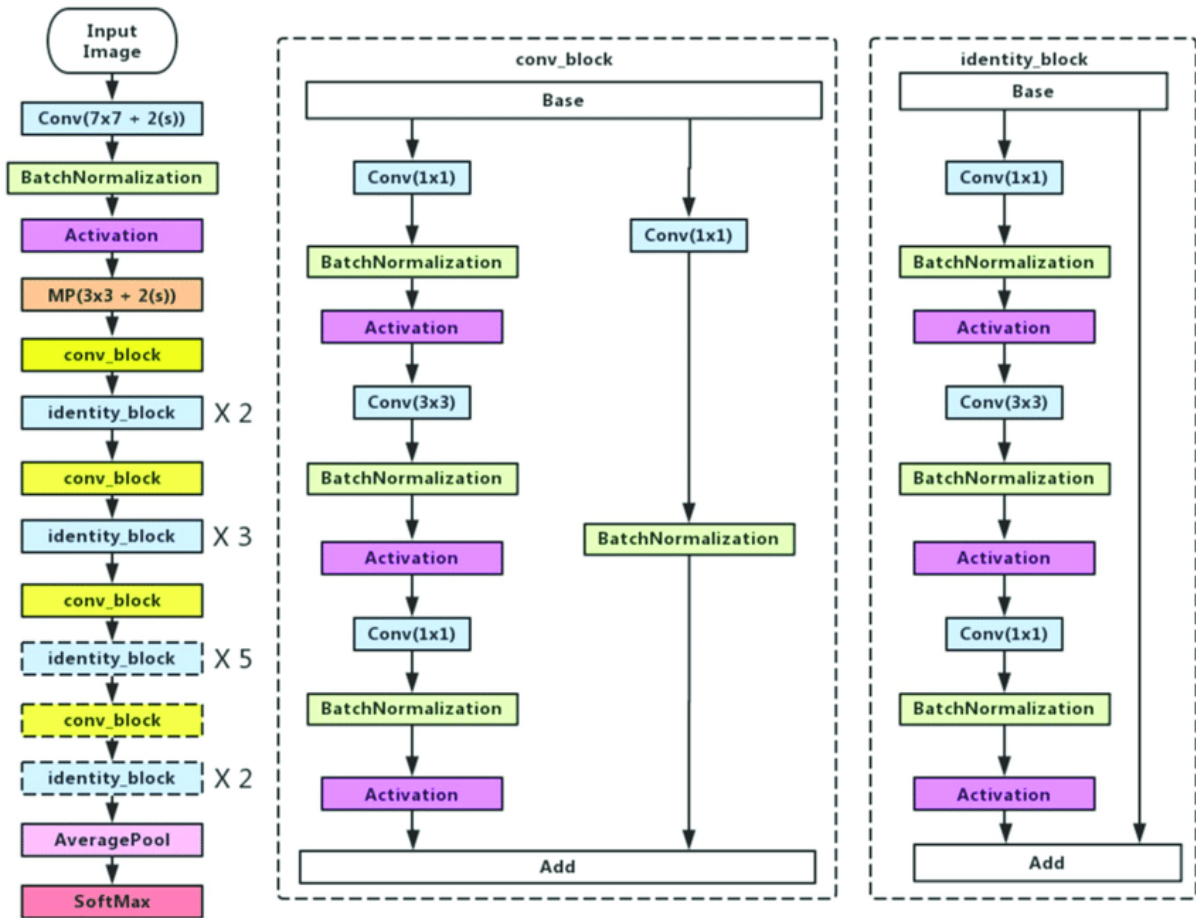


Figure 3.2 Network Architecture of modified ResNet50 [16]

3.1.5.3 MCNN

The Multi-Column Convolutional Neural Network (MCNN) [17] is a deep learning model that excels at crowd counting. MCNN can take input pictures of any size and compute the estimated density by employing multiple CNN with distinct receptive fields of varying sizes. As a result, each CNN inside MCNN captures various elements within the input photos. Finally, the model stacks each feature's output.

MCNN lacks encoder-decoder structures and can only calculate by merging the output of each CNN.

The following is a graph of a neural network:

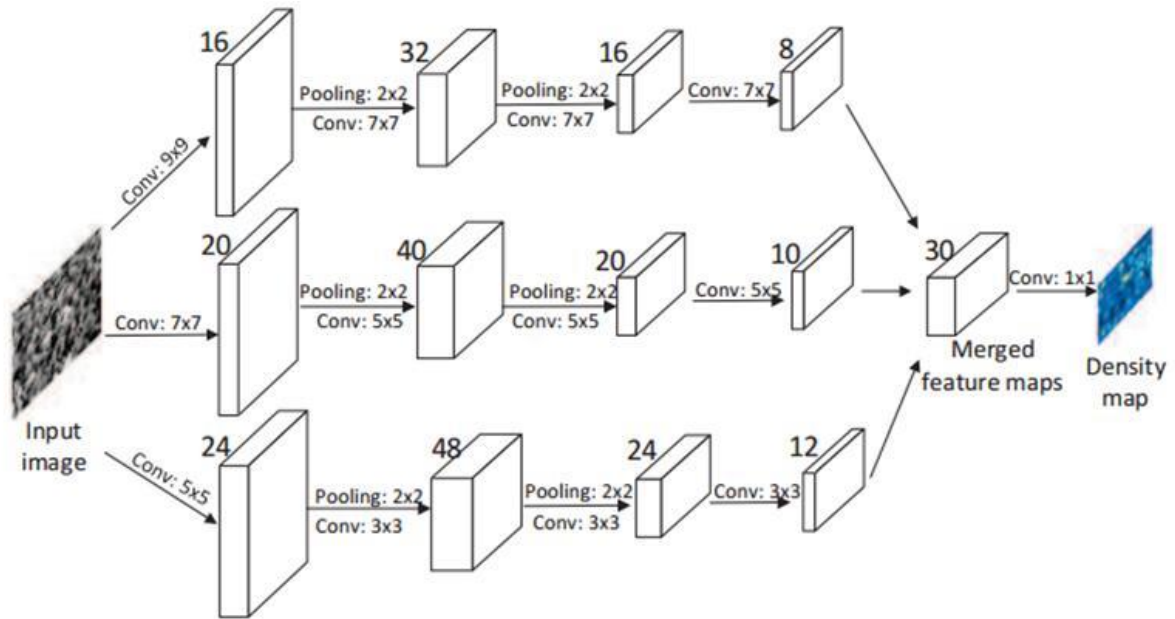


Figure 3.3 Network Architecture of modified MCNN [17]

3.1.5.4 AlexNet

AlexNet [18] is a well-known historical CNN model created by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. AlexNet performed admirably in the ImageNet Large Scale Visual Recognition Challenge in 2012. AlexNet is the best model, with a top-5 error of 15.3 percent higher than the first runner-up.

AlexNet served as the encoder in our survey. To ensure that the feature map may be separated, the padding parameter of the convolutional layers is modified. The encoder employs only the first five convolutional layers, as well as a downsampling layer. The decoder is comparable to the VGG decoder.

The following is a graph of a neural network:

