Arab American University

Faculty of Graduate Studies

Accurate Pedestrian Detection for Human Crowds Using Deep Learning Techniques

By

Mohannad Mohammad Izzat Hendi

Supervisor

Prof. Dr. Mohammed Awad

This Thesis Was Submitted in Partial Fulfillment of the Requirements for the Master's Degree in Data Science and Business Analytics.

June 2023

Accurate Pedestrian Detection for Human Crowds using Deep Learning Techniques

By

Mohannad Mohammad Izzat Hendi

This Thesis was Defended Successfully on **23/07/2023** and Approved By:

Committee Members	Signature
1. **Prof. Mohammed Awad/** Supervisor:	
2. **Dr. Ahmed Ewais/** Internal Examiner:	
3. **Dr. Yousef Awwad/** External Examiner:	

Declaration

*I, Mohannad Mohammad Hendi, a student at the Arab American University's Faculty of Graduate Studies, affirm that this thesis titled "**Accurate pedestrian detection for human crowds using deep learning techniques**" is entirely my own work. I have appropriately referenced and acknowledged all resources, including those from the internet, that were utilized in this thesis. I confirm that I comprehensively grasp the concept of plagiarism and understand that any form of plagiarism will result in the immediate rejection of my thesis.*

Mohannad Mohammad Hendi

Signature:

Date: 02/12/2023.

Student ID: 201912866.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Prof. Mohammed Awad, for his continuous guidance, encouragement, and support throughout my research. His expertise and mentorship have been invaluable in helping me navigate the challenges and complexities of this project.

I am also grateful to my friend, Rabee Alqasem, for his technical assistance and insightful suggestions, which have significantly contributed to the quality of my work.

Lastly, I would like to express my heartfelt gratitude to my loving wife and my family for their unwavering love, support, and understanding throughout this journey. Their constant encouragement and belief in me have been the pillars of strength that have enabled me to persevere and achieve this milestone.

Abstract

Pedestrian detection, especially in crowded environments, is a pivotal task for various applications such as surveillance, autonomous vehicles, and crowd management. The accuracy and efficiency of pedestrian detection systems are highly contingent on the methodologies employed.

This thesis investigates and evaluates the development of deep learning-based pedestrian detection models, with a particular focus on anchor box optimization and the integration of clustering algorithms. Two distinct approaches are employed for anchor box optimization: K-Means and Fuzzy C-Means clustering. These methods aim to fine-tune anchor boxes to be more representative of the dimensions and shapes of pedestrians in the dataset. An in-depth empirical analysis is conducted to evaluate the performance of the two models.

The model incorporating K-Means for anchor calculation achieves a mean Average Precision (mAP) of 87.6% and an F1-score of 83.7%. In contrast, the model utilizing Fuzzy C-Means achieves a slightly higher mAP of 88.1% and an F1-score of 84.2%. In the inference results section, real-world images without prior annotations are used to evaluate the practical performance of both models.

Through visual inspection and comparison of bounding boxes, the study ascertains the effectiveness of each model in detecting pedestrians under real-life conditions. The thesis then concludes by highlighting the impact of anchor box optimization through clustering algorithms and providing insights into the practical deployment of deep learning models for pedestrian detection in crowded environments. In the initial phase, YOLOv7, a state-of-the-art object detection model known for its precision and speed, is adapted for

pedestrian detection through transfer learning on the CrowdHuman dataset. This research adds to the field of pedestrian detection by underlining the significance of anchor box optimization using clustering algorithms.

Importantly, the role of YOLOv7 emerges as a pivotal factor influencing our results. By integrating YOLOv7, we establish a robust baseline that informs our exploration of anchor box optimization and clustering algorithms, contributing to a comprehensive understanding of their impact in crowded environments.

Table of Contents

List of Figures

List of Tables

List of Abbreviation

AI	Artificial Inelegance
AUC	Area Under Curve
CNN	Convolutional Neural Networks
DL	Deep Learning
FC	Fully Connected
FN	False-Negative
FP	False-Positive
ML	Machine Learning
PC	Personal Computer
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristic Curve
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
YOLO-X	You Only Look Once- Series
B-BOX	Bounded Box
CV	Computer Vision
DNN	Deep Neural Network
IoU	Intersection Over Union
mAP	Mean Average Precision

Chapter 1
Introduction

1.1 Introduction

In recent years, managing large gatherings of people, known as crowd management, has become a critical aspect of public safety and organization, especially in areas with high foot traffic such as transport hubs, concerts, and sporting events [1]. Efficient crowd management is heavily reliant on technologies that can monitor and analyze human behavior, among which object detection plays a key role. Object detection, a fundamental computer vision task, deals with locating and classifying objects within images or videos [2]. Within the scope of crowd management, an essential subset of object detection is pedestrian detection. However, the unique challenge of pedestrian detection lies in the complexity of crowded environments where individuals are in close proximity, often overlapping and obscuring each other. Traditional algorithms struggle to accurately detect pedestrians in such scenarios due to occlusions, variations in scale, and a diverse range of appearances [3].

This limitation can have serious ramifications, as inaccurate pedestrian detection undermines public safety and the effectiveness of various applications that depend on it. Addressing these challenges requires innovative approaches, and here, Artificial Intelligence (AI) and, specifically, deep learning, prove to be invaluable. Deep learning, a subset of machine learning which is itself a subfield of AI, involves neural networks with numerous layers, enabling them to learn complex patterns in data [4]. These neural networks are particularly adept at handling the intricacies of images and videos, making them a natural fit for object and pedestrian detection tasks. Deep learning emerges due to

the necessity to handle the immense volumes of data produced on a daily basis, a challenge that existing machine learning models struggle to manage. DL has the capacity to grasp data attributes without human intervention, a quality that finds ideal application in the realm of computer vision. Further enhancing the capabilities of deep learning models, transfer learning has emerged as an efficacious technique. Transfer learning involves the adaptation of pre-trained models, initially trained on large datasets, for specific tasks by fine-tuning them on a smaller, task-specific dataset [5]. This method allows for leveraging the intricate features learned by these models while customizing them for specialized challenges, such as pedestrian detection in crowded environments. Figure 1-1 shows an example of Pedestrian detection.

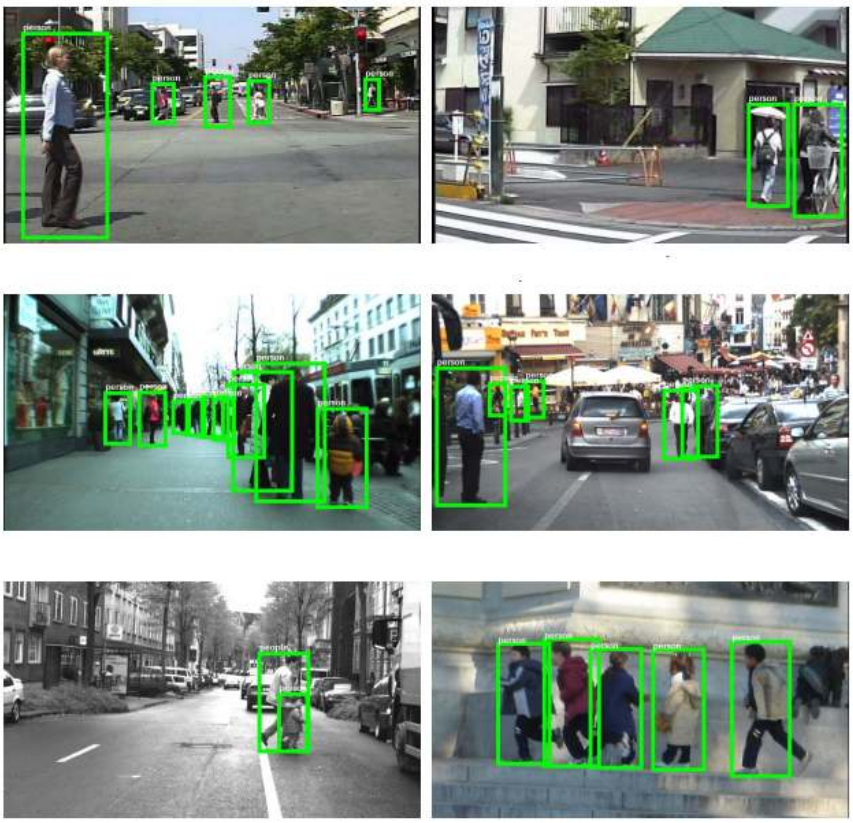

Figure 1.1-1-1: Pedestrian detection Examples [3]

In this thesis, a significant contribution is the development of a novel pedestrian detection algorithm that employs deep learning and transfer learning. This algorithm adapts the YOLOv7 model, an advanced version of the You Only Look Once (YOLO) object detection architecture, which is renowned for its real-time object detection capabilities [6]. The modified YOLOv7 model, through transfer learning, is specialized in handling the challenges posed by crowded environments, particularly occlusions, and varying scales. Not only does this algorithm significantly improve pedestrian detection performance, but it also provides a framework that can be utilized by future researchers and practitioners to further advance the field.

The motivation for this research is deeply rooted in the escalating importance of accurate pedestrian detection, particularly in crowd management systems. With the rapid urbanization and evolution of technologies such as surveillance systems and smart city solutions, there is an urgent demand for reliable pedestrian detection algorithms capable of handling crowded scenes. A significant component in addressing this challenge is the utilization of the Fuzzy C-Means algorithm for clustering bounding boxes (anchors) around the detected pedestrians. By grouping objects with similar shapes, this clustering technique ensures that the model adapts to different scales and aspect ratios, which is particularly essential in dense crowds. Moreover, the tight clustering of bounding boxes enhances the model's discernment and minimizes false positives. By harnessing the power of deep learning, transfer learning techniques, and integrating the Fuzzy C-Means algorithm, this thesis offers an innovative solution that considerably enhances the performance of pedestrian detection in crowded environments. The resulting algorithm is not only advantageous to computer vision but is also a significant stride toward improving

public safety and advancing applications such as crowd management systems that rely on accurate pedestrian detection.

1.2 Objectives

The principal objective of this thesis is to devise an algorithm that is robust, capable of real-time processing, and exceptionally accurate in detecting pedestrians, particularly in crowded environments. More specifically, the thesis endeavors to adapt the YOLOv7 model, a pre-trained deep-learning model, and modify it for the unique challenges that human crowds present. The goal here is to leverage the strengths of the YOLOv7 model and further optimize it to handle the intricate scenarios typical in crowded environments. In summary, the objectives are as follows:

- Developing a YOLOv7-based pedestrian detection algorithm: Customizing and optimizing the YOLOv7 model to be specifically attuned to the challenges of detecting pedestrians in crowded environments.
- Optimizing anchor bounding boxes using clustering algorithms: Experimenting with k-means and fuzzy c-means clustering algorithms to optimize the selection of anchor bounding boxes during the data processing phase. Investigate the impact of these optimizations on the accuracy and speed of pedestrian detection.
- Evaluating the proposed algorithm's performance: Thoroughly assessing its performance in various crowded settings, focusing on its real-time processing capabilities, accuracy, and ability to handle occlusions.
- Comparing with existing techniques: Benchmarking the proposed algorithm against other state-of-the-art pedestrian detection methods to highlight its strengths, understand its contribution to the field, and identify areas for further development.

1.3 Contribution

Pedestrian detection systems are one of the most important technical components to avoid possibly dangerous situations. Therefore, pedestrian detection is an active and challenging research topic. Thus, the main objective of this work is to implement a new deep-learning model based on YOLO-v7, to obtain a state-of-the-art system specialized in the detection of pedestrians. The pressing need for such an algorithm stems from the challenges crowded settings pose, such as occlusions and varying pedestrian appearances, which are crucial to address for the sake of public safety and enhancing the quality of life. To that end, this thesis is poised to capitalize on deep learning and transfer learning methodologies. More specifically, the thesis endeavors to adapt the YOLOv7 model, a pre-trained deep-learning model, and modify it for the unique challenges that human crowds present. The goal here is to leverage the strengths of the YOLOv7 model and further optimize it to handle the intricate scenarios typical in crowded environments by using clustering algorithms (k-means and fuzzy c-means) to optimize the selection of anchor bounding boxes during the data processing phase. The model is expected to perform efficiently and accurately in real-time, making it a practical solution for applications such as surveillance and crowd management systems. Another vital aspect of the objectives is evaluating the performance of the proposed YOLOv7-based pedestrian detection algorithm. This involves subjecting the algorithm to a range of crowded scenarios and monitoring how effectively it can detect pedestrians amidst the challenges of occlusions and varying scales. It is essential to understand how well the algorithm performs under different conditions to assess its reliability and potential limitations. Furthermore, the thesis contextualizes the developed algorithm's performance by comparing it to existing pedestrian detection techniques. This

comparative analysis is critical to objectively determine the proposed algorithm's standing in the field and understand its unique contributions and areas where it may need further refinement.

1.4 Overview

In this section of the Chapter 1 introduction, a summary will be presented for the rest of the upcoming chapters that will be discussed in this thesis. Each chapter is designed to gradually build upon the information and insights gained from the preceding ones.

Chapter 1: Introduction: establishes the foundation of this thesis by introducing the reader to the field of pedestrian detection, emphasizing its significance, particularly in crowded environments. The chapter commences with a background section, providing essential context and the impetus for conducting this research. Subsequently, the objectives of the thesis are delineated, outlining the specific research goals that guided this study. Concluding this chapter is an overview section, which acts as a roadmap, acquainting the reader with the structure and content of the succeeding chapters.

Chapter 2: Literature Review and Background: delves deeper into the domain of pedestrian detection, with an emphasis on historical and current methodologies. The chapter starts with an overview, summarizing the pedestrian detection pipeline, historical methodologies, and deep neural networks, and juxtaposing conventional approaches with deep learning methods. The chapter also contains a rigorous review of existing literature, critically analyzing prior studies in the field.

Chapter 3: Methodology and Model Development: As the cornerstone of this thesis, this chapter elucidates the methodology and systematic development of the proposed pedestrian detection model. It begins with an introduction, followed by a comprehensive

examination of the dataset. Attention is then given to data preprocessing, particularly the conversion of images and annotations to the YOLO format, and the techniques employed for anchor box selection, including K-Means and Fuzzy C-Means algorithms. The chapter progresses to discuss the selection of the YOLOv7 pre-trained model and provides a detailed exposition of the proposed methodology. The building models phase, integrating YOLOv7 with K-Means and Fuzzy C-Means algorithms, is also addressed, followed by a discussion of the performance metrics used to evaluate the model.

Chapter 4: Experiments and Results: presents an empirical evaluation of the model, analyzing the experiments conducted and their respective results. It includes the practical application of the deep learning model, highlighting the results of pedestrian detection using YOLOv7 in conjunction with K-Means and Fuzzy C-Means algorithms. Challenges and limitations encountered during the study are also candidly addressed.

Chapter 5: Conclusion and Future Directions: concludes the thesis by offering a comprehensive summary of the main findings, contributions, and implications of this research. The chapter reflects on the degree to which the initial objectives were achieved and looks ahead by suggesting potential avenues for future research and providing recommendations for researchers and practitioners.

Chapter 2
Background

2.1 Overview

2.1.1 Pipeline for Pedestrian Detection

The pedestrian recognition process tasks both classification and detection. Solving this problem in real life opens the way to a world with autonomous cars, smart surveillance, prevention, and risk mitigation. Pedestrian detection has been a pivotal research area in computer vision for several decades, with numerous methods proposed by researchers to address this challenge. A typical pedestrian detection pipeline generally encompasses three primary stages: region proposals, feature extraction, and region classification [7]. Initially, potential pedestrian regions within the input image are identified using various techniques such as sliding windows, selective search, or advanced approaches like region proposal networks (RPN) in two-stage detectors, including Faster R-CNN. Subsequently, distinctive features are extracted from each region, utilizing hand-crafted features like Histogram of Oriented Gradients (HOG) [8], Local Binary Patterns (LBP), or Haar-like features for traditional methods, or employing Convolutional Neural Networks (CNNs) to automatically learn hierarchical feature representations for deep learning-based methods [9]. Finally, the extracted features are used to classify each proposed region as pedestrian or non-pedestrian; traditional methods often rely on classifiers like Support Vector Machines (SVM) or AdaBoost, while deep learning-based methods incorporate the CNN itself for classification as part of an end-to-end trainable model [9]. It is worth noting that some deep learning-based techniques, such as single-stage detectors (e.g., YOLO, SSD), streamline the process by combining region proposal and classification into a single step, enhancing efficiency and facilitating real-time applications.

Furthermore, ongoing advancements in deep learning, including anchor-free detectors and transformer-based architectures, continue to refine and improve pedestrian detection methodologies.

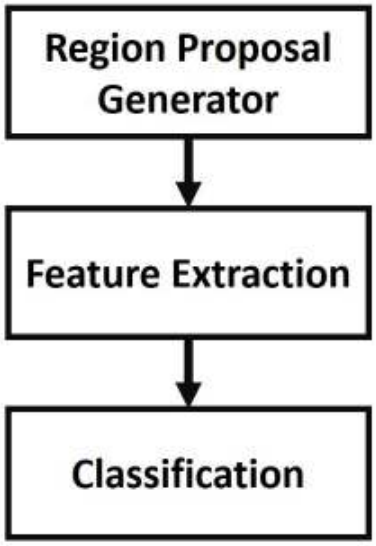

Figure 2.1-1 Pedestrian detection pipeline [7]

Following a literature overview, existing algorithms are classified into two groups based on the detection framework: (1) traditional methods and (2) deep learning.

2.1.2 Traditional Methods

In this section, a brief introduction to the key concepts related to traditional pedestrian detection techniques is provided. The primary components of these methods include region proposals, feature extraction, and region detection (classification).

Region Proposals: The initial step entails generating candidate regions or bounding boxes, which potentially contain pedestrians. Techniques such as sliding window approaches, selective search[10], or image segmentation are utilized to identify areas in the image where pedestrians are likely to present, subsequently reducing the search space for the following stages.

Feature Extraction: After obtaining the region proposals, features are extracted from each proposed region to help distinguish pedestrians from the background or other objects. Conventional approaches rely on handcrafted features, such as Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), or Haar-like features, which capture various image aspects, such as gradients, edges, or textures. These features are designed to be invariant to different lighting conditions, rotations, or scales. Region Detection (Classification): The final stage involves classifying each region as containing a pedestrian or not. This classification is typically carried out using machine learning algorithms, such as Support Vector Machines (SVM), AdaBoost, or decision trees, which are trained on a labeled dataset containing positive (pedestrian) and negative (background or other objects) examples. The process is similar to Figure 2.1.2. Once trained, the classifier can determine whether a given feature representation corresponds to a pedestrian. It should be noted that traditional pedestrian detection pipelines may also incorporate additional steps, such as non-maximum suppression, to eliminate multiple overlapping bounding boxes and retain only the most confident detections.

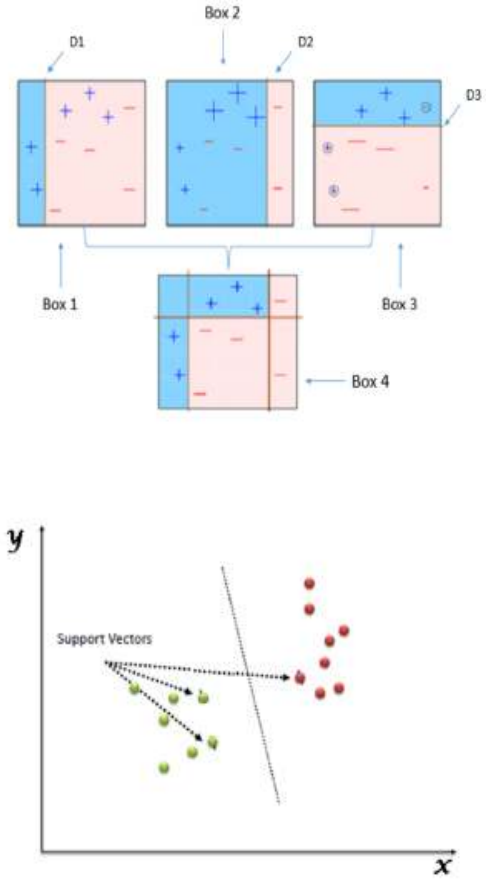

Figure 2.1-2:Left diagram that explains the Adaptive Boosting[11], *Right Diagram explain SVM*[12].

Traditional pedestrian detection methods utilize hand-crafted features and classifiers, including Harr + Adaboost, Edgelet + Bayesian, and HOG + SVM, among others. Two primary strategies are employed to address occlusion in these traditional algorithms. One approach relies on component detectors, while the other is based on an occluded classifier. [13] B. Leibe, E. Seemann developed a method for detecting pedestrians in crowded scenes, which serves as a precursor to occlusion-based pedestrian detection approaches. Their technique hinges on the integration of local and global cues through probabilistic top-down segmentation, known as intra-class occlusion. [14] suggested using logical inference, HOG, and SVM to tackle occlusion. By partitioning the input image into twelve regions, extracting feature vectors for each region, and employing an SVM

classifier, this design captures detailed information about different human body parts, such as the head, legs, arms, and torso. [15] developed a multi-object tracking method from a moving platform, which combined 3D scene tracking, complete object and object portion detectors, and explicit 3D object-object occlusion reasoning for handling partially or entirely invisible objects. Experiments with multi-person tracking demonstrated that the model can detect occluded and truncated pedestrians by enhancing weak evidence from partial human detectors through the aggregation of geometric scene constraints and data across multiple frames.

However, this approach is computationally expensive. To address this issue, Mathias and Benenson introduced Franken classifiers [16], which involve training a set of occlusion-specific classifiers more cost-effectively. Sixteen occlusion-specific classifiers can be trained at just a tenth of the cost of a single comprehensive training.

In conclusion, the traditional pedestrian detection pipeline comprises generating region proposals, extracting handcrafted features from each proposed region, and classifying these regions using machine learning algorithms. This pipeline has served as the foundation for numerous pedestrian detection approaches before the widespread adoption of deep learning methods like Convolutional Neural Networks (CNNs).

2.1.3 Deep Neural Networks

2.1.3.1 Conventional Neural Networks (CNN)

Deep Neural Networks (DNNs), including specialized architectures like Convolutional Neural Networks (CNNs), consist of multiple interconnected layers of artificial neurons or nodes. Unlike traditional methods, these networks can automatically extract and classify features from the region proposals of images. CNNs, in particular, have been

extensively employed in various computer vision tasks due to their effectiveness in processing grid-like data, such as images. Each connection between the neurons possesses an associated weight, while every neuron features a bias term. During training, these weights and biases are adjusted as learnable parameters, enabling the network to minimize the discrepancy between its predictions and the ground truth labels.

In the context of CNNs, the process commences with the input layer, which accepts the raw pixel values of the input image. Subsequently, the data traverses through several hidden layers, including convolutional layers, activation layers, and pooling layers, before reaching the output layer. Convolutional layers execute a convolution operation between the input data and a set of filters (or kernels), with each filter responsible for detecting specific features. The resulting output is referred to as a feature map. Activation layers apply non-linear activation functions, such as Rectified Linear Units (ReLU), to the outputs generated by the convolutional layers. Pooling layers reduce the spatial dimensions of the feature maps, effectively summarizing the learned features while decreasing the number of parameters and computational complexity. Figure 2.1.3.1 is an example that shows how the pedestrians in the image can be detected through a convolution neural network.

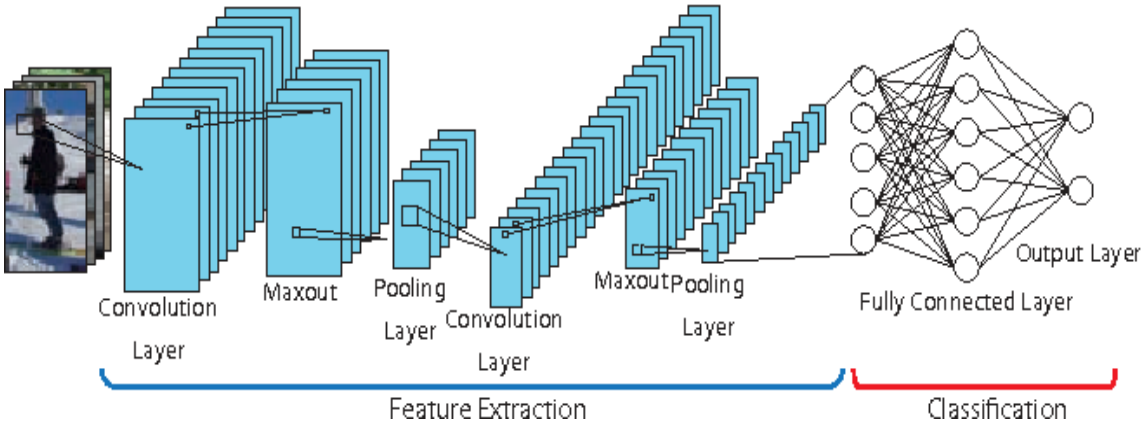

Figure 2.1-3 Example of pedestrian detection based on CNN network[17]

Following the traversal of data through the hidden layers, it arrives at the fully connected layers. These layers consolidate the high-level features acquired by the preceding layers to execute the ultimate classification or regression tasks. Ultimately, the output layer generates the network's predictions, typically employing a softmax activation function to produce probability distributions across potential classes. The network is trained through backpropagation and gradient descent (or its variants) to optimize the weights and biases. By minimizing a loss function that quantifies the difference between the network's predictions and the true labels, the network learns to deliver accurate predictions on a novel, unseen data, especially in computer vision tasks.

2.1.3.2 Transfer Learning

Transfer learning is a powerful technique in machine learning and deep learning that leverages knowledge acquired from previously trained models on related tasks to improve performance on a new, yet similar task. This approach has gained significant attention in recent years, particularly in computer vision tasks such as pedestrian detection, as it offers several advantages over training models from scratch[5][18]. The primary motivation for using transfer learning methods is the ability to capitalize on pre-trained models, which have already learned valuable features and representations from large-scale datasets, such

as ImageNet [19] or [20]. This allows for faster convergence and improved performance, even when the new task has limited labeled data. Additionally, transfer learning can help reduce the time and resources spent on developing and deploying models by reducing the computational demands of training deep learning models from scratch. There are two main transfer learning methods: feature extraction and fine-tuning. In feature extraction, a pre-trained model is utilized as a fixed feature extractor, where the output of a specific layer is fed into a new classifier for the target task [21]. This method benefits from the already learned feature representations, which can be useful for the new task, but does not allow the pre-trained model to be adapted to the target data. On the other hand, fine-tuning involves training the entire pre-trained model, or a portion of it, on the new task data [22][23]. This allows the model to adapt to the specific nuances and characteristics of the target data, often leading to better performance. However, fine-tuning requires more computational resources and labeled data than feature extraction, as the model's weights and biases need to be updated during training. Several deep learning models have been utilized for transfer learning in pedestrian detection tasks. For instance, CNN-based architectures such as AlexNet [24], VGG [25], and ResNet [26] have been used as pre-trained models to build pedestrian detectors. More recently, object detection frameworks like YOLO [27][28][29][6], SSD [30], and Faster R-CNN [31] have been employed for transfer learning in pedestrian detection due to their real-time performance and high accuracy. These models are pre-trained on large-scale datasets and can be fine-tuned or used as feature extractors for pedestrian detection tasks, significantly reducing the time and resources required for model development. In summary, transfer learning is a valuable approach in pedestrian detection that leverages the knowledge acquired from pre-trained models to enhance performance on a new task. By employing transfer learning

methods such as feature extraction and fine-tuning, researchers can develop efficient and accurate pedestrian detectors that capitalize on the strengths of existing deep learning models.

2.1.4 Traditional Methods vs Deep Learning Methods

Deep learning algorithms differ greatly from traditional algorithms in several ways. Traditional methods for feature extraction rely on specially created features made by experts, while deep learning algorithms, like CNNs, can automatically learn and extract features from data. When compared to deep learning models, which are made up of numerous interconnected layers of artificial neurons, traditional techniques often have lesser model complexity. Because of this, deep learning models typically need more data for training to avoid overfitting and achieve higher generalization. Deep learning algorithms frequently surpass conventional techniques in terms of effectiveness, especially for challenging tasks like computer vision and natural language processing. Deep learning models may learn new features and patterns from data without the need for manual feature engineering, making them more versatile. Deep learning models typically require more powerful hardware, such as GPUs, which consumes a long time in training compared with traditional models, where more computational resources will be employed. In general, the decision between traditional and deep learning techniques depends on the particular task, the data at hand, and the available computational resources.

Aspect	Traditional Methods	Deep Learning Methods
Feature extraction	Hand-crafted, manually designed features	Automatically learned features
Model complexity	Simpler models, fewer parameters	Complex models, multiple layers, many parameters
Data requirements	Fewer data required	Large amounts of labeled data typically needed
Performance(Recall-MAP)	Satisfactory with simple tasks	State-of-the-art performance on complex tasks
Adaptability	Manual feature design for new tasks	Easily adaptable using transfer learning
Computational resources	Less computationally demanding	Require powerful hardware (e.g., GPUs)

Table 2.1-1 Comparison between traditional method and deep learning methods

2.2 Related Work

Deep learning utilizes a neural network to learn pedestrian features on its own. It provides a faster detection speed and a higher detection accuracy while also saving time on manual feature selection. In deep learning for pedestrian detection, there are two primary methods to handle situations when a pedestrian is partially obscured or occluded. The first approach trains the neural network to recognize specific segments or parts of a pedestrian, even if the entire pedestrian isn't visible. The second approach refines the decision-making component of the neural network to better differentiate between pedestrians and non-pedestrians, especially in challenging situations

As [32] mentioned three mainframes of deep learning-based pedestrian detection algorithms in their paper. (1) a convolutional neural network (CNN), (2) A depth belief network (DBN), and (3) a recurrent neural network (RNN). CNN is a deep learning model for processing data with a grid pattern, such as images, that is inspired by the organization of the animal visual cortex and developed to learn spatial hierarchies of features automatically and adaptively, from low- to high-level patterns. CNN is a mathematical construct consisting of three types of layers: convolutional, pooling, and fully connected. The first two layers, convolution, and pooling extract features, and the third, a fully connected layer, map the extracted features into the final output, such as classification. Additionally, it is divided into two classes based on their mechanism: (1) two-stage detector algorithms, and (2) One-stage detectors.

R-CNN [22] is a significant milestone in CNN for object detection. It is the first neural network to propose a region proposal to realize object detection based on CNN's excellent feature extraction and classification capabilities. The region proposal obtains potential objects by sliding proposals of varying width and height, as in selective search. R-CNN introduces crop/warp before CNN normalizes candidate photos with fixed size to standard. To obtain the most accurate object detection result, it follows CNN with SVM classification and bounding-box regression. However, the region proposal causes R-CNN to use large amounts of data, time, computing, and energy. So additional R-CNN enhancement is needed. The pipeline of R-CNN is shown in Figure 2.2-1.

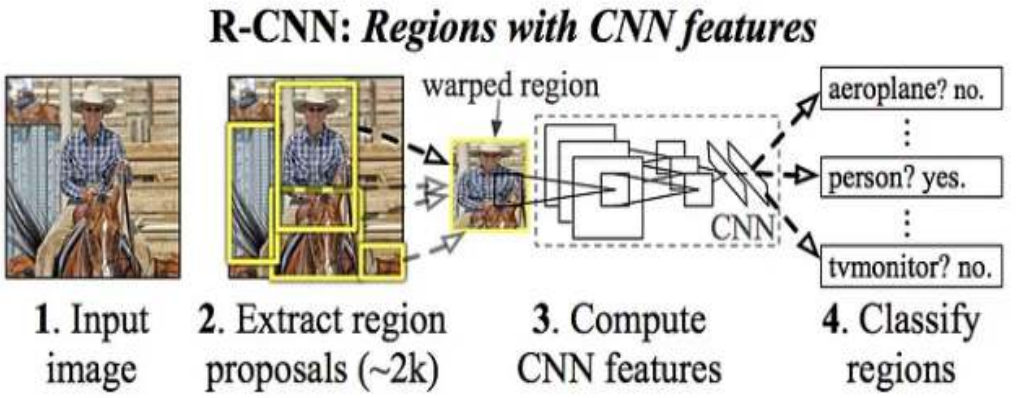

Figure 2.2-1 The pipeline of R-CNN [22]

The same author [33] who developed the R-CNN paper addressed some of its limitations by creating a faster object detection algorithm known as Fast R-CNN. This method bears a resemblance to the R-CNN approach, but instead of supplying the region proposals directly to the CNN, the input image is fed into the CNN to produce a convolutional feature map. From this feature map, region proposals are identified and warped into squares. Then, a Region of Interest (RoI) pooling layer is employed to resize these regions into a fixed dimension, making them suitable for input into a fully connected layer. A softmax layer is utilized to predict the class of the proposed region and the offset values for the bounding box based on the RoI feature vector.

The primary reason Fast R-CNN outpaces R-CNN in terms of speed is the reduced necessity to input 2000 region proposals into the convolutional neural network for each instance. Rather, a single convolution operation is performed for each image, generating a feature map from which region proposals are derived. Figure 2.2-2 shows the pipeline for Fast R-CNN.

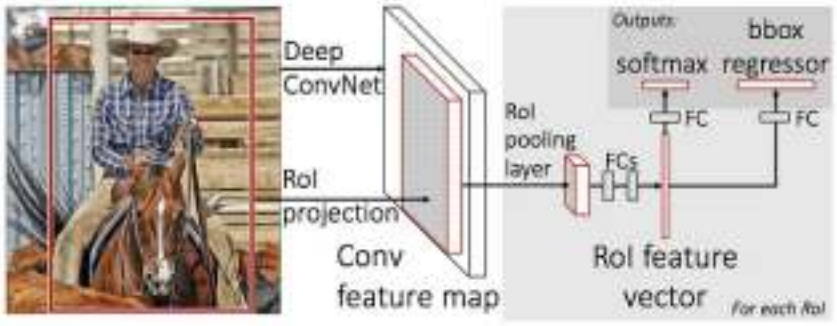

Figure 2.2-2 The pipeline of fast R-CNN

After considering R-CNN and Fast R-CNN, which both rely on selective search for region proposals, it is worth mentioning the work of Shaoqing [31] who introduced an object detection algorithm that removes the need for selective search. This approach, like Fast R-CNN, generates a convolutional feature map from the input image. However, instead of using a selective search on the feature map, a separate network predicts region proposals. These proposals are then resized using an RoI pooling layer, allowing for image classification within the proposed regions and bounding box offset value predictions. This method streamlines the object detection process and improves overall efficiency and performance. The differences between R-CNN, Fast R-CNN, and Faster R-CNN are shown in figure 2.2-3.