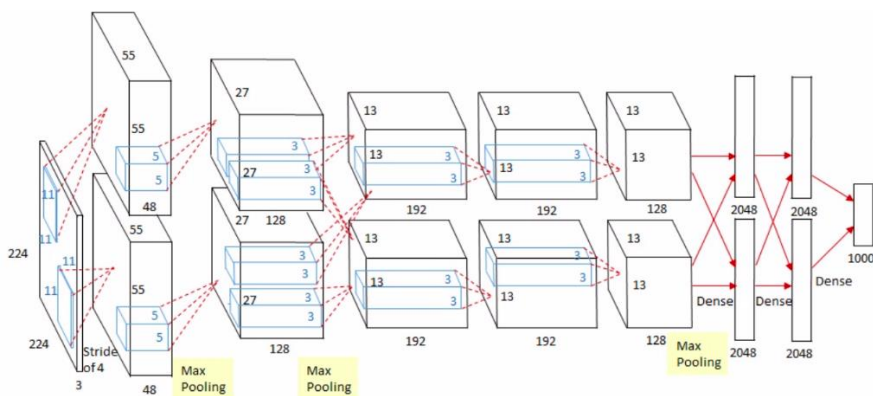
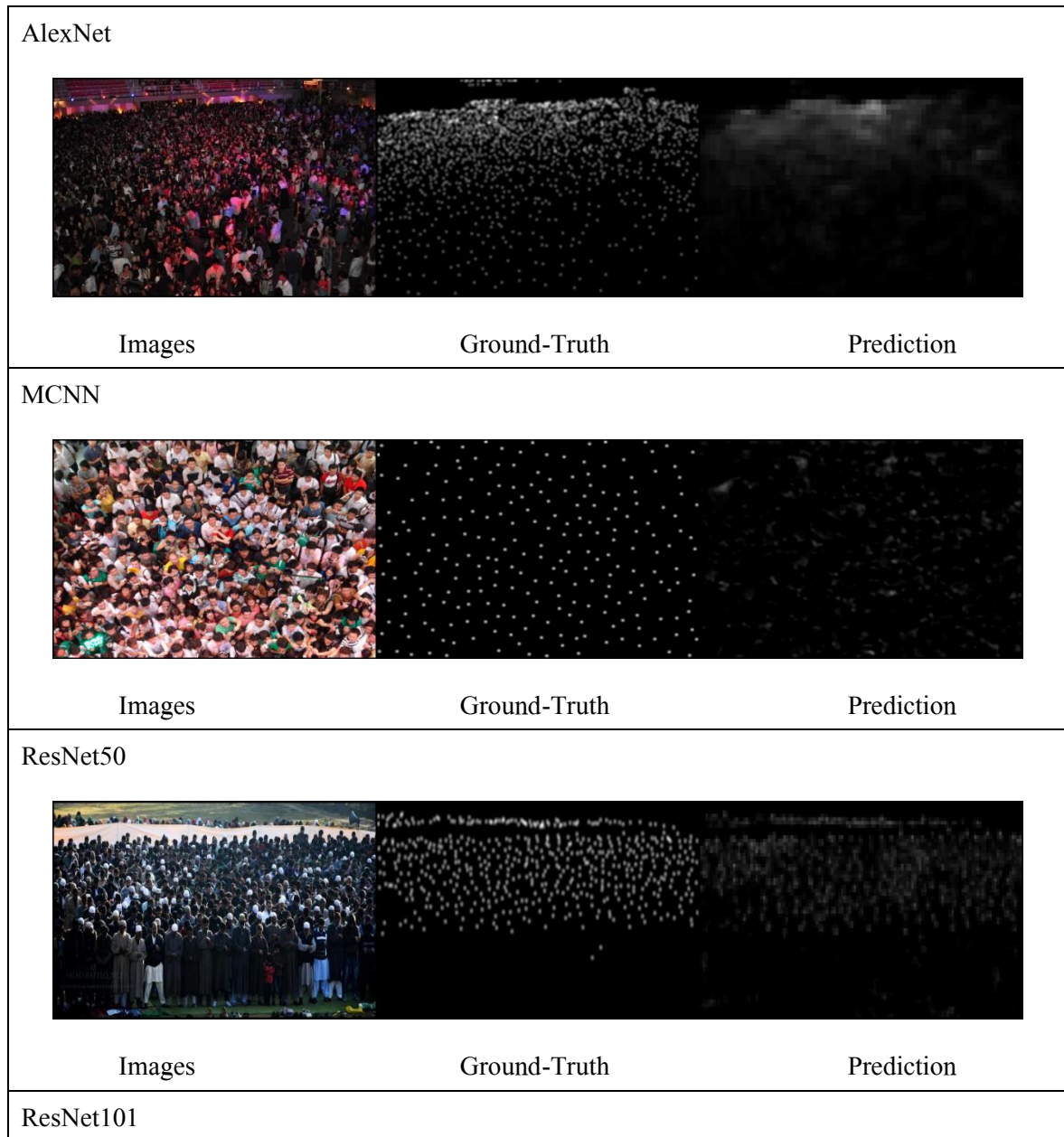


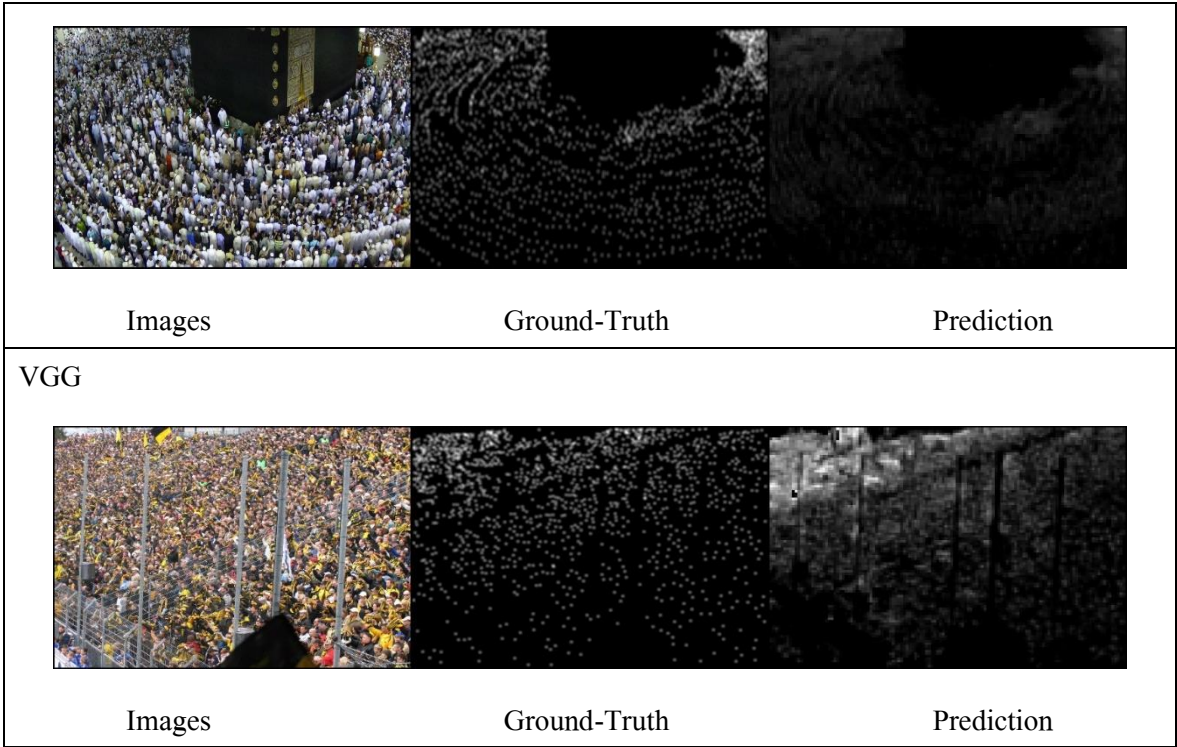
Figure 3.4 Network Architecture of modified AlexNet [18]

3.1.6 Result

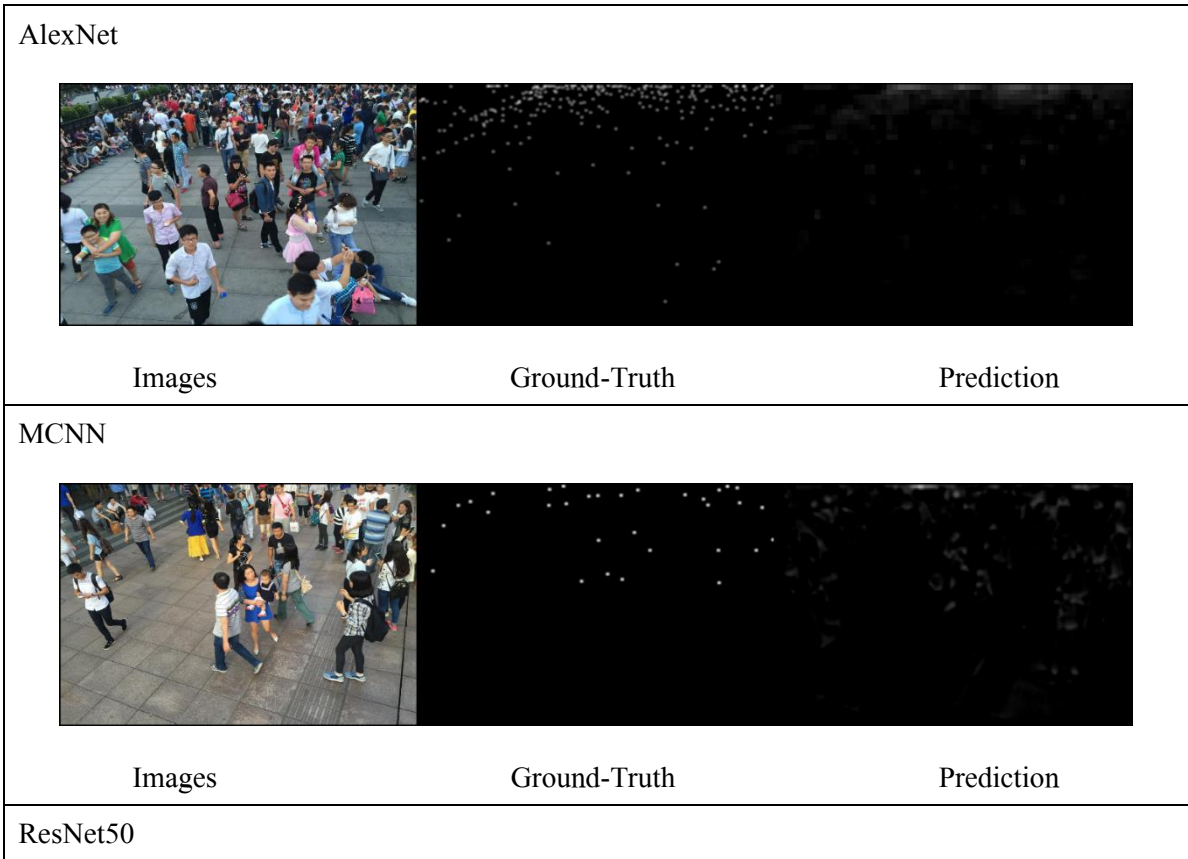
The outcome is divided into two sections. The first section is devoted to visualization, and the second section is the model evaluation findings.