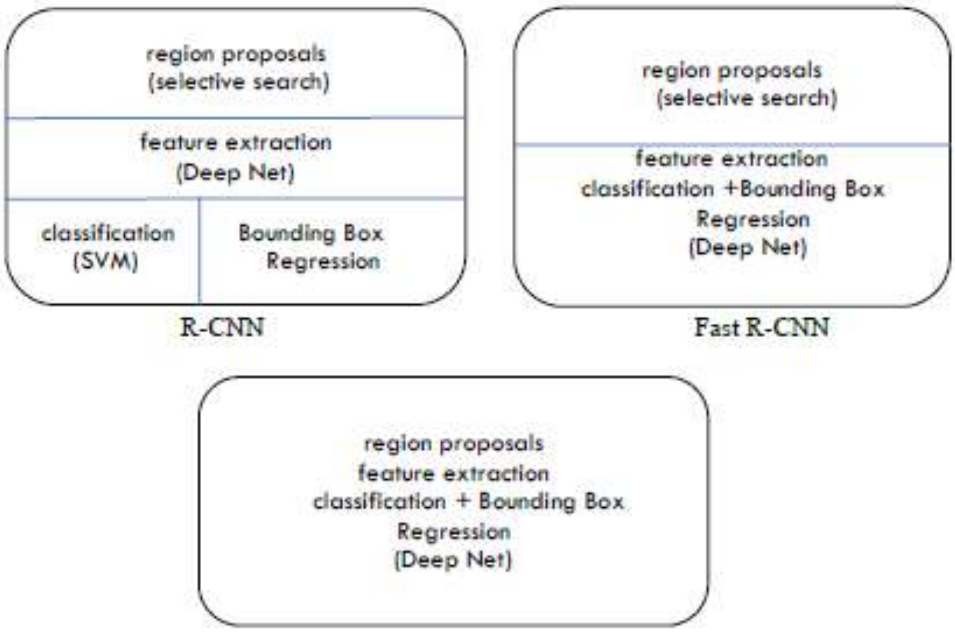

Figure 2.2-3 Methods comparison among R-CNN, Fast R-CNN and Faster R-CNN [7]

The authors [34] demonstrated a robust and successful pedestrian detection system based on the combination of many well-trained DNNs. They trained an SSD to generate rich pedestrian candidates. Then, to refine the candidates, they designed a soft-rejection-based network fusion approach that fused binary classifiers based on ResNet-50 and GoogleNet. They also proposed a method for improving pedestrian detection using semantic segmentation. Experiments on the most common dataset with various settings showed that their method works well on pedestrians of various scales and occlusions. All previous models were outperformed by the proposed method, which was also faster.

The author of Yolov3 [29] introduces prediction cross-scaling via the feature pyramid network approach. It predicts boxes on 3 separate scales and extracts features from each. The objects in scale 1 are sampled by the penultimate layer's convolutional layer. The 16*16 size feature map is added to scale 2, and the accuracy of detecting medium items is improved. The 32*32 size feature map is employed in scale 3, which enables the

detection accuracy of small-scale objects comparable to that of medium-sized objects. Furthermore, to avoid overlapping category labels for the objects, the YOLOv3 no longer employs the Softmax algorithm to classify each box. Instead, independent multiple logical classifiers are utilized, with the binary cross entropy loss representing the classification loss. Because recognizing the above-mentioned enhancements, Yolo v3 can now reach accuracy comparable to RetinaNet on the COCO dataset, while being roughly four times faster. Some advantages of YOLOv3 include its high accuracy, its ability to run on standard CPUs, its flexibility, and improves the detection accuracy of small target objects through multi-scale detection. Potential disadvantages of YOLOv3 include its relatively high computational requirements and the need for a large amount of training data. YOLOv7 [6] is the latest algorithm of the YOLO family at present, which has the strongest comprehensive performance in full-scale detection. Furthermore, It is faster; YOLOv7 can process images up to 60 frames per second, compared to YOLOv3. It is more accurate; YOLOv7 is less likely to over fit the training data and thus can generalize better to new data.

A recurrent Neural Network (RNN) receives sequence data as input and evolves the sequence recursively with all nodes linked by a chain. Common Recurrent Neural Networks include Bidirectional RNN (Bi-RNN) and Long Short-Term Memory networks (LSTM). Stewart and Andriluka offer a model based on decoding an image into a sequence of people detections in crowded scenes [35]. For sequence creation, the recurrent LSTM layer is utilized to train the model end-to-end with a new loss function that operates on sets of detections.

In this paper [36], they developed a real-time pedestrian detection system that surpasses the speed and accuracy of existing models like YOLOv3 and tiny YOLOv3. By

introducing a novel shuffle unit as the backbone, the authors created a lightweight and efficient network architecture that enhances non-linearity and expression ability, resulting in better performance. A key contribution of this research is optimizing anchor box selection through k-means clustering on the CrowdHuman dataset, which achieves higher Intersection over Union (IoU) scores during the detection process. This precise bounding box prediction is essential for pedestrian identification applications. The researchers employed a model with 26 shuffle units and 2 convolution blocks, reducing the parameter count by 65.1% and FLOPs by 67.5% compared to Darknet-53. This reduction leads to a faster computation time and more efficient image processing. The model was trained on the CrowdHuman dataset, a large-scale dataset with thousands of annotated pedestrian images, and showed a 62.44% mAP improvement over YOLOv3. The enhanced accuracy of the model stems from its deeper network and optimized anchor box selection. Experiments showed that the model's improvements primarily came from the detection of small objects. The results indicate that the proposed model is more efficient in terms of speed and accuracy, making it ideal for various computer vision applications. To sum up, the study presents a real-time pedestrian detection system that optimizes anchor box selection using k-means clustering on the CrowdHuman dataset [37] and incorporates a novel shuffle unit as its backbone. This model advances the state-of-the-art in real-time crowd pedestrian detection and has potential applications in various computer vision scenarios.

In this thesis, a comprehensive approach to pedestrian detection in crowded environments is presented by utilizing the YOLOv7 as a pre-trained deep learning model. Specifically, two distinct clustering algorithms, k-means, and fuzzy c-means, are employed to determine the anchors bounding boxes during the data processing phase. The essence of

integrating these clustering algorithms is to ascertain if different methods of computing the anchors bounding boxes can impact the accuracy and efficiency of the pedestrian detection algorithm. Consequently, two models are trained: the first model incorporates YOLOv7 with anchors bounding boxes ascertained by k-means clustering, and the second model utilizes YOLOv7 with anchors calculated through fuzzy c-means clustering. Throughout the evaluation, it was observed that the model employing fuzzy c-means clustering for calculating the anchors exhibited superior performance in terms of mean average precision compared to the model that used k-means clustering. This indicated the effectiveness of fuzzy c-means in capturing the inherent complexities and variations in the distribution of bounding boxes within crowded scenes. Moreover, it is crucial to note that all the models were trained using the CrowdHuman dataset, which is considered a benchmark for human detection, particularly in crowded environments. The dataset facilitated a realistic and challenging environment for the models, ultimately contributing to the rigor and practical relevance of this research. The findings of this thesis signify the value of the judicious selection of clustering algorithms in enhancing the performance of deep learning-based pedestrian detection systems in densely populated areas.

Chapter 3
Methodology

1.1 Introduction

In this chapter, we aim to provide a detailed exposition of the research process undertaken in this study. Our primary focus lies in the utilization of transfer learning techniques, a prevalent deep learning approach that capitalizes on pre-trained models to enhance learning efficiency and performance in similar tasks. This study relies on a chosen pre-trained model as a core building block in our pursuit of a high-performing, real-time, pedestrian detection algorithm for crowded scenarios. An integral part of our research revolves around the selection of the appropriate pre-trained model, a decision that holds considerable sway over the ultimate performance of our pedestrian detection algorithm. Our model choice was determined through a meticulous evaluation of various factors, including past performance, computational efficiency, and task suitability. This chapter delves into the reasoning behind our model selection, shedding light on the specific characteristics that led us to deem it fit for our research. Anchor boxes, essential elements in object detection models, also form a central part of our study. They act as reference bounding boxes, guiding the model to predict the accurate shape and size of objects within images. We will elucidate the preprocessing methods used for the creation of these anchor boxes, particularly the application of clustering algorithms for their selection, taking you through the process step by step. Moreover, the training phase of the selected pre-trained model constitutes a significant portion of our research methodology. We offer an in-depth view of the training process, discussing the selection of training parameters, our use of the CrowdHuman dataset, and strategies we employed to enhance the model's precision in detecting pedestrians in crowded environments. Finally, we consider model

performance evaluation as an essential part of our study. Evaluations allow us to ascertain the efficacy of our methodologies and techniques. As such, we present an elaborate overview of our evaluation process, explicating the metrics employed to measure the model's detection accuracy and processing speed.

1.2 **Dataset Description**

The CrowdHuman dataset [37] serves as an authoritative standard for assessing the efficacy of object detectors, specifically in densely populated scenarios. The distinguishing features of this dataset are its extensive size, richly annotated data, and significant diversity, all of which collectively contribute to its worth as an instrumental resource in deep learning projects. The dataset comprises an extensive collection of images - 15,000 for training, 4,370 for validation, and 5,000 for testing. Such an ample aggregation of data forms a substantial resource for both the training and assessment of the model. It includes a staggering count of 470K human instances extracted from the training and validation subsets, with an average of 23 individuals per image, thereby reflecting the complexity and intricacy of the dataset, and its ability to emulate real-world crowded settings. An integral attribute of the CrowdHuman dataset is its comprehensive annotation system. Table 3.2-1 shows the volume, density, and diversity of different human detection datasets.

	Caltech	KITTI	CityPersons	COCOPersons	CrowdHuman
# images	42,782	3,712	2,975	**64,115**	15,000
# persons	13,674	2,322	19,238	257,252	**339,565**
# ignore regions	50,363	45	6,768	5,206	**99,227**
# person/image	0.32	0.63	6.47	4.01	**22.64**
# unique persons	1,273	$<2,322$	19,238	257,252	**339,565**

Table 1.2-1 Volume, density and diversity of different human detection datasets[37].

As an example of the components in the CrowdHuman dataset, we analyze an image presented in a single figure. In this figure, Figure 3.2-1 is the unaltered image from the CrowdHuman dataset, portraying a crowd with people in various stances. Adjacent to it are four images in Figure 3.2-2, each demonstrating different annotation layers. The first of these displays bound boxes around the heads, utilizing the original CrowdHuman annotations, and is essential for facial recognition applications. The second image in the series encompasses the entire body within bounding boxes, which is beneficial for full-body analysis. The third focuses on the visible portions of individuals by employing visual body-bound boxes. The fourth and final image combines all the annotations, displaying head, full body, and visual body bounding boxes simultaneously. This aggregated image exemplifies the multifaceted annotation capabilities in the CrowdHuman dataset, crucial for human detection and analysis in crowded scenes.

Figure 1.2-1 Original image from CrowdHuman dataset.

Figure 1.2-2 Image a) represents head annotations, Image b) represents full-body annotations, Image c) represents visual body annotations, and Image d) represents combinations of all the annotations.

The following collection of images in Figure 3.2-3 serves as an example from the CrowdHuman dataset, showcasing the precision with which annotations have been applied. These annotations include bounding boxes that have been expertly drawn around heads, full bodies, and visible portions of individuals within crowded scenes.

Figure 1.2-3 Collection of images with their annotations

Each human instance within the dataset is meticulously annotated with a head bounding-box, a visible-region bounding-box, and a full-body bounding-box as shown in Figure 3.2-1. Such a detailed annotation aids profound analysis and accurate detection, especially while dealing with diverse and obscured instances. In addition, the dataset incorporates various levels of occlusions, thereby escalating its complexity. This feature is paramount as it closely represents real-world circumstances where individuals in crowded environments may be partially concealed or obstructed. By utilizing such data to train our model, we aspire to augment its capacity for accurately detecting pedestrians, irrespective of the challenging conditions. Through the course of this study, we intend to employ the CrowdHuman dataset as a robust foundation that can catalyze future advancements in human detection tasks. We posit that its heterogeneous and richly annotated data will significantly contribute to the evolution of more sophisticated and precise detection models.

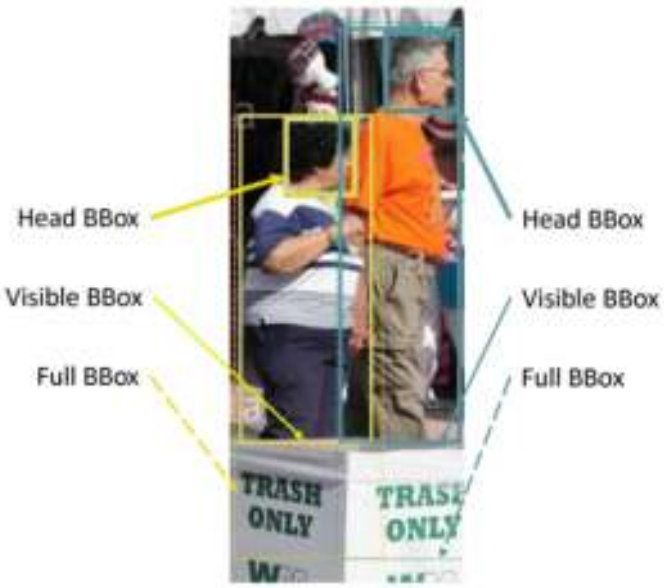

Figure 1.2-4 provides an illustrative example of the three kinds of annotations: Head Bounding-Box, Visible Bounding-Box, and Full Bounding-Box[37]

The CrowdHuman dataset comprises annotation_train.odgt and annotation_val.odgt files, contains annotations for our dataset. These files are favored for their reader-friendly nature, with each line functioning as a JSON string that encapsulates all annotations associated with a particular image. The structure of the JSON annotation contains two fields: an "ID", which signifies the image's filename, and "gtboxes", encompassing one or more 'gtbox' elements. These 'gtbox' elements each denote a specific box annotation with several key components. One of these components is the "tag" field, which designates the box type as either "person" or "mask". There are also "vbox", "fbox", and "hbox" fields, signifying the visible box, full box, and head box's dimensions respectively. These are arrays consisting of x and y coordinates, width (w), and height (h). Furthermore, "gtbox" may encompass two optional fields, namely "extra" and "head_attr". The former carries attributes pertaining to the person, while the latter concerns the head's attributes. Both fields may contain "ignore", "box_id", and "occ" keys, but their presence is not guaranteed. Significantly, during the preprocessing phase of our model training, we chose to solely focus on body data and intentionally disregarded 'mask' data. When a 'mask' tag is present, it indicates that the box pertains to crowd, reflection, or elements akin to a person, and should be ignored. This is typically signaled by the 'ignore' attribute being set to 1 in the 'extra' field. However, for our model, these elements were considered irrelevant, so only 'person' tagged data - representing body annotations - was selected and utilized for training.

1.3 **Data Preprocessing Phase**

1.3.1 Convert Images Name & Annotation to Yolo Format

In the data preprocessing phase, the annotations from the CrowdHuman dataset were converted to a format compatible with YOLO. This was a critical step, as the YOLO model requires a specific input format for annotations. The CrowdHuman dataset annotations were initially in a JSON format, whereas YOLO requires the annotations to be in text files with specific information about the bounding boxes of objects. The first step involved obtaining the dimensions of each image in the dataset. Since the dimensions of images are essential for normalizing the bounding box coordinates, each image was read, and its width, height, and number of channels were recorded. Next, for each annotation corresponding to an image, a line was generated in the YOLO-compatible annotation file. Each line contained information about the class of the object and the normalized center coordinates, width, and height of the bounding box. The center coordinates were normalized by dividing by the width and height of the image respectively. Similarly, the width and height of the bounding box were normalized. This normalization is crucial for making the model invariant to the size of the input image. It is worth noting that during this conversion process, care was taken to ensure that only relevant objects were included. In the context of this work, only annotations corresponding to humans were considered. Any annotations that were too small to be relevant were also excluded.

This was done to avoid training the model on objects that are too small to be detected reliably. Once the annotation text files were created for each image, two additional text files were generated: one for training and the other for testing. These files contained the

file paths to the images used for training and testing, respectively. They are crucial for informing the model where the training and testing data is located. Finally, a configuration file with a .data extension was created. This file contained essential information for training the YOLO model, including the number of classes, paths to the training and validation datasets, names file, and backup directory. This data preprocessing phase ensured that the annotations from the CrowdHuman dataset were structured appropriately for efficient training of the YOLO model. With the annotations converted to YOLO format, the training process could be initiated.

In the data preprocessing phase, one of the most critical steps is to verify the accuracy and correctness of the annotations associated with the images in the dataset. This is of paramount importance as the quality of the annotations directly impacts the performance of the object detection model. To address this need, a specialized script is developed that systematically goes through the images and their corresponding annotation files. The script functions by reading the annotation files, which store the details of the bounding boxes in a structured format. These details include information about the class of the object and the coordinates of the bounding box. With this information at hand, the script overlays these bounding boxes on the images. Different colors are employed for bounding boxes to enhance visibility and provide a clear distinction. For example, one might use red rectangles to indicate humans. This visual inspection is not merely a luxury but a necessity. It ensures that the objects within the images are correctly and precisely tagged, which is an absolute requirement for the effective training of object detection models. When we visualize these annotations, we get a clear picture of how the model will interpret the raw data.

This step can be seen as a sanity check for the integrity of the dataset. Furthermore, it helps in identifying any misalignment between bounding boxes and the actual objects. Such discrepancies, if left unchecked, can adversely affect the model's ability to learn. By utilizing this script, researchers and developers can ensure that the dataset is reliable and well-prepared for training. This verification step serves as a diagnostic tool to detect any anomalies or errors in the dataset annotations early in the development cycle. This early detection is invaluable as it allows for the necessary corrections and refinements to be made before investing time and resources in training the model. The emphasis on meticulous verification underlines the commitment to high standards of data quality, which, in turn, is expected to culminate in a more robust and accurate object detection model. Figure 3.3-1 displays a verifying step for an image.

Figure 1.3-1 Verification step for image from CrowdHuman dataset

1.3.2 Anchor Box Selection Techniques

In the data preprocessing section of our thesis, we employ two methods to extract anchor boxes, which play a crucial role in achieving accurate object detection and localization. We utilize two clustering algorithms, namely k-means and fuzzy c-means, to calculate predefined bounding boxes. These anchor boxes serve as predefined shapes that our model utilizes to predict the bounding boxes of objects in an image. The incorporation of anchors during the training of a YOLO (You Only Look Once) model is vital for enhancing detection and localization accuracy. These anchors enable the model to effectively detect and localize objects of varying sizes and shapes, even in scenarios where objects may be partially occluded or possess unconventional orientations. By adjusting the anchor shapes to align with the distribution of object sizes and shapes in the training data, our model improves its predictive accuracy. Without the presence of anchors, our model may struggle to accurately determine the location and size of objects within the image, resulting in diminished overall detection accuracy.

Therefore, the utilization of anchors becomes an indispensable component in training a YOLO model to effectively detect and localize objects within the processed images. Consequently, after this preprocessing step, we utilize the anchor values as parameters for the YOLOv7 model. To extract the anchor boxes, we employ two clustering algorithms: k-means and fuzzy c-means. These algorithms enable us to calculate predefined bounding boxes that serve as anchor shapes for our YOLO model. The k-means algorithm partitions the data points into clusters by minimizing the squared distance between the data points and the centroid of each cluster. Similarly, the fuzzy c-means algorithm assigns membership degrees to data points, determining the extent to which each data point belongs to each cluster.

1.3.2.1 K-Means Algorithm

K-means clustering is a widely-used, iterative algorithm that aims to partition a set of points into K clusters, where each point belongs to the cluster with the nearest mean. The algorithm begins by randomly selecting K points as initial centroids. Each point in the dataset is assigned to the nearest centroid, and then the centroids are recalculated as the mean of all the points in the cluster. These two steps are repeated until the centroids no longer change, or other termination conditions are met[38], [39]. Figure 3.3-2 shows an illustration of the K-means algorithm.

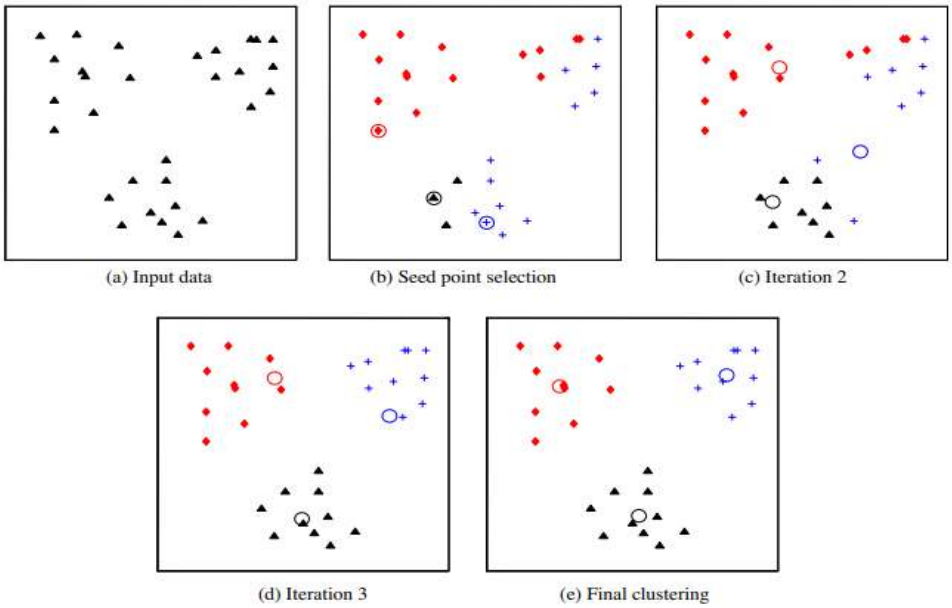

Figure 1.3-2 . Illustration of K-means algorithm. (a) Two-dimensional input data with three clusters; (b) three seed points selected as cluster centers and initial assignment of the data points to clusters; (c) and (d) intermediate iterations updating cluster labels and their centers; (e) final clustering obtained by K-means algorithm at convergence [38].

The mathematical formulation of the K-means algorithm can be defined as an optimization problem. Let's denote the dataset as X = {x1, x2, ..., xn}, where xi represents each data point. Let C = {c1, c2, ..., ck} be the set of centroids for the K clusters, and let μk be the mean of points in cluster ck. The objective of the K-means algorithm is to minimize the within-cluster sum of squares (WCSS), also known as inertia, which

measures the sum of squared distances between each data point and the centroid to which it is assigned. Mathematically, this can be expressed as:

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in c_k} \|x_i - \mu_k\|^2 \qquad (3.1)$$

where

$$J(c_k) = \sum_{x_i \in c_k} \|x_i - \mu_k\|^2 \qquad (3.2)$$

Is the squared Euclidean distance between data point xi and the mean of the cluster ck, μk. This represents the squared error between the mean of cluster ck and the points within that cluster. The K-means algorithm iteratively refines the centroids C to minimize J(C) by reassigning data points to the cluster whose mean is closest and then updating the means of the clusters based on the newly assigned points. This process is repeated until convergence, i.e., when the assignments no longer change, or a maximum number of iterations is reached. The result is a set of clusters that collectively have the smallest possible sum of squared distances to their respective centroids.

In the context of object detection in deep learning, K-means clustering can be employed to optimize anchor boxes. Anchor boxes are predefined bounding boxes that are used as references for predicting the actual bounding boxes of objects in an image. The choice of anchor boxes is crucial because it can significantly impact the performance of the object detection model. The YOLOv2 paper introduced the concept of using K-means clustering on the training set bounding boxes to select anchor boxes that best represent the size and shape of the objects in the dataset[28]. In this thesis, K-means clustering is utilized to calculate anchor bounding boxes tailored to the CrowdHuman dataset. By customizing

the anchor boxes based on the dataset's specific characteristics, the object detection model can be more sensitive to the various scales and aspects of pedestrians in crowded environments. Using the K-means algorithm, the sizes and ratios of the anchor boxes are optimized to best match the ground truth bounding boxes in the training data. This can contribute to improving the accuracy of the model by reducing the misalignment between the predicted bounding boxes and the ground truth, ultimately resulting in a higher mean Average Precision (mAP).

K-means with anchors workflow:

Initialization: Start by selecting k initial anchors randomly. These anchors are typically bounding box dimensions (width and height) from the dataset.

Assignment: For each bounding box in the dataset, calculate its IoU (Intersection over Union) with each of the k anchors. Assign the bounding box to the anchor with which it has the highest IoU. This is a substitute for the typical distance metric used in k-means, tailored for the task of bounding box dimension clustering.

Update: For each of the k clusters of bounding boxes, calculate the new anchor dimensions as the average width and height of all bounding boxes in that cluster.

Repeat: Continue the Assignment and Update steps until the anchor dimensions do not change significantly, or a predefined number of iterations have been reached.

1.3.2.2 Fuzzy C-Means Algorithm

Fuzzy clustering represents a robust and versatile unsupervised technique for data analysis and model generation. Often, fuzzy clustering proves to be more intuitive compared to hard clustering, particularly in scenarios where data points are not strictly partitioned into distinct clusters. Instead of forcing data points at the intersections of clusters to belong exclusively to one cluster, fuzzy clustering assigns their degrees of

membership ranging between 0 and 1, reflecting their proportionate affiliation to the clusters[40]. Among the various fuzzy clustering algorithms, Fuzzy C-means (FCM) is the most prevalent. The FCM algorithm was initially introduced in its specialized form (with m=2) by Joe Dunn in 1974 [41]. The algorithm was then generalized for any m greater than 1 by Jim Bezdek in his Ph.D. dissertation at Cornell University in 1973 [42]. Figure 3.3-3 displays the mechanism for hard and soft clustering.

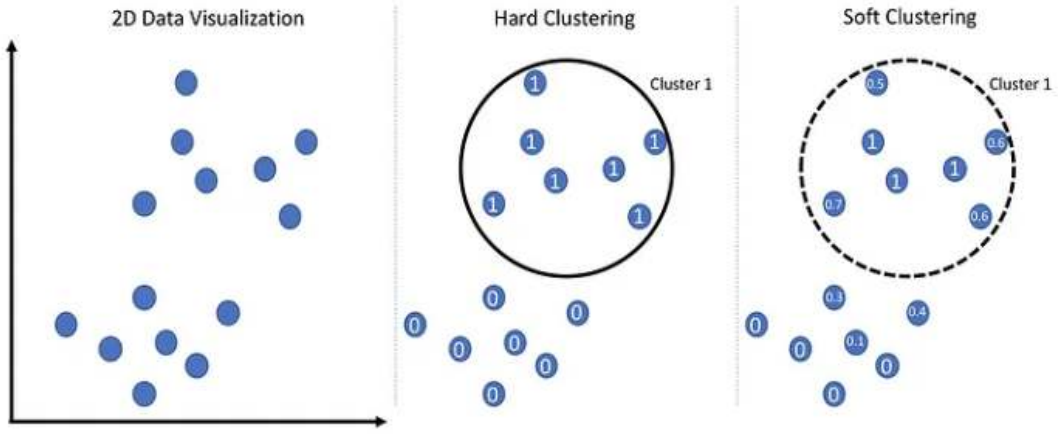

Figure 1.3-3 First image represents 2D data , Second image represents hard clustering (K-means),Third image represents soft clustering (Fuzzy C means).

The mathematical expressions that govern the algorithm are as follows:

The cluster center, c_i, for the i-th cluster, is calculated as the weighted average of all the data points, where the weights are the membership degrees raised to the power of the fuzzifier, m. The equation for computing c_i is:

$$c_i = \frac{\sum_{j=1}^{N} \mu_{ij}^{m} x_j}{\sum_{j=1}^{N} \mu_{ij}^{m}} \qquad (3.3)$$

Here, μ_ij denotes the membership degree of data point x_j in cluster i.

The membership degree, μ_ij, is more involved. It is computed relative to the distances of the data point from the various cluster centers and considers the fuzziness parameter, m. The equation for μ_ij is:

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_j - c_i\|}{\|x_j - c_k\|} \right)^{\frac{2}{m-1}}} \qquad (3.4)$$

In the context of object detection, applying Fuzzy C-means clustering for the calculation of anchor bounding boxes can be beneficial. By allowing for a more nuanced membership of bounding boxes in clusters, Fuzzy C-means can effectively capture the varying scales and aspect ratios of objects within the dataset. This is particularly useful for datasets where objects may have significant overlaps or similarities in dimensions. In this thesis, Fuzzy C-means clustering is applied as a pre-processing step to optimize the anchor bounding boxes for the CrowdHuman dataset. The anchor boxes are calculated such that they are highly representative of the diverse shapes and sizes of the objects in the dataset. This optimization can lead to improved performance in object detection, especially in terms of localization accuracy and handling occlusions in crowded scenes. Furthermore, as Fuzzy C-means is distinct from K-means, it provides an alternative methodology for calculating anchors, offering flexibility in handling various datasets and scenarios.

Fuzzy C-Means with anchors workflow

Initialization: Select c initial anchors randomly, just like in k-means.

Membership Assignment: For each bounding box in the dataset, calculate its degree of membership to each anchor. This could be done based on the IoU with each anchor, similarly to k-means, but the calculation would be more complex because each bounding box can belong to multiple anchors with different degrees of membership. A typical

membership function might assign higher membership to anchors with higher IoU, and lower membership to anchors with lower IoU.

Update: For each of the c clusters of bounding boxes, calculate the new anchor dimensions. This calculation would be a weighted average of the width and height of all bounding boxes in the dataset, where the weights are the degrees of membership of each bounding box to the anchor.

Repeat: Continue the Membership Assignment and Update steps until the anchor dimensions do not change significantly, or a predefined number of iterations have been reached.

Choice of Number of Clusters (Anchors): In this study, we have chosen to utilize nine clusters, following the conventional practice in YOLO-based models. This decision is based on the observations from previous research that nine anchors typically provide a balance between model complexity and detection performance across a wide range of datasets. While the number of anchors is a hyperparameter that can be optimized, this optimization requires additional computational resources and might not lead to a significant improvement in the model's performance. Hence, we decided to start with nine anchors, aligning with the established practices in the field. Our choice was reinforced by the results we obtained, which demonstrate the satisfactory performance of our YOLO model on the CrowdHuman dataset. The incorporation of anchor boxes addresses various challenges in object detection, such as objects with different sizes, shapes, or orientations. These predefined shapes enable the model to generalize well and accurately detect objects, even when they are partially occluded or have uncommon orientations. Without anchors, the model might struggle to make precise predictions, leading to reduced detection accuracy overall. Figure 3.3-4 shows the anchors for each algorithm.

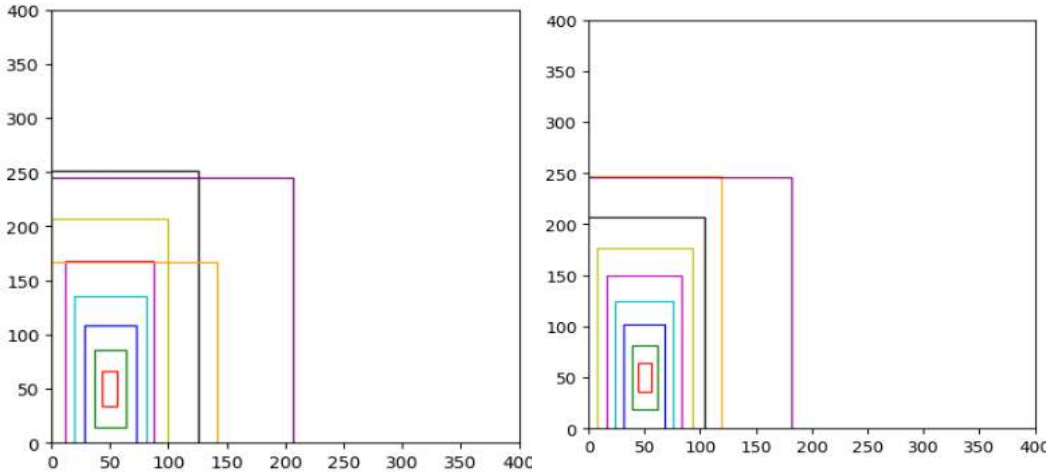

Figure 1.3-4 Generate 9 clustering boxes as anchors. (a) Clustering box dimensions using k-means. (b) Clustering box dimensions using fuzzy-c-means

1.4 **Pre-Trained Model Selection (YOLOv7)**

The critical decision to select a pre-trained model in a deep learning research project can directly impact the overall system performance. After thoughtful consideration, we have chosen YOLOv7 as the pre-trained model for our study [6], guided by its historical performance, architectural design, computational efficiency, and task suitability - pedestrian detection in crowded environments. YOLOv7, an advanced version of the You Only Look Once (YOLO) series of models, is recognized for its exceptional performance in object detection tasks. It strikes a delicate balance between precision and speed, a characteristic vital for real-time pedestrian detection applications, where accuracy and processing speed are of utmost importance.

The YOLOv7 model leverages the strengths of its predecessors and integrates several architectural enhancements. It abides by the core principle of YOLO models - predicting bounding boxes and class probabilities by processing an image in a single pass. However, YOLOv7 introduces improvements in terms of upgraded backbone networks, more effective anchor box determination, and enhanced strategies for handling object scale

prediction. These collective enhancements contribute to improved detection accuracy and speed. The choice of YOLOv7 is further substantiated by its compatibility with the CrowdHuman dataset, which we employ for training our model. YOLOv7 has demonstrated its effectiveness across diverse datasets, and its adaptability is anticipated to be advantageous when dealing with the complexities of the CrowdHuman dataset. In conclusion, we selected YOLOv7 as the pre-trained model for this study based on its robust architecture, proven performance, and suitability for pedestrian detection in crowded environments. The subsequent sections will offer a more detailed discussion of the utilization of this pre-trained model in our study.

In YOLOv7, there is a new design for how the network is built, and you can see this in Figure 3.4-1. To make YOLOv7 better, the creators made several changes. These include better ways of combining layers, adjusting the size of the model, and making changes to the settings. YOLOv7 is good at finding objects in pictures quickly compared to other similar models. Here's how YOLO works [43]:

i. First, it makes the input image smaller before it goes through the network.
ii. Then, it uses a 1x1 convolution to lessen the number of channels. After that, it uses a 3x3 convolution to produce the final output.
iii. It also uses extra tricks like batch normalization and dropout to make sure the model doesn't overfit. This means it will be good at handling new images it hasn't seen before.