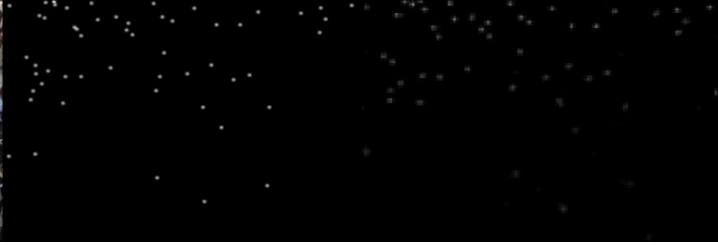
3.1.6.1 QNRF Dataset Visualization





1.1.1.1. SHTB Dataset Visualization



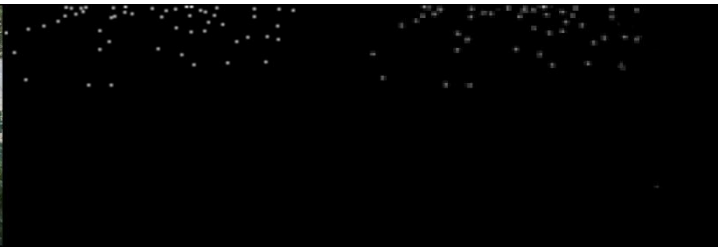


Images

Ground-Truth

Prediction

ResNet101



Images

Ground-Truth

Prediction

VGG



Images

Ground-Truth

Prediction

3.1.7 Model Evaluation Results

The goal of the study is to determine which model performs the best in crowd counting in terms of MAE and MSE measures. The SHTB outcome is as follows:

Shanghai Tech Part B Result

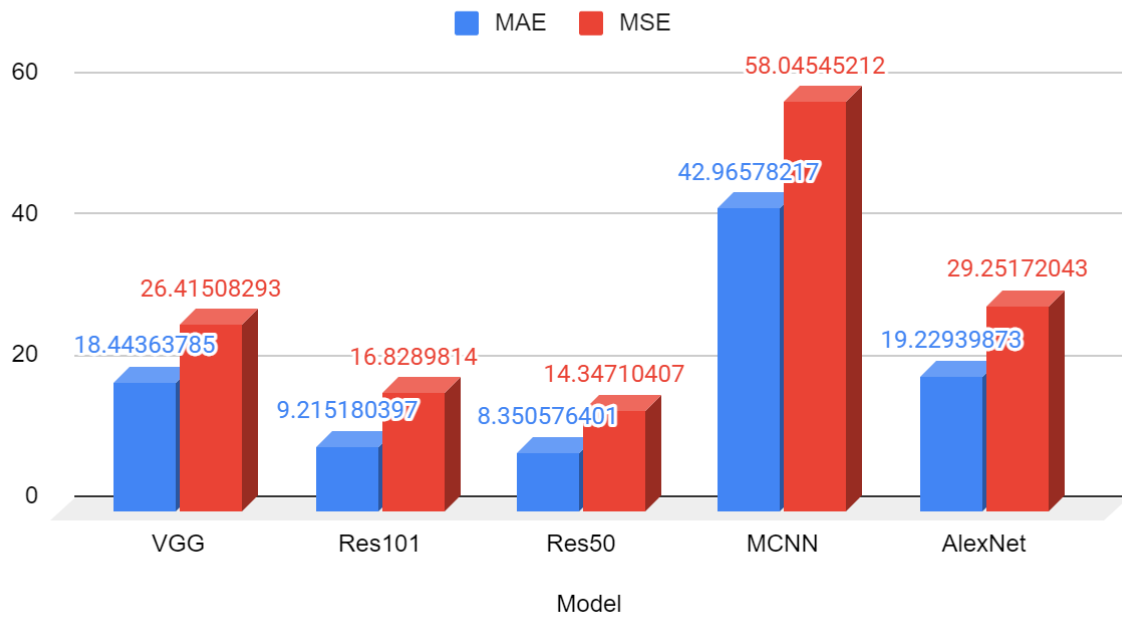


Figure 3.5 Result of Classification model for the Shanghai Tech Part B Dataset

The best model in the SHTB results is the ResNet50, with an MAE of 8.35 and an MSE of 14.347. The ResNet101 is the second model. The MAE and MSE are 9.21 and 16.82, respectively. The MCNN is the poorest model, with 42.96 MAE and 58.04 MSE. Overall, the ResNet family model excels in feature extraction. Furthermore, the ResNet101 model has not yet been well-trained for 30 epochs. It may produce better results if more epochs are used.

UCF-QNRF Result

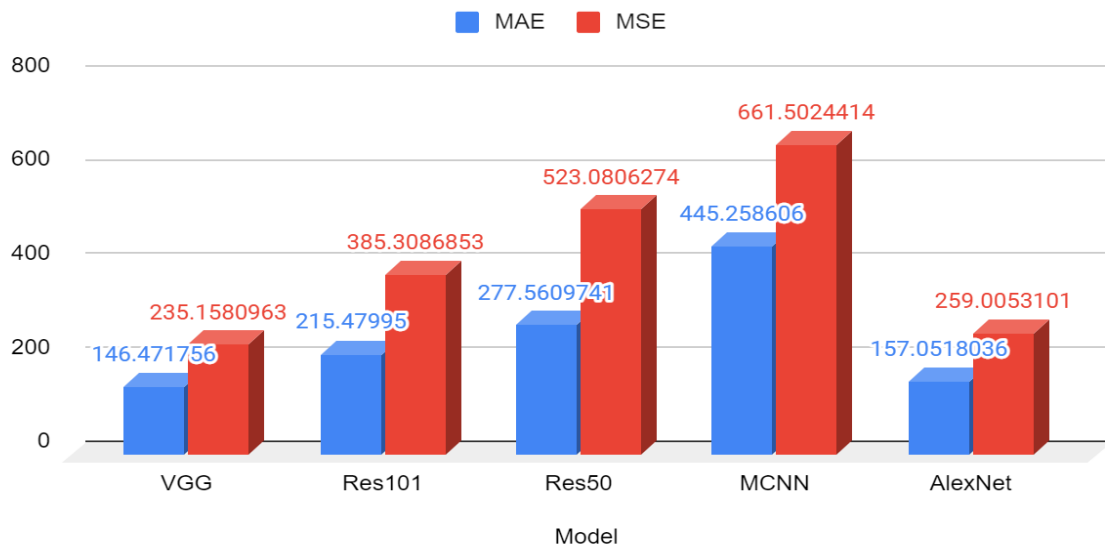


Figure 3.6 Result of Classification model for the UCF-QNRF Dataset

The best model in the QNRF results is the VGG, which has 150 MAE and 235 MSE. The ResNet101 and ResNet50 are the second and third models. The poorest model is MCNN, which has an MAE of 400 and an MSE of 660. Because of the large number of persons in QNRF, all of the models have significant MAE and MSE. As there are many more characteristics to learn from each image, more epochs may be required to validate the outcome.

3.1.8 Summary

Density generation is used to alter the encoder-decoder architecture in this experiment. The model's accuracy can be improved by increasing the magnification factor. In terms of assessment results, the ResNet family outperforms the VGG family in the closer view image for the SHTB dataset and the ResNet family in the far view image for the QNRF dataset. As a result, this report verifies the use of Resnet50 in certain cameras that are closer to the crowd and VGG in some cameras that are further away from the crowd. This report may not take into account computing time and latency. At a later stage, model optimization should be used.

3.2 Deepstream Queuing Throughput Accuracy

Height (m)	Horizontal Distance (m)	Distance	File name	Count	Actual	Reason for miscount
25	36	44	drone_1_1_ds.m p4	6	6	
50	50	71	drone_1_2_ds.m p4	8	8	
30	50	58	drone_1_3_ds.m p4	2	2	
25	30	39	drone_2_1_ds.m p4	6	7	Line moved as camera moved
25	30	39	drone_2_2_ds.m p4	5	6	People bunch up together, blocking tracker
20	26	33	drone_2_3_ds.m p4	4	4	
40	25	47	drone_3_1_crop ped.mp4	N/A		Unable to detect or track due to limited resolution/overhead
50	25	56	drone_3_2_crop ped.mp4	N/A		angel
50	85	99	drone_3_3_crop ped.mp4	N/A		Although the camera is still 4K, the image quality is significantly worse

Table 3.2 Deepstream queuing output accuracy

Key findings:

- Despite having the same pixel count, image quality is vital.
- The line shifts as the drone move, making a possible area for improvement
- To avoid as many overlapping persons as feasible, the camera should be positioned 45 degrees to the line of the queue exit.
- if the tracker can identify the person when it crosses the boundary, the movement is recorded.

3.3 Analysis of Human Size inside a Frame

After the analysis of the accuracy of the throughput, the throughput depends on the performance of the detection model and the tracker. The tracker will work well if the detection model can propose a similar bounding box in a different frame, therefore the bottleneck is the performance of the model. The question would be what is the total pixel count to represent a person or how large the human inside the frame is needed to represent a human by the understanding of the detection model.