In simple words, YOLOv7 makes sure that important features in images are combined well, which is great for finding objects of different sizes. By changing the size of the model, it can work well with different amounts of data and computer power. Also, by

adjusting the settings, the model becomes strong and can work well in different situations. All these improvements make YOLOv7 top-notch in finding objects in images quickly.

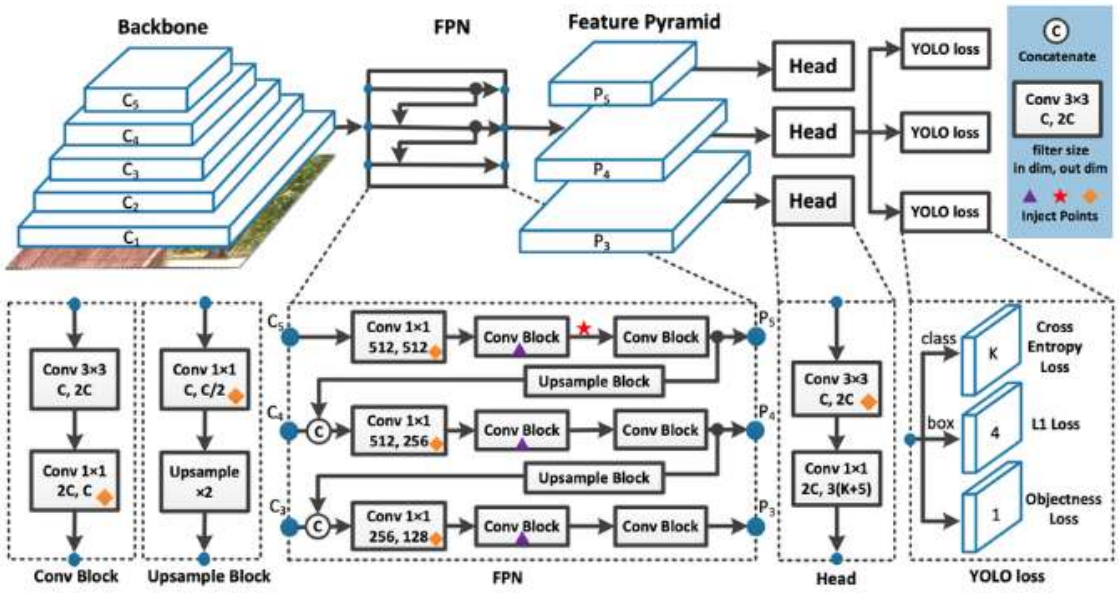

Figure 1.4-1 YOLOv7 network architecture [6]

1.5 **Proposed Method & Building Models Phase**

In the proposed method, we take advantage of a pre-existing model known as YOLOv7. This model is particularly good at spotting different objects in images quickly and accurately. It has already been trained with a large collection of images. It was originally trained on the COCO dataset. COCO stands for Common Objects in Context. This dataset is a big deal in the world of image recognition. It's like a massive album of over 330,000 images. What's special about it is that each image doesn't just have objects, but also has labels telling us what's what – for 80 different categories! And if that wasn't enough, it also comes with five captions for each image to describe the scene. This dataset is a treasure for researchers and is used to build and test some of the best models for spotting and labeling objects in images [20]. Now, coming to our task: we don't build a model

from scratch. Instead, we take YOLOv7 and give it a bit of extra training that's specific to our goal. This is known as Transfer Learning. Imagine YOLOv7 as a skilled worker who already knows a lot but needs a little extra training for a specialized job. In this case, we're interested in spotting people in images. For that, we use the CrowdHuman dataset, which is like a collection of images focusing on people, or as some might say, pedestrians. Here's how it all comes together: We take the YOLOv7 model, which is already pretty smart because it's been trained on the extensive COCO dataset. We then tune it up with additional training on the CrowdHuman dataset. This blend of knowledge makes the model tailored and sharp for spotting people in images. This approach is efficient because we don't start from zero. We build upon something that's already quite capable and make it just right for what we need. Figure 3.5-1 shows our model's general pipeline.

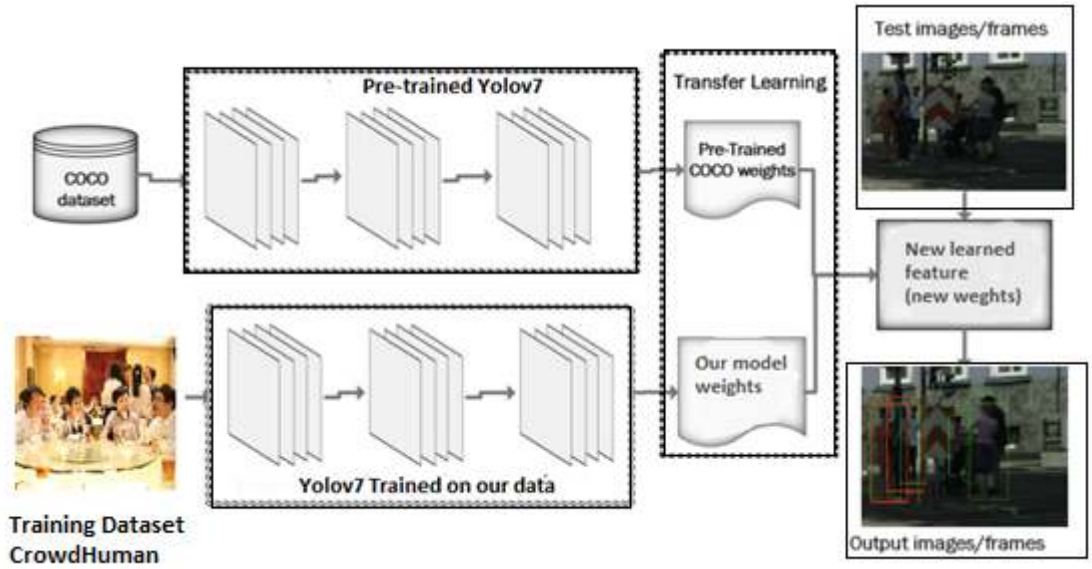

Figure 1.51.5-1 Our Approach general pipeline

In the Building Models Phase of this project, several critical steps are taken to develop an adept model for detecting pedestrians. To kick things off, anchors are calculated during data preprocessing. Anchors are essentially reference boxes that are instrumental in

detecting objects. K-means clustering and Fuzzy C-means clustering are the two algorithms applied here to fine-tune the anchors to the CrowdHuman dataset. This ensures that these anchors align well with the true dimensions and shapes of pedestrians in the images. Next up, the groundwork is laid by using a pre-trained model. In this scenario, YOLOv7 is the chosen model, renowned for its precision and speed in object detection. Since YOLOv7 is already trained on the extensive and diverse COCO dataset, it is well-equipped with learned features that can be a solid foundation for the pedestrian detection task. The building models phase rounds off with training the model through transfer learning. YOLOv7 is fine-tuned for the specialized task of pedestrian detection using the CrowdHuman dataset. Importantly, two distinct models are developed - the first model incorporates anchors calculated via K-means clustering, while the second model utilizes anchors derived from Fuzzy C-means clustering. This differentiation in anchor calculation methods is expected to offer valuable insights into which approach optimally supports the task of pedestrian detection. Figure 3.5-2 shows our model schema.

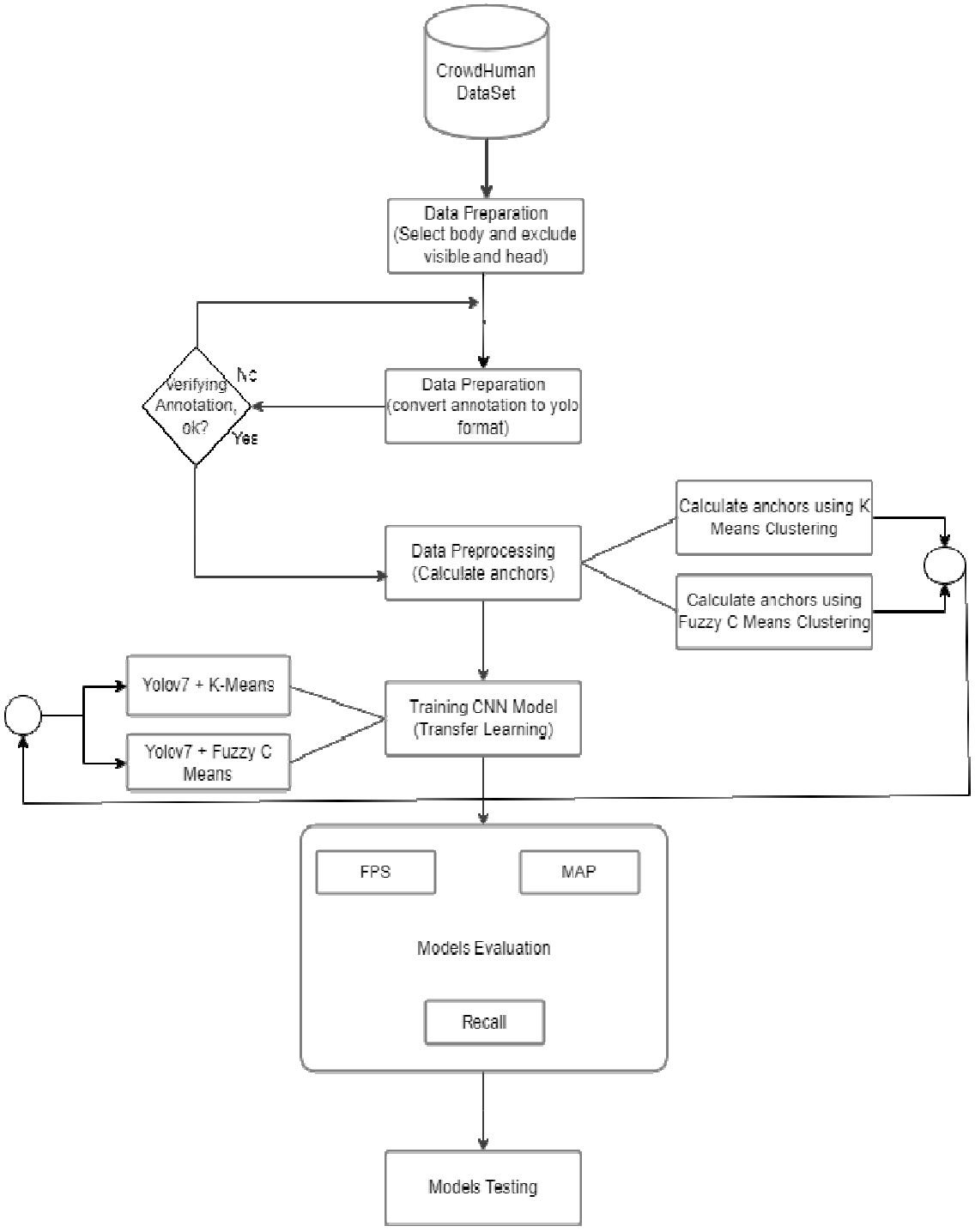

Figure 1.5-2 Our models schema

1.5.1 Yolov7 and Anchors calculated by the K Means algorithm

In this segment, we focus on developing a model by combining YOLOv7 with anchors calculated through the K-means algorithm. YOLOv7 is an advanced object detection

model known for its high speed and accuracy. It's extremely efficient in recognizing and locating objects within images. To tailor YOLOv7 for pedestrian detection, we employ anchors - reference boxes used to predict the bounding boxes of detected objects. These anchors are fine-tuned using the K-means algorithm. Essentially, K-means clustering helps in adjusting the anchors to better represent the shapes and sizes of pedestrians in the dataset. This combination of YOLOv7 and K-means-optimized anchors aims to create a robust model for precise pedestrian detection. Below is the pseudocode that outlines the steps involved in this process.

Algorithm 1: The general procedure that was used transfer learning by utilizing yolov7 on CrowdHuman dataset

Input: Dataset, hyperparameters;
Output: Detect pedestrian;

```
Begin
// Step 0: Download the CrowdHuman dataset
function downloadCrowdHuman():
    download the dataset from the CrowdHuman website
    return dataset
end function
// Step 1: Data Preparation - Select the body and exclude visible and head
function prepare data(dataset):
    for each image in the dataset:
        annotations = get annotations(image)
        for each annotation in annotations:
            if annotation. the label is 'body':
                keep annotation
            else:
                exclude annotation
        end for
    end for
    return filteredDataset
end function
// Step 2: Convert Annotations to YOLO format
function convertToYOLOFormat(filtered dataset):
    for each image in the filtered dataset:
        convert image. annotations to YOLO format
    end for
    return yoloFormattedDataset
end function
// Step 3: Data Preprocessing - Calculate anchors using K Means Clustering
function calculateAnchors(yoloFormattedDataset):
    bounding boxes = extractBoundingBoxes(yoloFormattedDataset)
    anchors = kMeansClustering(bounding boxes)
```

```
    return anchors
  end function
  // Step 4: Training CNN Model using Transfer Learning
  function train model (yoloFormattedDataset, anchors):
    yoloV7 = loadPretrainedYOLOv7()
    yoloV7.setAnchors(anchors)
    train yoloV7 using yoloFormattedDataset
    return trained model
  end function
  // Step 5: Save the trained model
  function saveModel(trained model):
    save the trained model to disk
  end function
  // Main Process
  dataset = downloadCrowdHuman()
  filteredDataset = prepareData(dataset)
  yoloFormattedDataset = convertToYOLOFormat(filtered dataset)
  anchors = calculateAnchors(yoloFormattedDataset)
  trained model = train model(yoloFormattedDataset, anchors)
  save model(trained model)

End
```

Figure 1.5-3 First model pseudocode

1.5.2 Yolov7 and Anchors calculated by Fuzzy C Means algorithm

In our approach, we make use of YOLOv7, which is a highly efficient and robust deep-learning model for real-time object detection. Given its effectiveness in recognizing various objects in images, it is a natural choice for our needs. However, to enhance its performance for our specific task, which is pedestrian detection, we employ a unique technique to calculate anchors using the Fuzzy C Means algorithm. Anchors are essentially predetermined bounding boxes that can detect objects at different scales and aspect ratios. The Fuzzy C Means algorithm is a clustering technique, and instead of assigning each data point to a distinct cluster, it calculates the membership degree, signifying that data points can belong to multiple clusters to a certain degree. By using this algorithm to calculate anchors, we can achieve a more adaptable and potentially more

accurate representation of the bounding boxes required for pedestrian detection. The following pseudocode provides a detailed overview of this process.

Algorithm 2: The general procedure that was used transfer learning by utilizing yolov7 on CrowdHuman dataset

Input: Dataset, hyperparameters;

Output: Detecting pedestrians;

```
Begin
  // Step 0: Download the CrowdHuman dataset
  function downloadCrowdHuman():
    download the dataset from the CrowdHuman website
    return dataset
  end function
  // Step 1: Data Preparation - Select the body and exclude visible and head
  function prepare data(dataset):
    for each image in the dataset:
      annotations = get annotations(image)
      for each annotation in annotations:
        if annotation. the label is 'body':
          keep annotation
        else:
          exclude annotation
      end for
    end for
    return filteredDataset
  end function
  // Step 2: Convert Annotations to YOLO format
  function convertToYOLOFormat(filtered dataset):
    for each image in the filtered dataset:
      convert image. annotations to YOLO format
    end for
    return yoloFormattedDataset
  end function
  // Step 3: Data Preprocessing - Calculate anchors using Fuzzy C Means Clustering
  function calculateAnchorsUsingFuzzyCMeans(yoloFormattedDataset):
    bounding boxes = extractBoundingBoxes(yoloFormattedDataset)
    anchors = fuzzyCMeansClustering(bounding boxes)
    return anchors
  end function
  // Step 4: Training CNN Model using Transfer Learning
  function trainModelUsingFuzzyCMeans(yoloFormattedDataset, anchors):
    yoloV7 = loadPretrainedYOLOv7()
    yoloV7.setAnchors(anchors)
    train yoloV7 using yoloFormattedDataset
    return trained model
  end function
  // Step 5: Save the trained model
  function saveModel(trained model):
    save the trained model to disk
  end function
```

```
// Main Process
dataset = downloadCrowdHuman()
filteredDataset = prepareData(dataset)
yoloFormattedDataset = convertToYOLOFormat(filtered dataset)
anchors = calculateAnchorsUsingFuzzyCMeans(yoloFormattedDataset)
trained model = trainModelUsingFuzzyCMeans(yoloFormattedDataset, anchors)
save model(trained model)

End
```

Figure 1.5-4 Second model pseudocode

1.6 **Performance Metrics Selection**

In the evaluation phase of the experiment, it is imperative to assess the performance and reliability of the pedestrian detection models rigorously. We have developed two models that employ YOLOv7 architecture with different approaches to calculating anchors - one using K-means clustering and the other using Fuzzy C-means clustering. To objectively analyze and compare the performance of these models, a diverse set of evaluation metrics and visualization tools are used. These include Intersection over Union (IoU), the confusion matrix, precision, recall, mean average precision (mAP), F1 score, frames per second (FPS), F1 curve, Precision (P) curve, Precision-Recall (PR) curve, and Recall (R) curve. IoU evaluates the overlap between the predicted bounding boxes and the ground truth, while the confusion matrix forms the foundation for calculating precision and recall. Precision is indicative of the accuracy of pedestrian detections, while recall measures the ability of the model to detect all pedestrians within the scene. The PR curve is a plot that showcases the trade-off between precision and recall. The F1 curve, as a function of the threshold, helps to find the point where precision and recall are harmoniously balanced, and the F1 score is the harmonic mean of precision and recall. The mAP summarizes the PR curve and incorporates IoU. Frames per second (FPS) quantifies the inference speed of the model, critical for real-time applications. The combination of these metrics and

visualization tools facilitates a comprehensive analysis, enabling the identification of the strengths and limitations of each model in terms of accuracy, reliability, and speed, which is vital for practical applications and further improvements.