In our application, PeopleNet model is the detection model, which is based on one NVIDIA DetectNet_v2 detector with ResNet34 as a feature extractor. According to our research, the ResNet model is the best feature extractor and 34 layers are chosen because it performs fair inference time.

3.3.1 Methodology

To find out the total pixel count, a testing platform is built for performing image inferences. The PeopleNet model is deployed on a machine, that is equipped with NVIDIA RTX 3090, Intel i7-8700, and 16 GB RAM.

For the model development pipeline, the pretrained and unpruned PeopleNet is downloaded from NVIDIA NGC, and It is processed with TAO-toolkit to optimize the model and make it able to perform inference with the TensorRT engine. Regarding the PeopleNet, the model accepts 960x544x3 dimension input tensors and outputs 60x34x12 bounding box coordinate tensor and 60x34x3 class confidence tensor. After the inferences, the NMS clustering algorithm is used to propose the bounding box. The threshold confidence is set to 0.6, which is the most optimal under hyperparameter tuning.

In terms of the experiment, the screenshot is extracted from a video of the vaccination center in Hong Kong. The default frame size is 3840 x 2160x3. Firstly, the frame is resized to 960x544x3, and then scale down the image by 5% to create a total of 20 images. To be able to perform inferences, the scaled-down image is padded with white tile. The reason of padding is to make the people's pixel count as small as possible so that this experiment can find out what is the minimum pixel count is needed to represent humans.

After the data collection, data and statistical analysis is used to understand the relationship.

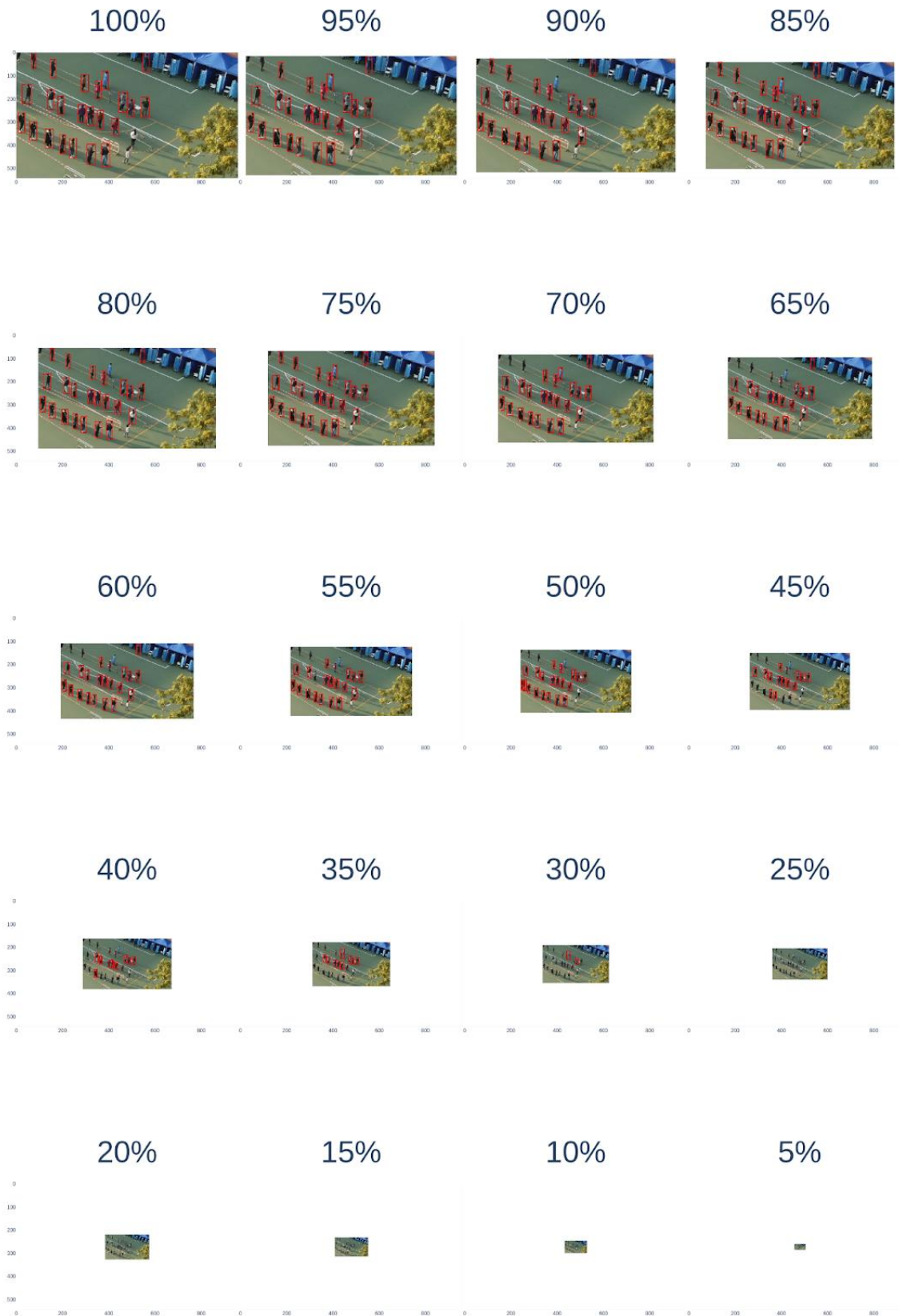


Figure 3.7 Gallery of Inferred Images

3.3.2 Result

This experiment aims to find out the smallest pixel count to represent humans, and the correlation between the scaling factor, the number of bounding boxes, and the total pixel count of the smallest bounding box inside a frame. The Scaling factor is the percentage of scaling down of the original image, which 1 means the original image.

Scaling factor	Number of BBox	Pixel count of smallest BBOX	Scaling factor	Number of BBox	Pixel count of smallest BBOX
0.05	0	0	0.55	21	560
0.1	0	0	0.6	21	645
0.15	0	0	0.65	19	810
0.2	0	0	0.7	26	864
0.25	0	0	0.75	25	795
0.3	2	275	0.8	24	880
0.35	9	297	0.85	25	1062
0.4	10	372	0.9	24	1152
0.45	13	408	0.95	25	1533
0.5	22	408	1	24	1420

Table 3.3 Result of the Analysis of human size inside a scaled down Image

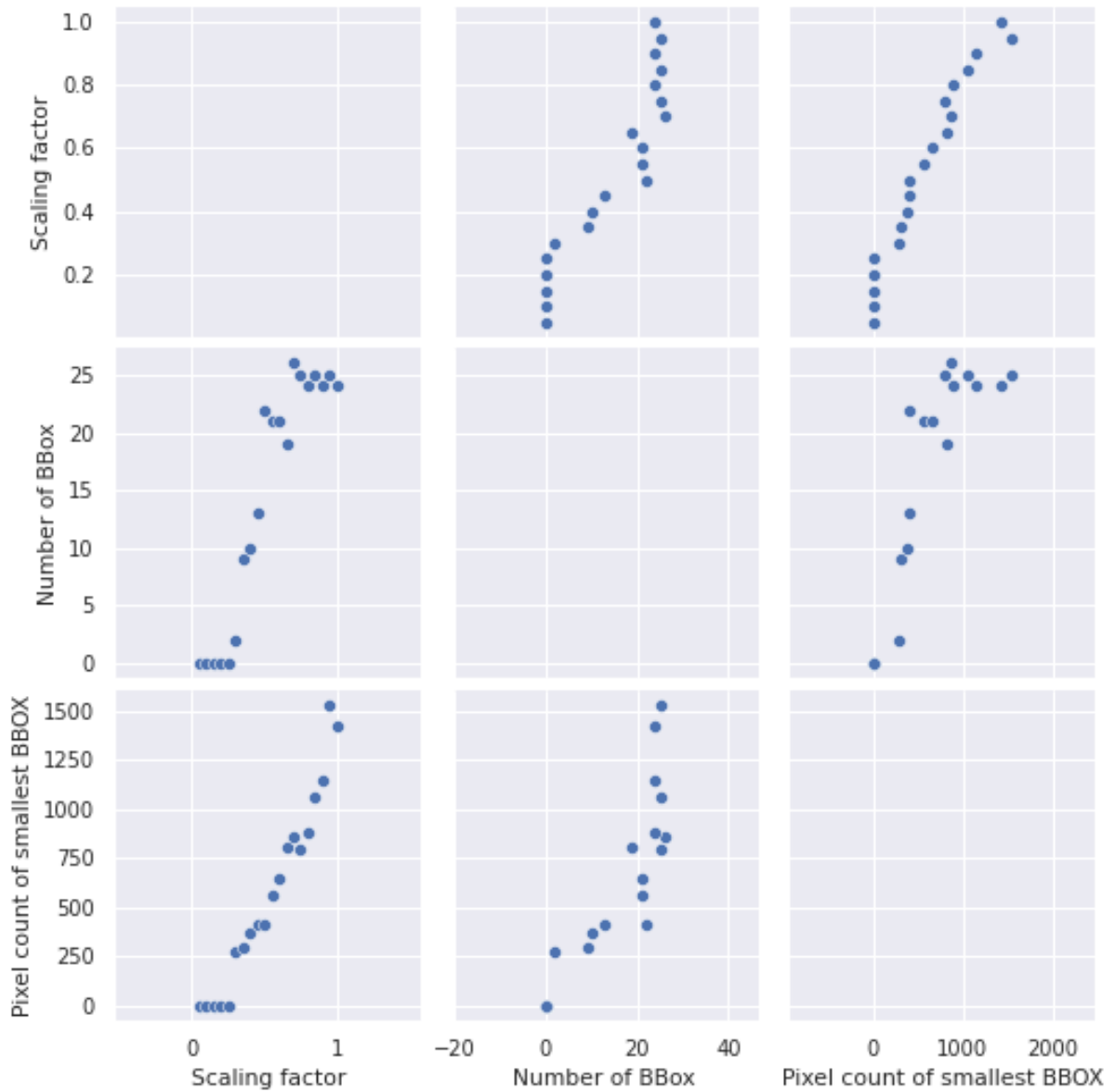


Figure 3.8 Pairplot of the Result of the Analysis of human size inside a scaled down Image