Intersection over Union (IoU): IoU, also known as the Jaccard index, is a metric that evaluates the overlap between the ground truth bounding box and the predicted bounding box. In object detection, IoU is used to determine the accuracy of the detection by comparing how closely the predicted bounding box aligns with the ground truth bounding box. IoU is calculated as the area of overlap or intersection between the ground truth and predicted bounding boxes, divided by the area of their union. It is mathematically defined as

$$IoU = \frac{Area\ of\ Intersection}{Area\ of\ Union} \qquad (3.5)$$

IoU values range from 0 to 1, where a value of 1 indicates a perfect overlap and 0 indicates no overlap. Figure 3.7-1 displays the mathematical formulation to calculate IOU

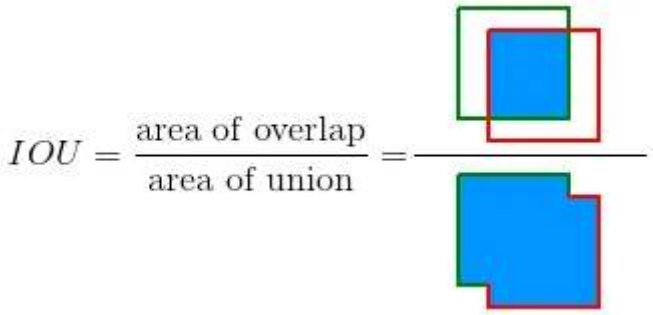

Figure 1.61.6-1 Computing the Intersection over Union

Confusion Matrix: The confusion matrix is a table used to evaluate the performance of a classification model in object detection tasks, including pedestrian detection. It consists of four values: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). In terms of pedestrian detection:

- True Positive (TP): A true positive is when the model correctly identifies a pedestrian, i.e., a detection for which the Intersection over Union (IoU) is greater than or equal to a threshold α (IoU ≥ α). It indicates that the predicted bounding box sufficiently overlaps with the ground truth bounding box.
- False Positive (FP): A false positive is when the model incorrectly identifies a non-pedestrian as a pedestrian, i.e., a detection for which the Intersection over Union (IoU) is less than the threshold α (IoU < α). It means that the predicted bounding box does not align well with any ground truth bounding box.
- False Negative (FN): A false negative is when the model fails to identify a pedestrian. It implies an actual pedestrian instance that is missed by the classifier.
- True Negative (TN): In the context of object detection, true negative is often not applicable, because there are many possible bounding boxes in an image that should not be detected. However, in a general sense, a true negative would imply that the model correctly identifies that there is no pedestrian and does not make a detection. In object detection tasks, TN would include all possible incorrect detections that were not made.

The confusion matrix is fundamental for calculating various performance metrics, including precision and recall. IoU is especially important as it helps to quantify the quality of the bounding boxes predicted by the model [44].

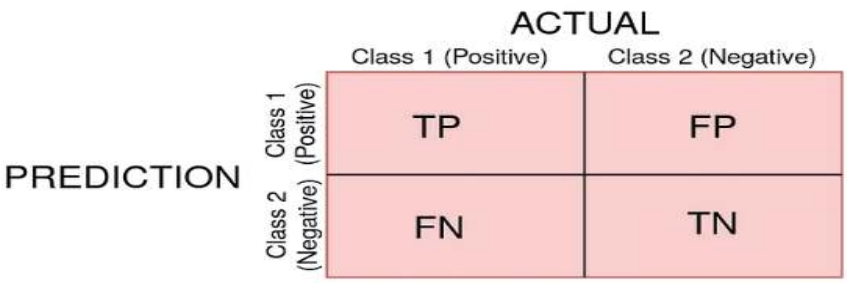

Figure 1.61.6-2 Confusion matrix

Precision (P): Precision uses values from the confusion matrix. It is the ratio of true positive detections to the sum of true positive and false positive detections. High precision means that the model correctly identifies a high proportion of actual pedestrians. Precision is defined as

$$Precision = \frac{TP}{TP+FP} \qquad (3.6)$$

Recall (R): Recall, also known as sensitivity or the true positive rate, also employs values from the confusion matrix. It is the ratio of true positive detections to the sum of true positives and false negatives. High recall indicates that the model identifies most of the pedestrians. The recall is defined as

$$Sensitivity = \frac{TP}{TP + FN} \qquad (3.7)$$

Mean Average Precision (mAP): mAP is a popular metric in object detection tasks. It calculates the average precision at different recall levels, effectively summarizing the precision-recall curve. Another important aspect of mAP is the Intersection over Union (IoU), which measures the overlap between the ground truth bounding box and the predicted bounding box. A detection is considered true positive if the IoU is above a certain threshold. Typically, mAP is evaluated at different IoU thresholds, such as 0.5 or ranging from 0.5 to 0.95 in increments of 0.05.

F1 Score: The F1 Score is the harmonic mean of precision and recall. It is particularly useful when the class distribution is uneven. It considers both false positives and false negatives and is defined as

$$F - Score = 2 * \frac{P recision * Recall}{Precision + Recall} \qquad (3.8)$$

Frames Per Second (FPS): FPS is a measure of the model's inference speed. It is especially important in real-time applications such as pedestrian detection in video

streams. Higher FPS means the model can process more frames in a given amount of time, which is vital for timely reactions in practical applications.

Precision-Recall (PR) Curve: The PR curve is a plot that shows the trade-off between precision and recall for different threshold values. A high area under the curve represents both high recall and high precision.

Chapter 4
Experiments and Results

4.1 Experiments and Results

In this chapter, we present the experiments conducted to evaluate the performance of the pedestrian detection models developed using YOLOv7 with anchors calculated by K-means clustering and YOLOv7 with anchors calculated by Fuzzy C-means clustering. The goal is to compare and analyze the effectiveness of these models using various performance metrics. The chapter is structured into sections that include the experimental setup, dataset description, results, and analysis.

- **Computing Environment**

The experiments were conducted on Google Colab Pro, which is an enhanced version of Google Colab, offering increased computing resources such as higher memory, faster GPUs, and longer runtime compared to the standard version. The decision to subscribe to the Pro version was driven by the need for robust computing capabilities to handle the training and evaluation of large models, as well as to efficiently work with the substantial CrowdHuman dataset.

To access Google Colab, a personal laptop with the following specifications was used:

- Processor: Intel(R) Core (TM) i7-8565U CPU @ 1.80GHz, 1.99 GHz
- Memory: 8.00 GB (7.89 GB usable)
- Operating System: 64-bit operating system, x64-based processor
- Windows 10.

Figure 4.1-1 Display laptop specification used to access Google Colab

Google Colab operates as a cloud-based virtual machine, equipped with an NVIDIA Tesla GPU, which is essential for accelerating the training process. The underlying operating system is Linux-based, and it provides access to a suite of software libraries and tools such as Python 3.8, TensorFlow 2.5, and OpenCV, which are integral for running the YOLOv7 algorithm. An essential feature of Google Colab is its seamless integration with Google Drive, allowing for easy access to datasets and storage of results. For this experiment, the CrowdHuman dataset was initially uploaded to Google Drive. Automation scripts were developed to facilitate the transfer of this dataset from Google Drive to the local disk space allocated by Colab. This was critical as it allowed for faster data loading times during training. In summary, Google Colab Pro served as a potent and accessible platform for executing experiments. With its powerful GPUs and integration with Google Drive, it enabled efficient model training, validation, and evaluation processes, without the constraints that would be encountered on a personal computer.

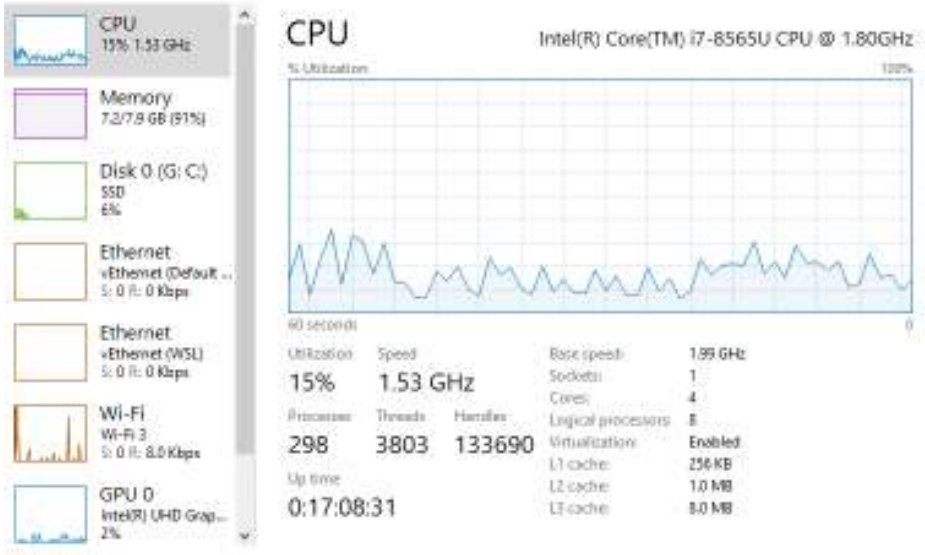

Figure 4.1-1 Laptop specification used to access Google Colab

- **Model Configurations**

The YOLOv7 algorithm was used as the backbone for both pedestrian detection models. For transfer learning, pre-trained weights were loaded, and the networks were fine-tuned using the CrowdHuman dataset. The learning rate, batch size, and other hyperparameters were selected after preliminary experiments to find the optimal configuration for the models.

4.2 Deep Learning Practical Experiment

4.2.1 Pedestrian Detection for benchmark dataset

The primary motivation behind evaluating our model on benchmark datasets is to understand and quantify its performance in real-world scenarios. The process of pedestrian detection is complex, especially in crowded environments where occlusions and varying scales are common. Given this context, it becomes imperative to test and validate the robustness of our algorithm on widely-recognized benchmark datasets. In this section, we delve into the experimental results of our pedestrian detection models, which were meticulously developed by harnessing the power of the YOLOv7 architecture through transfer learning techniques. These models were pre-trained on the CrowdHuman dataset, which is characterized by a diverse set of images that simulate various real-life conditions. To further augment the performance of our model in terms of anchor box optimization, we adopted clustering algorithms, specifically k-means and fuzzy c-means. The essence of employing clustering algorithms lies in their ability to group data points, in this context, bounding box dimensions, such that the distance between data points in the same cluster is minimized. This ensures a higher Intersection over Union (IoU) score, which is critical for accurate object detection. We aimed to assess the performance of our models on two revered benchmark datasets – CityPersons [45] and the Caltech Pedestrian

dataset [46]. These datasets were specifically chosen as they both encompass intricacies typical of crowded environments, albeit different. The CityPersons dataset predominantly captures urban pedestrian scenarios, while the Caltech Pedestrian dataset is extensive, containing videos taken from a vehicle, thus simulating conditions akin to autonomous driving applications. The subsequent subsections are dedicated to an in-depth analysis of our model's performance on each of the datasets. For a comprehensive evaluation, we employ metrics such as Mean Average Precision (mAP) and Frames Per Second (FPS) to gauge the accuracy and speed of the models respectively. Finally, we present a comparative analysis of the performance of our models (YOLOv7+k-means and YOLOv7+fuzzy c-means) on the two datasets. Through a series of tables and charts, we will dissect the strengths and limitations of each approach, while also understanding the suitability of the models in different applications. Table 4.2-1 shows the comparative results for our models when tested on benchmark datasets.

Model	**Dataset**	**mAP (%)**	**FPS**
YOLOv7 + k-means	CityPersons	62.1%	53
YOLOv7 + fuzzy c-means	CityPersons	65.3%	53
YOLOv7 + k-means	Caltech Pedestrian	64.6%	53
YOLOv7 + fuzzy c-means	Caltech Pedestrian	67.9%	53

Table 4.2-1 Comparative Evaluation Metrics

After training, the first model yielded a measured FPS (frames per second) of 53, while the second model achieved an identical FPS value of 53. These results indicate that utilizing the same pre-trained model with the same architecture has no significant effect

on the FPS. Moreover, the consistent FPS performance observed in both models further supports the notion that the chosen architecture remains resilient to model-specific variations. It is noteworthy that these measurements were obtained using the same hardware specifications, which ensures a fair comparison and strengthens the conclusion that the FPS remains unaffected by the model variations when employing the same architecture.

This section's insight facilitates understanding the capabilities and potential applications of our models in pedestrian detection, which is a cornerstone in systems such as crowd management, surveillance, and autonomous vehicles.

4.2.2 Results of Yolov7 & K-means

In the initial approach to developing the pedestrian detection model, the Building Models Phase utilized K-means clustering to calculate anchors during data preprocessing, which was essential for optimizing object detection. The YOLOv7 model, celebrated for its accuracy and speed in object detection, was employed as a pre-trained foundation. Leveraging transfer learning, YOLOv7 was meticulously fine-tuned on the CrowdHuman dataset for the specific task of pedestrian detection. The metrics gleaned from this approach were highly promising. The model achieved a mean Average Precision of 87.6%, which is indicative of its effectiveness in identifying pedestrians accurately. The F1-score, a harmonic mean of precision and recall, was also robust at 83.7%, signifying a well-balanced trade-off between precision and recall. Notably, the model attained a precision of 87% and a recall of 80%, demonstrating its proficiency in accurately detecting pedestrians while minimizing false negatives. Table 4.2-1 shows these training results.

Model Name	Mean Average Precision (mAP)	F1-score	Precision	Recall
YOLOV7 + K-Means Clustering	87.6%	83.7%	87%	80%

Table 4.2-2 Result for training Yolov7 with k-means (anchors)

Figure 4.2-1 displays results for Box loss, Objectness loss, Classification loss, Precision, Recall, MAP@0.5, MAP@0.5:0.95.

The confusion matrix further substantiated the model's capabilities; 85% of true positive pedestrian detections were correct, while only 15% of true pedestrians were falsely identified as the background as shown in Figure 4.2-2. Interestingly, the model displayed a perfect score in avoiding false positives where the background was wrongly classified as a pedestrian. These results showcase the potential of employing K-means clustering for anchor calculations in conjunction with a re-trained YOLOv7 model for efficient pedestrian detection

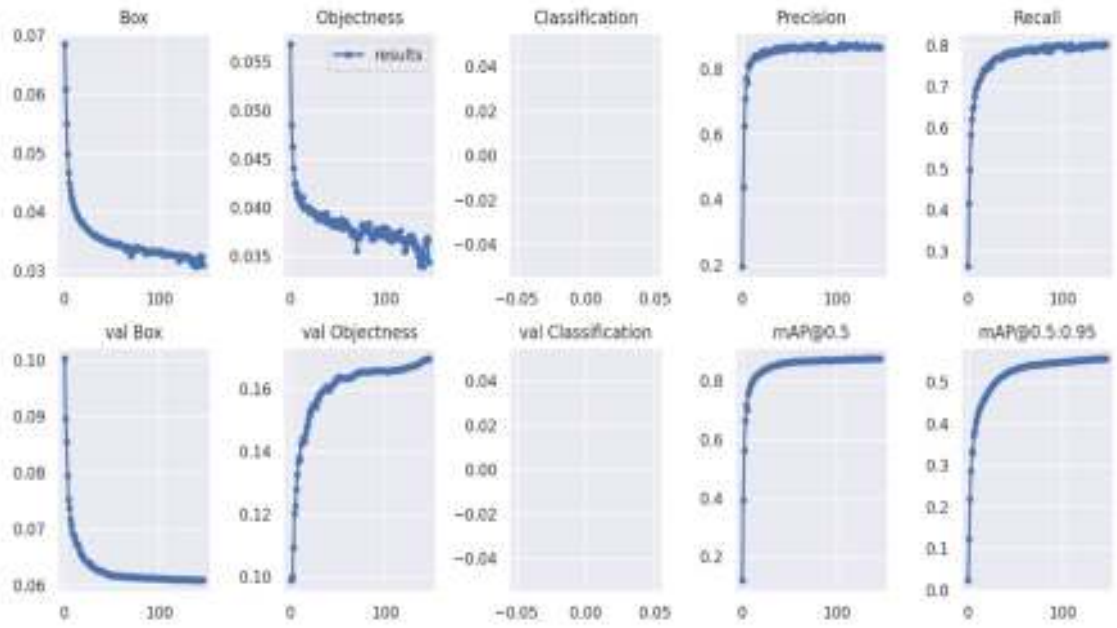

Figure 4.2-1 Box loss, Objectness loss, Classification loss, Precision , Recall, MAP@0.5, MAP@0.5:0.95

The Precision-Recall curve for our pedestrian detection model in crowded environments demonstrates an encouraging performance, particularly in terms of recall. The curve is close to the top-right corner indicating that the model is proficient in identifying a large proportion of actual pedestrians amidst the crowd. This is particularly vital in crowded settings where it is crucial to detect as many pedestrians as possible to ensure their safety and manage crowd dynamics efficiently. However, as the recall increases, there might be a trade-off with precision, indicating that the model may sometimes falsely identify non-pedestrian objects as pedestrians.