In Table 3.3 and Figure 3.8, there exists a positive relation between the scaling factor and number of bounding box, and between the scaling factor and the total pixel count of the smallest Bounding box. The smallest bounding box size is 275, which has a rectangular shape of 11x25, while the largest bounding box size is 1420, which has 20x71. Also, the smallest bounding box exists in the 0.3 scaled image and the largest bounding box exists in the 1 scaled image. This clearly verifies that the larger the human, the more humans can be found by the PeopleNet model.

Scaling factor vs Smallest Bounding Box Pixel Count

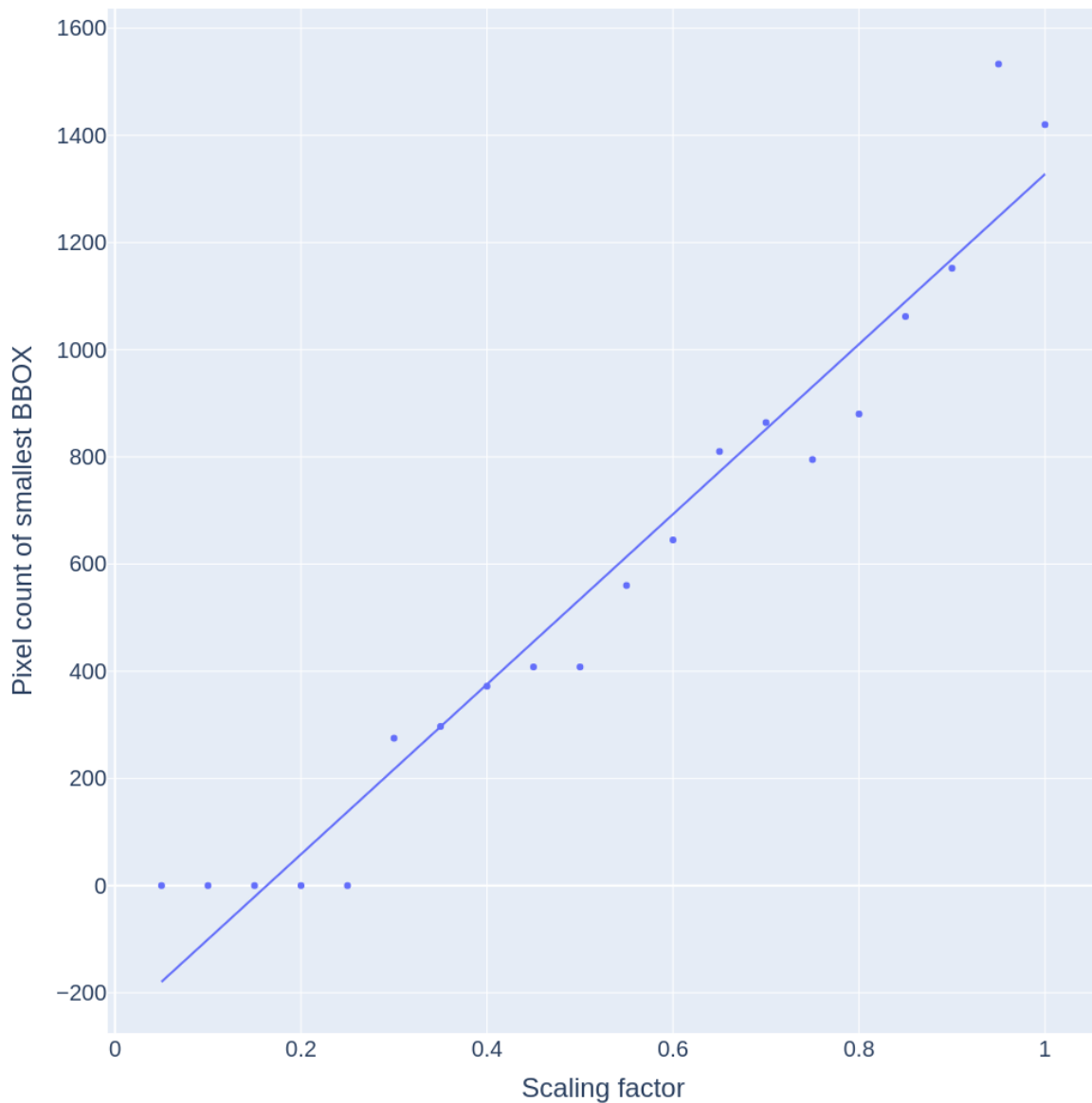


Figure 3.9 Scatter plot between scaling factor and the pixel count of smallest bounding box

In Figure 3.9, the slope between the scaling factor and the total pixel count of the smallest bound box is positive. The regression line is drawn with ordinary least squares and the R2 score is 0.95, which means that the two variables are strongly correlated. This is easy to understand as the detection model always outputs the bounding box with high confidence and filters the bounding box that has low confidence. The smaller the human, the lower the confidence level of the bounding box. Therefore the 0-0.25 scaled image has no output, as the model struggles to determine if it is a human or not.

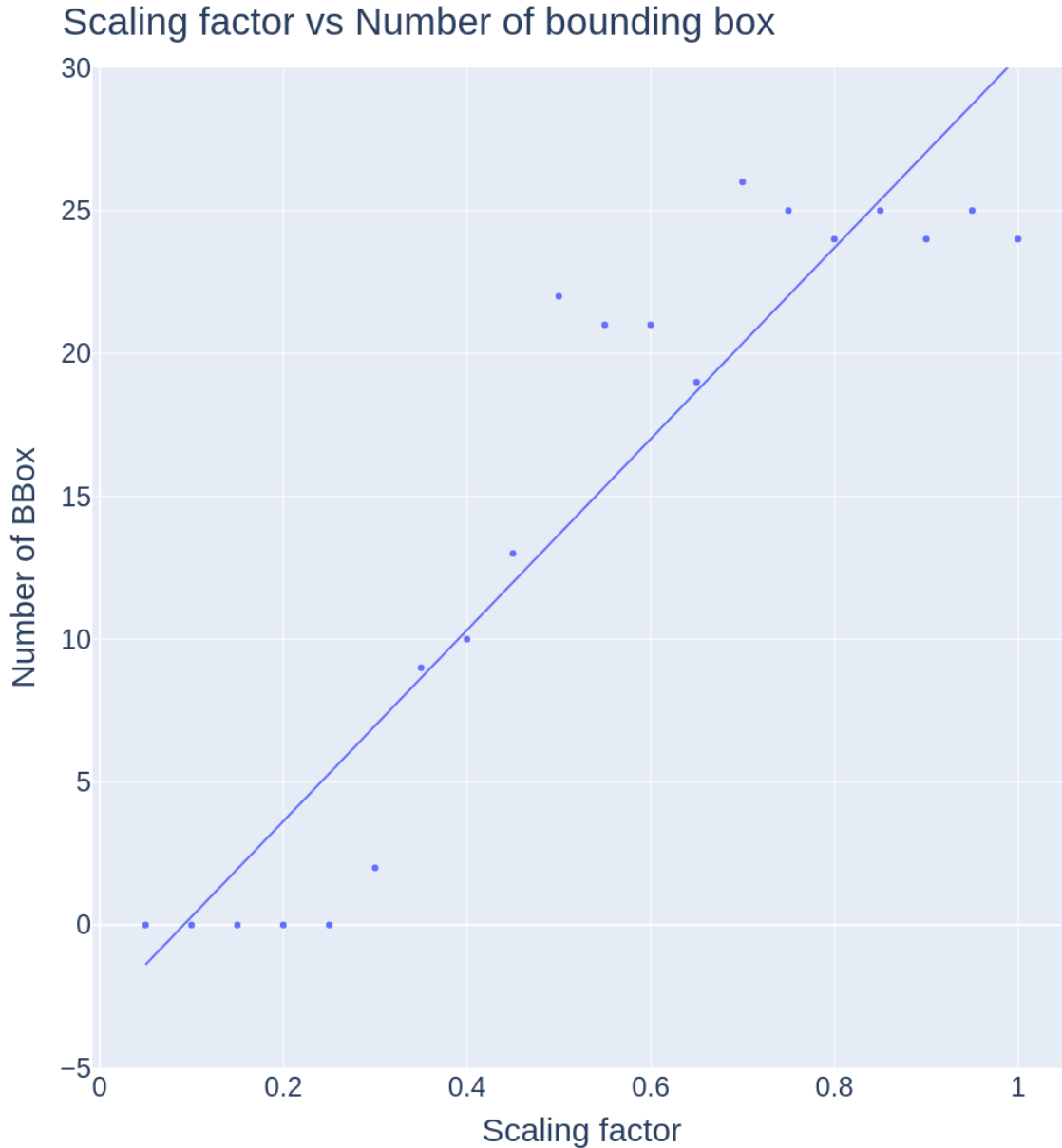


Figure 3.10 Scatter plot between scaling factor and the number of bounding boxes

In Figure 3.10, the scaling factor and the number of bounding boxes have a positive relationship. The regression line is drawn with ordinary least squares and the R2 score is 0.86. This can easily be explained as the detection model can propose more bounding boxes with high confidence when the image is larger. Therefore most of the bounding boxes will not be filtered by the clustering algorithm. When the image is smaller, the detection box outputs bounding boxes with lower confidence, and the clustering algorithm will filter boxes with low confidence. In summary, the threshold confidence level in clustering algorithms also is an important metric to be tuned before deployment.