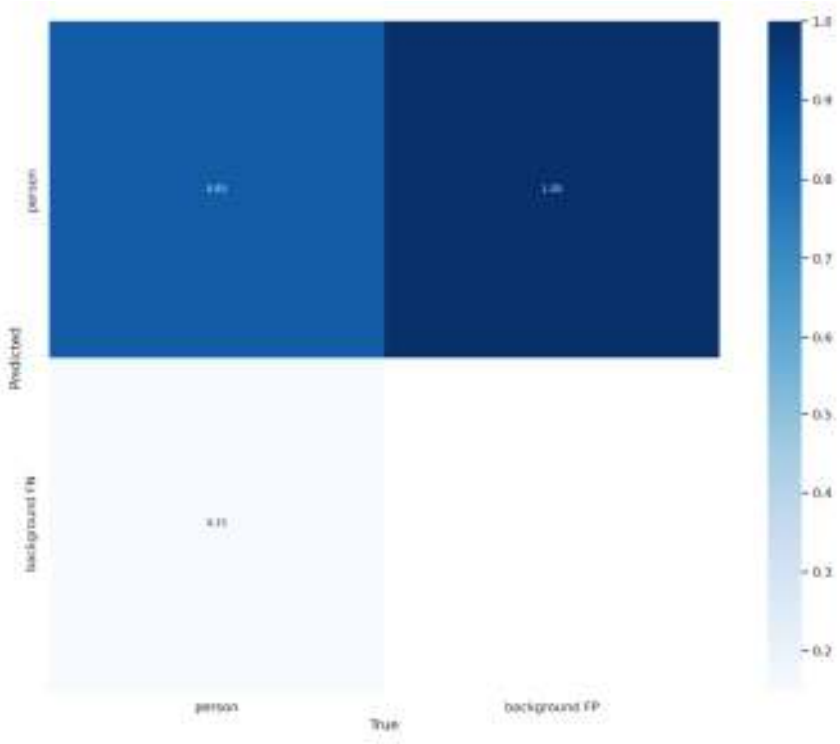

Figure 4.2-2 First model confusion matrix

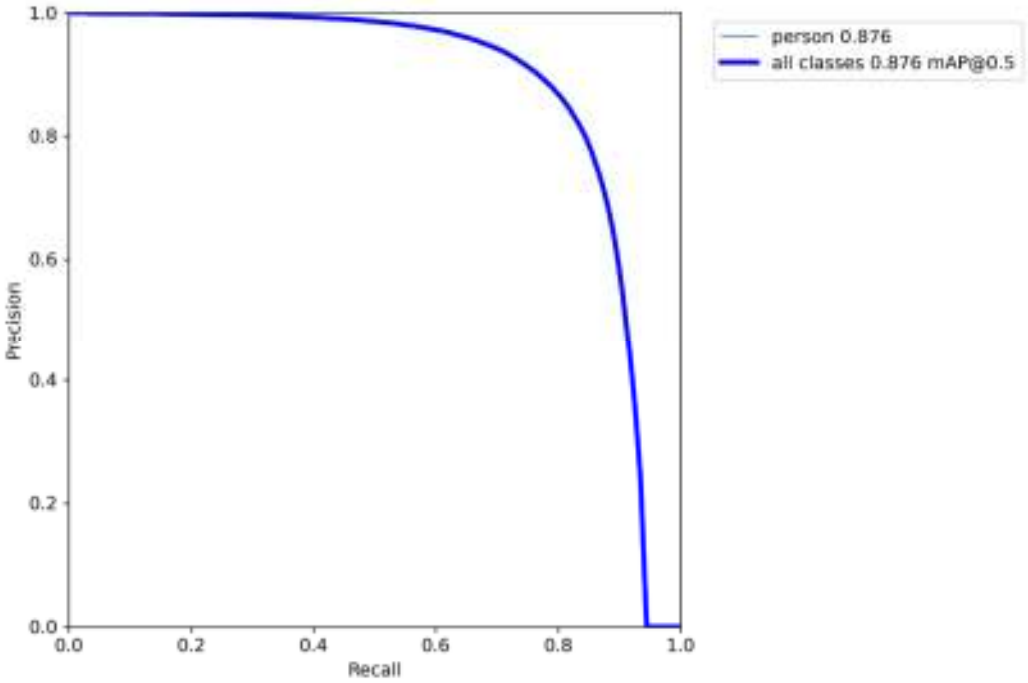

Figure 4.2-3 P_R_Curve

In the context of pedestrian detection in crowded environments, achieving high recall is generally favorable, but it is also important to reduce the number of false positives for more accurate crowd analysis and management.

4.2.3 Results of Yolov7 & Fuzzy C-means

In the second approach of the Building Models Phase, anchors were fine-tuned using the Fuzzy C-means clustering algorithm, as opposed to the K-means clustering used in the first approach. By leveraging Fuzzy C-means clustering, the model could potentially achieve a more nuanced understanding of the dimensions and shapes of pedestrians within the CrowdHuman dataset. Utilizing YOLOv7, a pre-trained model renowned for its accuracy and speed in object detection, and the power of transfer learning, the model was further fine-tuned specifically for pedestrian detection. The results obtained through this second approach were impressive. The model attained a mean Average Precision (mAP) of 88.1%, indicating a high level of accuracy in detecting pedestrians across different

levels of confidence thresholds. Additionally, the F1-score, which is the harmonic mean of precision and recall, reached an encouraging value of 84.2%. In terms of precision and recall, the model achieved values of 87.5% and 81.2% respectively, signifying its ability to accurately identify pedestrians while minimizing false negatives. Table 4.2-2 shows these training results

Model Name	Mean Average Precision (mAP)	F1-score	Precision	Recall
YOLOV7 + Fuzzy C-Means	88.1%	84.2%	87.5%	81.2%

Table 4.2-3 Result for training Yolov7 with Fuzzy C-means (anchors)

Figure 4.2-4 display results for Box loss, Objectness loss, Classification loss, Precision , Recall, MAP@0.5, MAP@0.5:0.9

Analyzing the confusion matrix further reveals the model's efficacy; it yielded a value of 0.86 for correctly predicting pedestrians when they were indeed present and had an error rate of 0.14 for falsely predicting background when there were pedestrians. This approach showcases the robustness and precision of integrating Fuzzy C-means clustering with YOLOv7 for pedestrian detection in crowded environments. This information serves as vital input for comparing and evaluating the two distinct approaches employed in this study.

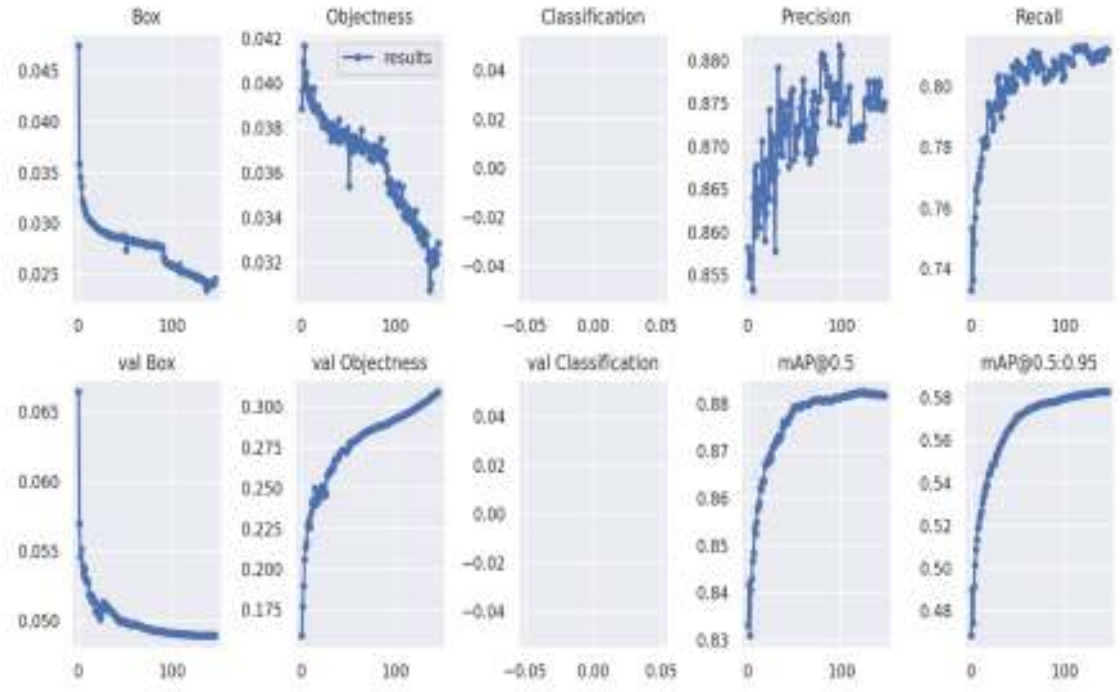

Figure 4.2-4 Box loss, Objectness loss, Classification loss, Precision , Recall, MAP@0.5, MAP@0.5:0.95

The Precision-Recall curve, a graphical representation of the trade-off between precision and recall, exhibits promising results for our pedestrian detection model in crowded environments. The curve's proximity to the top-right corner signifies that the model is highly adept at detecting a significant proportion of pedestrians in the crowd, which is reflected in the high recall values. In bustling settings, it's essential to identify as many pedestrians as possible for their safety and efficient crowd management. Nevertheless, as the curve reflects, an increase in recall may be accompanied by a decrease in precision.

Figure 4.2-5 Second model confusion matrix

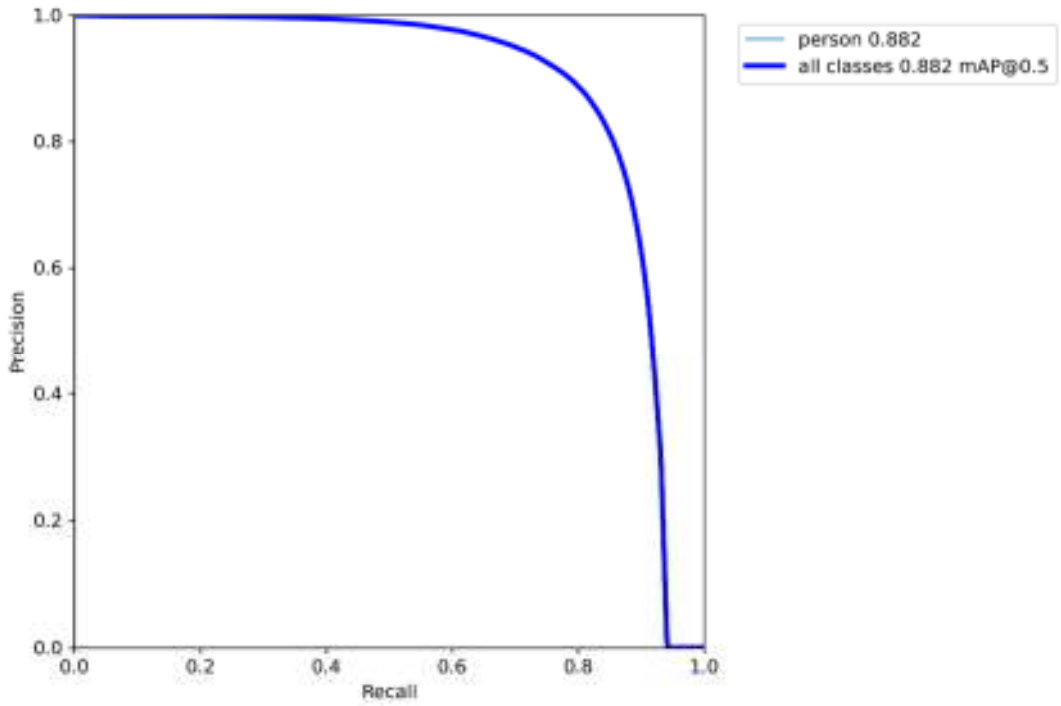

Figure 4.2-6 P_R_Curve

This trade-off suggests that the model might occasionally misclassify non-pedestrian objects as pedestrians. While the high recall is desirable for detecting pedestrians in crowded areas, it's also crucial to mitigate the number of false positives to ensure more precise crowd analysis and effective management.

The evaluation of the pedestrian detection models was carried out using several critical metrics, including mean Average Precision (mAP), F1-score, Precision, and Recall. These metrics provide an understanding of how well the models perform in identifying pedestrians accurately and their ability to maintain a balance between precision and recall. Below is a tabulated summary of the performance metrics for two models: YOLOV7 with k-means clustering (YOLOV7+kmeans) and YOLOV7 with fuzzy c-means clustering (YOLOV7+fuzzy c-means).

Model Name	Mean Average Precision (mAP)	F1-score	Precision	Recall
YOLOV7+kmeans	87.6%	83.7%	87%	80%
YOLOV7+fuzzy c-means	88.1%	84.2%	87.5%	81.2%

Table 4.2-4 Our two models' training results

Upon reviewing the performance metrics, it is evident that both models have exhibited commendable results, albeit with slight differences. The YOLOV7+fuzzy c-means model shows a marginally higher mean Average Precision at 88.1% compared to 87.6% for YOLOV7+kmeans. This indicates that the inclusion of fuzzy c-means clustering slightly enhances the model's effectiveness in recognizing pedestrians accurately across various confidence thresholds.

Moreover, the F1-score, which evaluates the harmonic mean of precision and recall, is also somewhat higher in the YOLOV7+fuzzy c-means model at 84.2% compared to the 83.7% of the YOLOV7+kmeans model. This suggests that fuzzy c-means clustering provides a slightly better balance between precision and recall.

In terms of precision, both models perform almost similarly, with only a 0.5% difference. However, when it comes to recall, the YOLOV7+fuzzy c-means model shows an improvement of 1.2% over the YOLOV7+kmeans model, implying that the former is more adept at minimizing false negatives.

In conclusion, while both models display strong performances, the YOLOV7+fuzzy c-means model appears to have a slight edge over the YOLOV7+kmeans model in terms of mean Average Precision, F1-score, and recall. This suggests that integrating fuzzy c-means clustering could be advantageous for enhancing pedestrian detection accuracy and reliability.

4.2.4 Inference Results

In the section titled "Inference Results," a critical element of the analysis is presented where the developed models - YOLOv7 integrated with anchors calculated via K-Means and YOLOv7 integrated with anchors calculated via Fuzzy C-Means - are tested under practical conditions. For this, a set of sample images, which do not have prior annotations, are processed through both models to detect pedestrians. Bounding boxes, which indicate the areas in the images where pedestrians have been identified, are superimposed on the images. These images serve as a tangible demonstration of the models' performance in detecting pedestrians. Through the visualizations provided by these annotated images, a critical assessment of the efficacy of each model in identifying pedestrians is facilitated.

Furthermore, it enables a comparison between the two models, thereby allowing for an evaluation of the relative merits of utilizing K-Means and Fuzzy C-Means for anchor box calculation. Analysis of the bounding boxes, coupled with the consideration of confidence scores and positioning, grants an understanding of the models' real-world application and their practical effectiveness in detecting pedestrians in dense settings. This section, therefore, plays an essential role in empirically validating the developed models on real-world images that lack prior annotations. Such a real-world evaluation is invaluable for understanding the operational capabilities and limitations of the models and is vital in achieving the research objectives. The insights derived from the practical application of the models contribute significantly to the overall conclusions and contributions of this thesis.

Figure 4.2-7 Original images with annotations

This image Figure 4.2-7, taken from the validation dataset, provides a real-world example of pedestrian detection, complete with original annotations that highlight regions of interest. The annotations, depicted as bounding boxes, offer valuable insights into how models will interpret and engage with authentic data during the validation process.

Figure 4.2-8 K-means model (Predict)

In the second image Figure 4.2-8, we present the same scene from the validation dataset but with bounding boxes predicted by our custom model that incorporates YOLOv7 with K-means clustering. This side-by-side comparison with the original annotations allows for an in-depth analysis of how effectively our model has been trained to detect pedestrians and draw the predicted bounding box.

Figure 4.2-9 Fuzzy C means (Predict)

Showcasing the same scene from the validation dataset, the third image Figure 4.2-9 is adorned with bounding boxes that are the predictions of our tailored model which synergizes YOLOv7 with fuzzy c-means clustering to optimize anchor selection. Setting this side by side with the original annotations, we are furnished with a detailed landscape for assessing the adeptness of our model in pinpointing pedestrians and delineating the predicted bounding boxes. To truly demonstrate the adaptability of our model, let's consider a completely new image, one unseen during the training or validation stages. This image contains numerous pedestrians, but it lacks any pre-existing bounding boxes. This scenario will put our YOLOv7 model's generalization capabilities to the test, as it must identify and draw bounding boxes around the pedestrians in this entirely novel context. Figure 4.2-10 displays an unseen image.