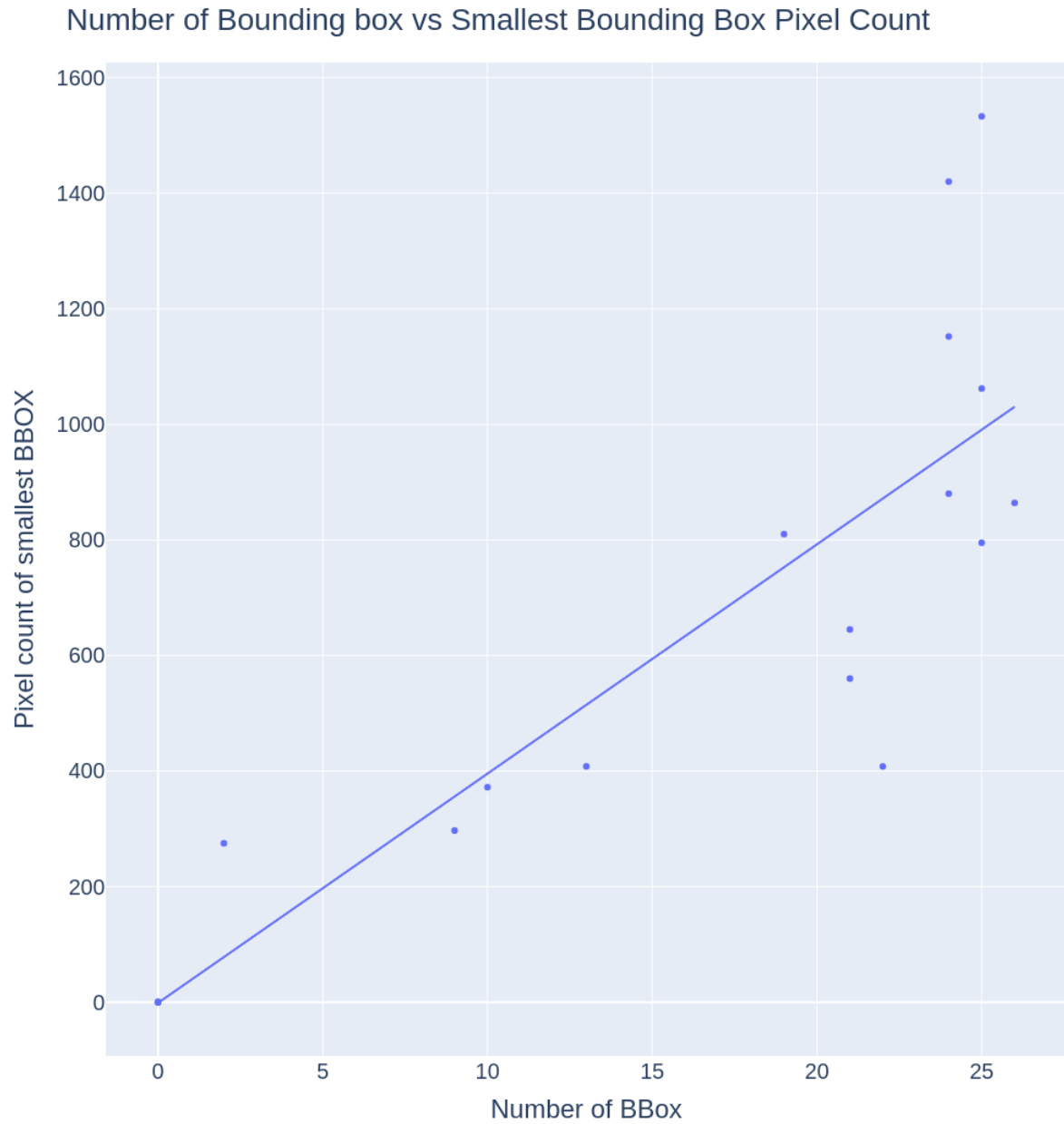


Figure 3.11 Scatter plot between number of bounding boxes and the pixel count of smallest bounding box. In Figure 3.11, the number of bounding boxes and the total pixel count of the smallest bounding have a constructive association. The regression line is drawn with ordinary least squares and the R² score is 0.77, which has a slightly lower score. As these two metrics also show a positive correlation with the scaling factor, there is an indirect mapping between these two metrics, therefore they are correlated.

3.3.3 Summary

This experiment studies the relationship between different-sized images and human bounding boxes. Regarding the size of humans, PeopleNet can detect humans with at least 270-pixel count with the setting of 0.6 as the NMS's threshold in the most optimal case.

With the cropping technique, our application can detect most humans. In terms of correlation studies, the detection model is not the bottleneck, but the bounding box clustering algorithm is. Better clustering algorithms may find smaller humans.

3.4 System Design

The system design diagram in Figure 3.12 Overview of System Design is divided into three parts: MLOps, Backend, and Frontend.

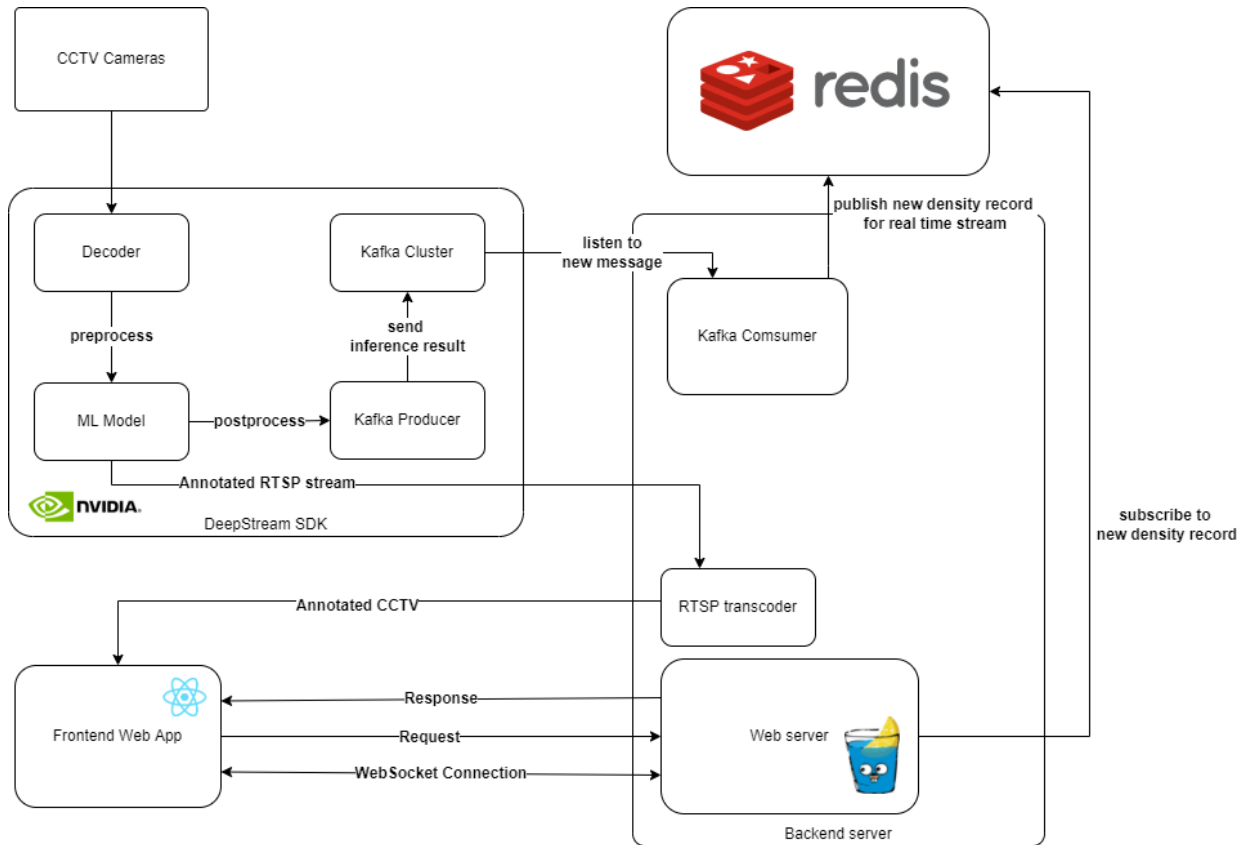


Figure 3.12 Overview of System Design

3.4.1 MLOps

In Our MLOps Cycle, Deepstream SDK is a key component, that decodes the stream, preprocesses the frames, and performs inference and tracking. In our testing pipeline, a static 4k resolution drone footage video file is located on the machine and Deepstream served the video as an RTSP stream, which is decoded in a highly efficient manner with hardware acceleration. Later, Deepstream started to dynamically batch the input frames to increase inference throughput and decrease inferences latency. ResNet10 is the deep learning model for bounding box regression as it offers fast computation time and low latency in inference time. A list of bounding boxes is returned, and Non-maximum Suppression (NMS) clustering is applied to filter some of the incorrect bounding boxes. The clustering algorithm and the inferences protocol are provided as Deep stream plugins.

After clustering, a tracking technique is applied to track humans on different frames and the employed tracking method is called the Kanade-Lucas-Tomasi (KLT) tracker, which manipulates the use of spatial intensity information to map the correct bounding boxes in different inference frames. KLT tracker is a lightweight tracker, that performs tracking well in different scenarios and offers fast computation.

After tracking, Deepstream's nvdanalytics plugin extract the metadata from the inferences and tracking result and applied certain levels of analysis, such as overcrowding detection, and direction detection. This plugin can help to count the number of people inside the queue and throughput of the count and wrap the information into JSON format, which can be easily understood by other systems. Deepstream also delivers this JSON message to the Kafka producer and delivers it to the Kafka cluster. Kafka is a distributed event streaming platform, that provides low latency message queue implementation. Also, the Kafka cluster can be served by the different brokers, that help to deliver messages. This can ensure that the message can be delivered with low latency and high availability. After the messages are delivered to the Kafka cluster, the backend system can initiate a Kafka consumer that subscribes to the topic and start to receive the messages from the cluster. Moreover, Deepstream also will generate an annotated version of the CCTV stream, with a crossing line and the people count in RTSP stream format.

3.4.2 Backend

Backend components include a Kafka consumer, gin web server, RTSP transcoder, and Redis.

3.4.2.1 Kafka Consumer

The Kafka consumer is responsible for reading inference results from the Kafka cluster and it will listen to a topic continuously with low latency.

3.4.2.2 Web Server

The backend system is mainly built with Gin, a high-speed HTTP web framework written in Golang. Gin captures the request from frontend and format the request information to understandable format. After translating the readable data in Golang, business logic is applied and a specific response is sent back to the client side. On the other hand, the web server also use WebSocket to serve the real-time data that generated from the Kafka Consumer in backend.

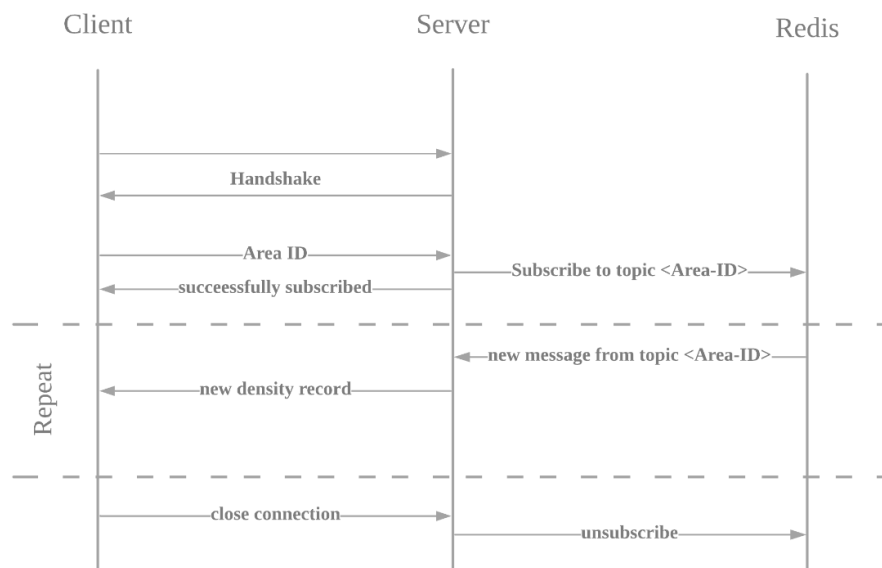


Figure 3.13 WebSocket Sequence Diagram

Figure 3.13 shows how a WebSocket connection is formed and how data is transferred. The client first initializes a handshake request to the server and the server decides to accept or reject the connection. After the server accepts the handshake, the client sends an Area ID to the server to indicate which area of information is needed. Then the server sends a successful response back to the client when the server successfully subscribes to the topic. The area information can be freely transferred from server to client in this stage. When the server receives new information from Redis, it will send it to the frontend which will be updated.

3.4.2.3 RTSP transcoder

As the modern browser doesn't support RTSP streaming, A RTSP transcoder is needed to decompose the video to Moving Picture Experts Group(MPEG), which can be streamed on the client-side browser with WebSocket.

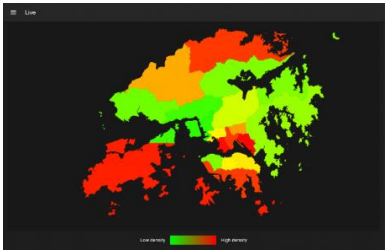
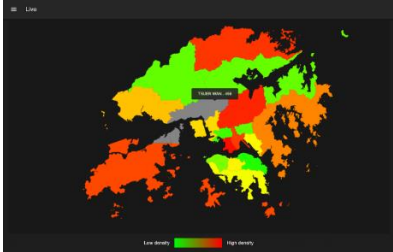
3.4.3 Frontend

Frontend handles most of the data visualization, which is one of our major focuses in this project. A user-friendly and straightforward dashboard is required to display all the information generated by Deepstream with deep learning. An online website is a perfect choice to suit the HKAA use case as the website doesn't have any hardware limitations and it is easily accessible with any device. Also, the website provides ease of adaption, and the website update doesn't require a software update, which lowers the cost of maintenance.

A website can easily update data in real-time through WebSocket, which backend will continuously send the density information to the client-side browser. When the client receives the data, the website will update accordingly in real-time. It also can push some notifications to the browser to alert the HKAA. The functionality of a website is well-suited to the HKAA's user case and enables them to gain business insight easily.

3.4.3.1 Real-time heatmap

In the prototype, heatmap is used to present the comparison of density information. The default map is saved as the GeoJSON, which is a type of format that store the geographic data. HKAA can create the airport map easily with the use of online tools then upload the GeoJSON through the web portal. The map of Hong Kong is used as demonstration.

Normal State	Hover State
 A density heatmap of Hong Kong showing high density areas in red and orange, and low density areas in green and yellow. The map is displayed on a dark background with a small inset map in the top right corner.	 A density heatmap of Hong Kong in a hover state, showing a tooltip over a specific area. The tooltip contains text and a small icon. The heatmap colors are the same as in the normal state.
Figure 3.14 Density heatmap	Figure 3.15 Density heatmap hover state

In Figure 3.14, a Hong Kong density heatmap is shown. The data is passed to the website through WebSocket. The heatmap aims to provide a big picture of density information, that which part of the area is denser and less dense. In the heatmap, red is illustrated as a crowded area, and green is illustrated as a less crowded area.

In Figure 3.15, the heatmap has a hover feature, that shows the number of people in a specific region when the user moves the cursor to a certain location. This can help the user to understand more about the density information for a certain without overloading the user interface.

3.4.3.2 RTSP stream and Entry exit data

To provide ease of access to the Deepstream video output, the output video stream is embedded on the web portal that contains a real-time counter inside the video and a crossing line that is marked as the front of the queue.



Figure 3.16 Screenshot of queuing flow rate estimation application

Figure 3.16 shows the annotated RTSP stream that comes from the Deepstream app on the left-hand side and the real-time update counter from the Kafka message on the right-hand side. The RTSP stream provides a real-time monitor to confirm the deep learning counting model is working since sometimes the CCTV may be blocked by some physical maintenance. The counter provides numeric format information to summarize all density information.

3.5 Future plans

Two main areas require more research in at a later stage. The first area is the people detection model and the tracker model. More analysis should be carried out. Another area is queue identification, which means finding the queue inside the frame instead of manually drawing the queuing area. An extra enhancement is queuing flow rate prediction.

3.5.1 Analysis of detection model and tracker

In HKIA, the CCTVs are located on a high roof and different CCTVs may have a different angle of view. To ensure the detection model is working well in a different situation, the model should have a certain level of generality. Therefore, more studies should be carried out with different independent variables, such as different framerate, image quality, and angle of view. Regarding the framerate, RTSP live stream sometimes may not be that stable depending on internet connectivity, finding the lowest framerate that the application can still detect is critical as it provides insight about whether the data is valid or not. Alert can be sent if the framerate dropped to a certain level.