Figure 4.2-10 Original Image without annotations

After feeding this image into our first model, which integrates YOLOv7 with K-means clustering, the model was able to predict and accurately draw bounding boxes around

each pedestrian present in the image. This showcases the adaptability and applicability of our model to real-world scenarios it has not encountered before. Figure 4.2-11 serves as a testament to the generalizability of our YOLOv7 + K-means model in detecting pedestrians in an unfamiliar environment.

Figure 4.2-11 K Means (Predict)

Similarly, when the same unseen image was input into our second model, which couples YOLOv7 with Fuzzy C-means clustering, the model succeeded in predicting and outlining bounding boxes around each pedestrian. This exemplifies the model's adeptness and flexibility in adapting to new, real-world situations that it hasn't been exposed to during training. Figure 4.2-12 illustrates the remarkable performance and generalization capabilities of our YOLOv7 + Fuzzy C-means model in identifying pedestrians in previously unencountered settings.

Figure 4.2-12 Fuzzy C Means (Predict)

4.3 Challenges and Limitation

In this chapter, we will explore the challenges and limitations encountered during the utilization of transfer learning for training a pedestrian detection model using YOLOv7 with the CrowdHuman dataset. Our focus will primarily be on the challenges associated with computational resources and time, as well as the intricacies involved in hyper parameter tuning.

- **GPU Processing Power**

Training a deep learning model such as YOLOv7 is computationally intensive. The complexity of the model and the sheer volume of the CrowdHuman dataset necessitate the use of high-performance GPUs. Unfortunately, limited access to advanced GPUs can drastically prolong the training phase. Moreover, the availability of such GPUs is often a

privilege that may not be readily accessible to all researchers, particularly those in resource-constrained settings.

- **Memory Limitations**

Beyond processing power, memory is another critical resource. Efficient training mandates that both the model and dataset be accommodated within the GPU's memory. However, the size of the CrowdHuman dataset may exceed the available memory. Consequently, researchers might be forced to make compromises such as reducing the batch size or image resolution. While such measures alleviate memory issues, they can detrimentally affect the performance of the model.

- **Training Duration**

Time is an invaluable asset in the realm of research. The training of a sophisticated model like YOLOv7 on an extensive dataset like CrowdHuman can span an extended period, ranging from several hours to days or even weeks. Such durations may not be feasible for all researchers, particularly those with impending deadlines or limited access to computational resources.

- **Exploration of Parameter Space**

Hyperparameters play an instrumental role in the training of deep learning models. The task of identifying an optimal set of hyperparameters is non-trivial and often likened to finding a needle in a haystack. The breadth of hyperparameters, including learning rates, batch sizes, and regularization factors, contributes to an extensive parameter space that is prohibitively time-consuming to exhaustively explore.

- **Balancing Between Overfitting and Underfitting**

The selection of hyperparameters has a bearing on the model's capacity to generalize. Improper selection can result in overfitting, where the model becomes too tailored to the training data, or Underfitting, where it fails to capture the underlying trends. Striking a balance is imperative but challenging.

In summary, the training of pedestrian detection models using transfer learning, YOLOv7, and the CrowdHuman dataset is fraught with challenges. These range from the practical constraints of computational resources and time to the more subtle and nuanced challenges involved in hyper parameter tuning. Addressing these challenges is vital for the effective and efficient development of pedestrian detection models. Future research could explore techniques for optimizing hyperparameters more efficiently or methods for reducing the computational burden of training sophisticated models.

Chapter 5
Conclusion and Future Work

5.1 Conclusion

In conclusion, this study embarked on a journey to develop an efficient and reliable pedestrian detection model specifically geared toward crowded environments. Two approaches were explored, employing different clustering algorithms to optimize anchor bounding boxes in conjunction with the YOLOv7 architecture. The first approach utilized K-means clustering to calculate anchors. The model achieved a mean Average Precision (mAP) of 87.6%, signifying its effectiveness in pedestrian detection. The F1 score stood at 83.7%, indicating a well-balanced trade-off between precision (87%) and recall (80%). The confusion matrix revealed that 85% of true positive pedestrian detections were correct, while only 15% of true pedestrians were falsely identified as the background. Notably, the model displayed no instances of false positives where the background was wrongly classified as a pedestrian. Conversely, the second approach employed Fuzzy C-means clustering for anchor calculations. It achieved a slightly higher mAP of 88.1% and an F1-score of 84.2%. Precision and recall values were marginally superior at 87.5% and 81.2% respectively. The confusion matrix demonstrated that this approach had a true positive rate of 86% and a false negative rate of 14%. Comparatively, the second approach utilizing Fuzzy C-means clustering exhibited a slight edge over the K-means-based approach in all key performance metrics including mAP, F1-score, precision, and recall. The enhanced performance could be attributed to Fuzzy C-means clustering's ability to provide a more nuanced understanding of the dimensions and shapes of pedestrians, which is critical in crowded environments. Based on the findings, it is recommended to

adopt the second approach employing Fuzzy C-means clustering in conjunction with the YOLOv7 model for pedestrian detection in crowded environments. The marginal improvements in accuracy and the balanced trade-off between precision and recall could be pivotal in real-world applications where every percentage point counts in ensuring public safety and effective crowd management. However, it is important to recognize that while the improvements are noteworthy, there is still room for further optimization and research. Future endeavors might explore hybrid clustering techniques, investigate different network architectures, or experiment with ensemble methods to potentially bolster the performance even further. As technology evolves, continuous refinement and adaptation will be essential in addressing the ever-changing dynamics of crowded environments. Through dedication to innovation and development, pedestrian detection systems can become an integral component in ensuring public safety in smart city ecosystems.

5.2 Future Work and Recommendations

In this section, we discuss potential directions for future research in pedestrian detection. The scope of the future work is to expand upon the research presented in this thesis, aiming for improvements in detection accuracy, robustness, and applicability in real-world scenarios.

- **Expanding the Annotation Space**

As an initial focus, the expansion of the annotation space holds promises. By employing head and full-body annotations, we can potentially enhance the capability of pedestrian detection models. Head annotations may prove invaluable, especially in instances where the full body is occluded or not entirely visible. Conversely, full-body annotations could

provide additional context when the head is not distinguishable. By training models with this augmented information, the detection systems may demonstrate increased versatility and robustness. Furthermore, exploring combinations of different annotations might uncover synergies that enhance detection performance across various scenarios.

- **Anchor Box Discovery through DBSCAN Applied to Bounding Boxes**

In forthcoming research, the potential of using DBSCAN for anchor box selection will be explored. This density-based algorithm's ability to handle outliers and capture varying object densities within bounding box annotations offers promise for more accurate anchor assignment. The combination of DBSCAN with existing clustering techniques could further elevate the precision of object detection models, particularly in scenarios involving complex scenes and diverse object scales.

- **Exploration of Object Detection Models**

After expanding the annotation space, it is advantageous to venture into the utilization of cutting-edge object detection models such as YOLO-NAS [47] and YOLOv8[48].

Utilizing YOLO-NAS:

YOLO-NAS (Neural Architecture Search) is an advanced variant of the YOLO family of object detectors. It employs neural architecture search to automatically optimize the network architecture, striving for an optimal trade-off between accuracy and efficiency [47]. Involvement with YOLO-NAS for pedestrian detection, particularly when trained on the CrowdHuman dataset with head and full body annotations, could be beneficial.

Utilizing YOLOV8:

YOLOv8, the latest addition to the YOLO series, is expected to outperform its predecessors in both accuracy and speed [48]. Therefore, it is worthwhile to experiment with YOLOv8 for pedestrian detection using the CrowdHuman dataset.

Bibliography

[1] *Introduction to Crowd Science - G. Keith Still - Google Books*. .

[2] Szeliski, *Computer Vision Richard Szeliski Algorithms and Applications*. 2022.

[3] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, 2012, doi: 10.1109/TPAMI.2011.155.

[4] "Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville - كتب Google." https://books.google.ps/books?hl=ar&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=Goodfellow,+I.,+Bengio,+Y.,+%26+Courville,+A.+(2016).+Deep+Learning.+MIT+Press.+IEEE+citation+format&ots=MNU4dvsFVT&sig=Ps5wtE1wDdlJ23mJRyA9zxvuSuc&redir_esc=y#v=onepage&q&f=false (accessed Jun. 10, 2023).

[5] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010, doi: 10.1109/TKDE.2009.191.

[6] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," pp. 1–15, 2022, [Online]. Available: http://arxiv.org/abs/2207.02696.

[7] H. M. Jeon, V. D. Nguyen, and J. W. Jeon, "Pedestrian Detection Based on Deep Learning," *IECON Proc. (Industrial Electron. Conf.*, vol. 2019-Octob, no. September, pp. 144–149, 2019, doi: 10.1109/IECON.2019.8927417.

[8] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2015, vol. 1, pp. 886–893, doi: 10.1109/CVPR.2005.177.

[9] M. M. Taye, "Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions," *Computation*, vol. 11, no. 3, p. 52, 2023, doi: 10.3390/computation11030052.

[10] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective Search for Object Recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Sep. 2013, doi: 10.1007/s11263-013-0620-5.

[11] "Boosting Algorithm | Boosting Algorithms in Machine Learning." https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-

algorithms-machine-learning/ (accessed Jun. 11, 2023).

[12] "Understanding Support Vector Machine Algorithm From Examples Along With Code | PDF | Support Vector Machine | Statistical Classification." https://www.scribd.com/document/386502572/Analyticsvidhya-com-Understanding-Support-Vector-Machine-Algorithm-From-Examples-Along-With-Code# (accessed Jun. 11, 2023).

[13] B. Leibe, E. Seemann, and B. Schiele, "Pedestrian detection in crowded scenes," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, 2005, vol. 1, pp. 878–885.

[14] M. J. Flores Calero, M. Aldas, J. Lazaro, A. Gardel, N. Onofa, and B. Quinga, "Pedestrian detection under partial occlusion by using logic inference, HOG and SVM," *IEEE Lat. Am. Trans.*, vol. 17, no. 9, pp. 1552–1559, 2019, doi: 10.1109/TLA.2019.8931190.

[15] C. Wojek, S. Walk, S. Roth, and B. Schiele, "Monocular 3D scene understanding with explicit occlusion reasoning," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1993–2000, 2011, doi: 10.1109/CVPR.2011.5995547.

[16] M. Mathias, R. Benenson, R. Timofte, and L. Van Gool, "Handling occlusions with franken-classifiers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1505–1512.

[17] H. Fukui, T. Yamashita, Y. Yamauchi, H. Fujiyoshi, and H. Murase, "Pedestrian detection based on deep convolutional neural network with ensemble inference network," *IEEE Intell. Veh. Symp. Proc.*, vol. 2015-Augus, pp. 223–228, 2015, doi: 10.1109/IVS.2015.7225690.

[18] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," *Adv. Neural Inf. Process. Syst.*, vol. 4, no. January, pp. 3320–3328, 2014.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, vol. 20, no. 11, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[20] T. Y. Lin *et al.*, "Microsoft COCO: Common objects in context," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes*

Bioinformatics), vol. 8693 LNCS, no. PART 5, pp. 740–755, 2014, doi: 10.1007/978-3-319-10602-1_48.

[21] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1717–1724, 2014, doi: 10.1109/CVPR.2014.222.

[22] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, vol. 2016-Septe, pp. 3431–3440, doi: 10.1109/CVPR.2015.7298965.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, vol. 45, no. 8, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[28] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 6517–6525, 2017, doi: 10.1109/CVPR.2017.690.

[29] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv Prepr. arXiv1804.02767*, 2018.

[30] W. Liu *et al.*, “SSD: Single shot multibox detector,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016, doi: 10.1007/978-3-319-46448-0_2.

[31] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.

[32] C. Ning, L. Menglu, Y. Hao, S. Xueping, and L. Yunhong, “Survey of pedestrian detection with occlusion,” *Complex Intell. Syst.*, vol. 7, no. 1, pp. 577–587, 2021, doi: 10.1007/s40747-020-00206-8.

[33] R. Girshick, “Fast R-CNN,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, pp. 1440–1448, 2015, doi: 10.1109/ICCV.2015.169.

[34] X. Du, M. El-Khamy, J. Lee, and L. Davis, “Fused DNN: A deep neural network fusion approach to fast and robust pedestrian detection,” *Proc. - 2017 IEEE Winter Conf. Appl. Comput. Vision, WACV 2017*, pp. 953–961, 2017, doi: 10.1109/WACV.2017.111.

[35] R. Stewart, M. Andriluka, and A. Y. Ng, “End-to-end people detection in crowded scenes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2325–2333.

[36] M. Xu, Z. Wang, X. Liu, L. Ma, and A. Shehzad, “An Efficient Pedestrian Detection for Realtime Surveillance Systems Based on Modified YOLOv3,” *IEEE J. Radio Freq. Identif.*, vol. 6, pp. 972–976, 2022, doi: 10.1109/JRFID.2022.3212907.

[37] S. Shao *et al.*, “CrowdHuman: A Benchmark for Detecting Human in a Crowd,” pp. 1–9, 2018, [Online]. Available: http://arxiv.org/abs/1805.00123.

[38] A. K. Jain, “Data clustering: 50 years beyond K-means,” *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010, doi: 10.1016/j.patrec.2009.09.011.

[39] M. J, “Some Methods for Classification and Analysis of MultiVariate Observations,” *Proc Berkeley Symp. Math. Stat. Probab.*, vol. 5, no. 1, pp. 281–297, 1965.

[40] R. Suganya and R. Shanthi, “Fuzzy C-Means Algorithm-A Review,” *Int. J. Sci. Res. Publ.*, vol. 2, no. 11, pp. 2250–3153, 2012, [Online]. Available: www.ijsrp.org.

[41] J. C. Dunn, “A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters,” *J. Cybern.*, vol. 3, no. 3, pp. 32–57, 1973, doi: 10.1080/01969727308546046.

[42] U. M. D. E. C. D. E. Los, *Pattern Recognition With Fuzzy Objective Function Algorithms*. .

[43] Ö. Kaya, M. Y. Çodur, and E. Mustafaraj, “Automatic Detection of Pedestrian Crosswalk with Faster R-CNN and YOLOv7,” *Buildings*, vol. 13, no. 4, pp. 1–17, 2023, doi: 10.3390/buildings13041070.

[44] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006, doi: 10.1016/j.patrec.2005.10.010.

[45] M. Cordts *et al.*, “The Cityscapes Dataset for Semantic Urban Scene Understanding,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 3213–3223, 2016, doi: 10.1109/CVPR.2016.350.

[46] P. Dollar, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: A benchmark,” pp. 304–311, 2010, doi: 10.1109/cvpr.2009.5206631.

[47] “YOLO-NAS by Deci Achieves State-of-the-Art Performance on Object Detection Using Neural Architecture Search.” https://deci.ai/blog/yolo-nas-object-detection-foundation-model/ (accessed Jun. 17, 2023).

[48] D. Reis, J. Kupec, J. Hong, and A. Daoudi, “Real-Time Flying Object Detection with YOLOv8,” 2023, [Online]. Available: http://arxiv.org/abs/2305.09972.

الملخص

يعد اكتشاف المشاة، خاصة في البيئات المزدحمة، مهمة محورية للعديد من التطبيقات مثل المراقبة والمركبات المستقلة وإدارة الحشود. تعتمد دقة وكفاءة أنظمة الكشف عن المشاة بشكل كبير على المنهجيات المستخدمة.

تبحث هذه الأطروحة وتقيم تطوير نماذج اكتشاف المشاة القائمة على التعلم العميق، مع التركيز بشكل خاص على تحسين صندوق التثبيت وتكامل خوارزميات التجميع. في المرحلة الأولية، تم تكييف YOLOv7، وهو نموذج متطور لاكتشاف الأشياء معروف بدقته وسرعته، لاكتشاف المشاة من خلال نقل التعلم على مجموعة بيانات CrowdHuman .يتم استخدام طريقتين متميزتين لتحسين صندوق التثبيت: K-Means و Fuzzy C-Means clustering تهدف هذه الأساليب إلى ضبط مربعات التثبيت لتكون أكثر تمثيلاً لأبعاد وأشكال المشاة في مجموعة البيانات. يتم إجراء تحليل تجريبي متعمق لتقييم أداء النموذجين.

يحقق النموذج الذي يشتمل على K-Means لحساب الارتساء متوسط دقة متوسط mAP يبلغ 87.6٪ ودرجة F1 تبلغ 83.7٪. في المقابل، فإن النموذج الذي يستخدم Fuzzy C-Means يحقق خريطة أعلى قليلاً بنسبة 88.1٪ ودرجة F1 تبلغ 84.2٪. في قسم نتائج الاستدلال، تُستخدم الصور الواقعية بدون تعليقات توضيحية سابقة لتقييم الأداء العملي لكلا النموذجين.

من خلال الفحص البصري ومقارنة المربعات المحيطة، تتأكد الدراسة من فعالية كل نموذج في اكتشاف المشاة في ظروف الحياة الحقيقية. تخلص الأطروحة إلى أن النموذج الذي يتضمن مجموعة Fuzzy C-Means يُظهر أداءً متفوقًا بشكل هامشي مقارنةً بالنموذج الذي يستخدم مجموعات K-Means . يساهم هذا البحث في مجال اكتشاف المشاة من خلال تسليط الضوء على تأثير تحسين صندوق التثبيت من خلال خوارزميات التجميع ويوفر رؤى حول النشر العملي لنماذج التعلم العميق لاكتشاف المشاة في البيئات المزدحمة.