On the other hand, Image quality is vital for the detection model and RTSP may sometimes have a lost frame issue as it is based on UDP protocol, which does not ensure the delivery of the packet. Studies with different Gaussian noise levels can be executed to analyze the relationship between noise levels and the performance of the detection model. Lastly, images from different angles of view may have different results for inference as training data may have a bias on the front face instead human back. Analysis between angle and performance can ensure that model can detect the different angles of humans.

3.5.2 Queue Identification

In the current setup, HKAA staff are required to manually draw the queue position for each CCTV, and this is a tedious task for HKAA although it is a one-off task. A queue identification model can be developed to streamline the initial setup cost for this application. The model receives a frame containing the queue and the model output bounding boxes that contain the queue. However, this requires a dataset to build the deep learning model and this queue information is lacking online. However, HKAA staff are automatically building a database when they have started to use this application since they are manually drawing queue positions. After a certain number of CCTV queue positions are drawn, the queue identification model can be developed.

3.5.3 Queuing Flow Rate Estimation

In Queuing flow rate estimation, Recurrent Neural Networks (RNNs) are adopted to solve sequential data prediction problems by pattern recognition.

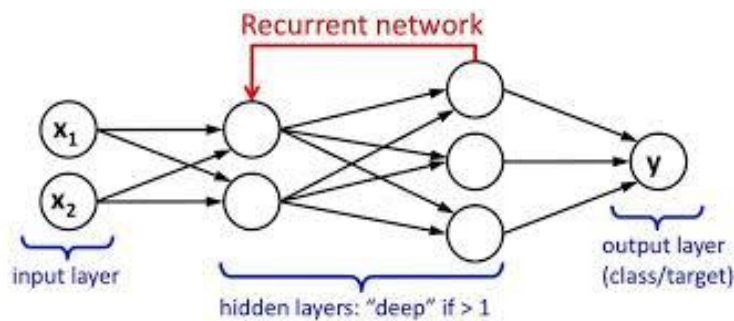


Figure 3.17 Neural Network Structure of RNNs [19]

Figure 3.17 shows RNN is a subfamily of the neural network that allows data temporarily stored inside the network and contains loops illustrated in red. RNN makes judgments based on previous patterns of data to make the future prediction. RNN has a chain-like structure that can handle the sequence of vectors as input and output. RNN is widely adopted in pattern recognition, for example [1,2,3,4] as input, then the RNNs model outputs 5 as the answer.

As simple numerical prediction is not adequate, long short-term memory (LSTM) is an RNN model that is able to learn long-term dependencies and capture the hidden information inside the sequential data. LSTM consist of forget gates, that can forget and remember details selectively [20]. In our project, different kinds of RNNs will be tested with our existing data. By using cross-validation, the most optimal model will be selected

3.6 Challenges and mitigation

In this project, Data collection and model selection have faced a certain level of challenges.

Data collection is vital for model training. As this project focus on HKIA, which requires CCTV to be high above compared to ground level. Due to privacy issues, HKAA cannot be able to provide some sample videos for testing. Footage that has a similar feature to HKIA's CCTV is quite limited online therefore our project decides to simulate a similar situation with drone footage over different Hong Kong vaccination centers, which may or may not contain a queue and people need to have certain waiting time to get vaccinated. The Hong Kong vaccination center will act as a demonstration purpose.

Density map estimation can estimate more humans compared to bounding box regression. However, density map estimation has a certain level of weakness, the data inside the generated map is quite sparse, which hard to define whether it contains humans or not. The most important is the tracking problem. Tracking cannot be performed with a density map, as most of the tracker is based on optical flow, which provides easy integration to bounding box regression. Also, the bounding box regression model provides more useful information compared to the density map. The bounding box model output an array of bounding boxes, that contain four points and can easily compute the height width inside the frame, this information is crucial to estimating the queue. Therefore, our project adopted the bounding box regression and cropping technique is used when the detection model cannot detect the human.

4. Conclusion

To address the problem raised by manual population counting, deep learning-based population density estimation and queuing flow rate estimation can be employed to optimize HKAA's management. Incorporating the mentioned concepts, our project aims to implement a dashboard web application for estimating instantaneous density information and predicting queuing flow rate with deep learning. Also, our project provides a working prototype' system design for large-scale CCTV video analytics. Notwithstanding that the experiment of density map estimation has concluded that the data inside the heatmap is quite sparse, which provides less useful information. The experiment with the bounding box model has summarized that the human inside the frame should be at least 275-pixel count to be detected by the PeopleNet model and a better clustering algorithm may boost the performance of the detection model further. HKAA will adopt the system into HKIA to provide more management insights if the result is positive. Lastly, it is believed that the application of the system would be a step forward to realize the smart airport vision and address the problem raised from manual population counting, deep learning-based population density estimation and queuing flow rate estimation can be employed to optimize HKAA's management. Incorporating the mentioned concepts, our project aims to implement a dashboard web application for estimating instantaneous density information and predicting queuing flow rate with deep learning. Also, our project provide a working prototype' system design for large scale CCTV video analytics. Notwithstanding that the experiment of density map estimation has concluded that the data inside the heatmap is quite sparse, which provide less useful information. The experiment with bounding box model has summarized that the human inside frame should be at least 275-pixel count to be detected by PeopleNet model and better clustering algorithm may boost the performance of detection model further. HKAA will adopt the system into HKIA to provide more management insights if the result is positive. Lastly, it is believed that the application of the system would be a step forward to realize the smart airport vision.

References

- [1] "HKAA Sustainability Report 2018/19," HKAA, [Online]. Available: https://www.hongkongairport.com/iwov-resources/html/sustainability_report/eng/SR1819/airport-city/smart-airport-city/. [Accessed 24 November 2021].
- [2] N. Yeung, "A cool idea: Hong Kong airport cuts energy consumption with new air-conditioning control system," [Online]. Available: https://www.scmp.com/news/hong-kong/health-environment/article/3144268/cool-idea-hong-kong-airport-cuts-energy?module=perpetual_scroll&pgtype=article&campaign=3144268. [Accessed 11 October 2021].
- [3] IBM, "What is Deep Learning?," [Online]. Available: <https://www.ibm.com/cloud/learn/deep-learning>. [Accessed 11 October 2021].
- [4] IBM, "Neural Networks," 2020. [Online]. Available: <https://www.ibm.com/cloud/learn/neural-networks>. [Accessed 24 November 2021].
- [5] T. Wood, "Convolutional Neural Network," DeepAI, [Online]. Available: <https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network>. [Accessed 11 October 2021].
- [6] L. Yann, H. Patrick, B. Leon and B. Yoshua, "Object Recognition with Gradient-Based Learning," AT&T Shannon Lab, USA, 1998.
- [7] H. Kaiming, Z. Xiangyu, R. Shaoqing and S. Jian, "Deep Residual Learning for Image Recognition," 2015.
- [8] G. Ross, D. Jeff, D. Trevor and M. Jitendra, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2013.
- [9] W. Jia and C. Antoni, "Adaptive Density Map Generation for Crowd Counting," 2019.
- [10] R. Ashmore, R. Calinescu and C. Paterson, "Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges," p. 4, 2019.
- [11] JumpGrowth, "Top 10 Web Development Frameworks in 2021 (Frontend & Backend)," [Online]. Available: <https://jumpgrowth.com/top-10-web-development-frameworks/>. [Accessed 11 October 2021].

- [12] Golang, "Effective Go," [Online]. Available: https://golang.org/doc/effective_go#concurrency. [Accessed 11 October 2021].
- [13] J. L. Gao, W. Zhao, D. Bin Wang, C. Gao and J. Wen, "C3F Framework: An Open-source PyTorch Code for Crowd Counting," *arXiv preprint arXiv:1907.02724*, 2019.
- [14] S. Karen and Z. Andrew, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2015.
- [15] D. HAN, "VGG16 学习 笔记 ," 2018. [Online]. Available: <http://deanhan.com/2018/07/26/vgg16/>. [Accessed 22 January 2022].
- [16] Q. Ji, J. Huang, W. He and Y. Sun, "Optimized Deep Convolutional Neural Networks for Identification of Macular Diseases from Optical Coherence Tomography Images," *Algorithms*, vol. 12, p. 51, 2019.
- [17] Y. Zhang, D. Zhou, S. Chen, S. Gao and Y. Ma, "Single-Image Crowd Counting via Multi-Column Convolutional Neural Network," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 589-597, 2016.
- [18] A. S. Krizhevsky, I. Hinton and G. E., "ImageNet Classification with Deep Convolutional Neural Networks," *Proceedings of the 25th International Conference on Neural Information Processing Systems*, vol. 1, p. 1097–1105, 2012.
- [19] DeepAI, "What is a Recurrent Neural Network?," DeepAI, [Online]. Available: <https://deepai.org/machine-learning-glossary-and-terms/recurrent-neural-network>. [Accessed 11 October 2021].
- [20] J. S. Sepp Hochreiter, "LONG SHORT-TERM MEMORY," Neural Computation, Germany, 1997.