PALLET DETECTION, TRACKING AND COUNTING USING COMPUTER VISION TECHNIQUES TO ACHIEVE THE WAREHOUSE AUTOMATION

SESHA VENKATA SRIRAM ERRAMILLI

Final Thesis Report

JUNE 2022

ACKNOWLEDGEMENTS

I dedicate this thesis work, which is part of my MSc with Liverpool John Moores University, to my father Mr Lakshmi Narayana Erramilli, my mother Mrs Venkata Lakshmi Bhavani Erramilli, and my entire family, who always supports me in every endeavour.

I am thankful to my thesis supervisor Dr Asif Iqbal Khan, a Post-doctoral researcher at Jamia Millia Islamia, New Delhi. With his guidance, I gained knowledge of the real-time object detection and tracking methodologies that are being used. Interacting with him always motivates me to delve deeper into the subjects and learn something new.

I am grateful to our trainers Mr Kaushik Ghatak, (founder, and director of the ValueQwest Pte. Ltd.) and Mr Ramachandran Kallankara (Analytics and Business Consultant, Freelancer) for providing the valuable training on the Supply chain management concepts with the real-time case studies, during my tenure as an SCM Engineer at Prakat solutions private limited, Bangalore.

And I thank all others who helped me directly or indirectly in the successful completion of this thesis work.

ABSTRACT

The popularity of data science is showing an increasing trend, as all kinds of the industry across the globe are adopting it for various use cases. Industries like healthcare, are utilizing state-of-the-art deep learning and computer vision techniques like Convolution neural networks to detect cancer, brain tumours and most recently detecting the presence of COVID 19. The success of Data science methodologies in the health care industries, like predicting the diseases in advance and saving the lives, has become possible because of the vast availability of the data both in the structured and the unstructured form, and properly utilizing them to make analyses. Similarly, in the Supply chain management domain, large amounts of data are available. Supply chain managers strive hard in maintaining stable and efficient supply chains so that their respective organizations can achieve and maintain customer satisfaction and can earn more profits. The key component of the entire supply chain management is warehouse management. Because the role of warehouses comes in the two areas, one is storing the raw materials that are purchased from the suppliers and the other area is storing the manufactured goods or the finished goods that are to be delivered to the customers. In both the areas of warehousing, labour plays a major role in picking the items, staging, putting them into storage, packing, labelling, scanning, and shipping. During the scanning process, the chances of getting errors like improper scanning of the barcodes or RFID tags are high. This study has proposed and implemented a computer vision methodology by developing a custom object tracker which detects the pallet boxes with the help of an object detection model, then tracks and counts those boxes in a frame. We trained two separate object detection models using YOLOv5x and YOLOv4 algorithms, and they achieved the mean average precisions as 99.5% and 96% respectively. Then we deployed the YOLOv5x model on to the custom object tracker to perform object tracking and counting operations, and this has achieved multiple objects tracking accuracy or MOTA score as 92%. We developed another object tracker using DeepSORT algorithm, which achieved a MOTA score as 99% and outperformed the custom object tracker This methodology enables to track the items accurately, which are entering and leaving the warehouse, thereby contributing to achieve the efficient warehouse management system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

SCOR	Supply Chain Operations Reference
WMS	Warehouse Management System
IOT	Internet of things
RFID	Radio frequency identification
AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
ReLu	Rectified Linear unit
YOLO	You Only Look Once
SSD	Single Shot multi-box Detector
SORT	Simple Online Realtime Tracking
IOU	Intersection over union
mAP	Mean average precision
MOTA	Multiple objects tracking accuracy
ONNX	Open Neural Network Exchange
DNN	Deep Neural Networks
NMS	Non-Maximum Suppression

CHAPTER 1

INTRODUCTION

1.1 Background of the Study:

Supply chain management plays a crucial role in every business. Aim of the supply chain management is to provide a product or service to a customer at the right place, right time, right quality and at the right price, and in this process make profits. Supply chain management deals with the material flow from the suppliers to the manufacturer, from the manufacturer to the wholesaler, from the wholesaler to the retailers, and finally from the retailers to the end customers. While the money flows in the opposite direction quite often, i.e., from end customers to the suppliers and the information flow is bidirectional. So, in a nutshell, we can say that the supply chain management integrates each part in the entire product life cycle (Effective Supply Chain Management, n.d.).

Due to the rapid increasing of the volatile demand and the products which have high customizability, German government has launched a strategic initiative called the industry 4.0 in 2011 (Christoph Jan Bartodziej, 2017). This initiative aims at integrating the cyber and physical systems to achieve more efficient supply chain management. Now a days, due to this Industry 4.0 revolution, Digital supply chain management systems are becoming popular which enables to integrate virtually all the components of the traditional supply chain. Some of the examples are, Enterprise resource management or ERP tool, Customer relationship management or CRM and Supplier relationship management or SRM.

Most companies do implement their respective supply chain management strategies to achieve and maintain the satisfaction of their customers. But due to the complexity nature of the business processes in every organization, a large amount of data is being generated at each department of the supply chain – say in supply, demand, sales, operations etc., (Górtowski, 2019).Unable to process this huge amount of data, the traditional systems are becoming inefficient (Tirkolaee et al., 2021).

Due to the increasing popularity of Data science across all the domains, researchers started exploring its application in the Supply chain management as well and as a result, companies also started utilizing Data science methodologies. The well-known use case of Data science application in the supply chain management is demand and sales forecasting. By using the nonlinear machine learning techniques, the sales team can accurately predict and forecast the inventory, sales and demand which leads to the increase in profits (Tirkolaee et al., 2021). The research proposed the application of multi-layer perceptron i.e., Artificial neural networks for solving the demand and sales forecasting problem of a retail store (Taghizadeh student, n.d.). Likewise, there are many use cases in the sales and demand forecasting.

Apart from the Sales and demand forecasting, there are other use cases of the supply chain like, supplier identification and evaluation, warehouse management, risk management, production planning, generating the optimum delivery routes, sentiment analysis of the customers, etc. Among them, warehouse management is very crucial use case, which involves a tremendous effort of manpower and hence there's a high possibility of occurrence of human errors in tracking the items that enter or leave the warehouse (Dewa et al., 2017).

There has been a lot of research going on in the warehouse management, one among them, has proposed an Unmanned aerial vehicle-based robot that moves in the warehouse and by utilizing the encoder decoder based convolutional neural networks architecture. It detects and scans the bar code, thus reducing the time of stocking and the number of mistakes in scanning the barcode decreases (Kalinov et al., 2020).

One more research has proposed a low-cost automated inventory tracking model for an automotive manufacturing company (Gray et al., 2021). The proposed model utilizes the SSDMobileNetV2 and the YOLOv4 tiny as separate object detection models, to identify the storage rack containing the vehicle parts and reads the barcode information using the Optical character recognition techniques. By augmenting this model to a high-definition camera, that is attached to a mechanically autonomous imaging module which is suspended aerially travels in the aisles of the inventory warehouse, reads the data about the inventory, and stores it in a data base thus enabling the access to the real time data.

The utilization of computer vision techniques is also increasing in many domains and warehouse management is not an exception as seen from the above two use cases. For example, YOLO series has many numbers of object detection models, that were trained on the standard data sets like MS COCO and Image net. These models can be customized as per our use case and can yield accurate results. Starting from the facial recognition attendance marking system to the latest driverless cars, computer vision methods are being applied.

In this research, we apply the computer vision techniques - object detection and tracking, to solve the item tracking problems in warehouses, by using our pallet images dataset.

1.2 Problem Statement:

1.2.1 Challenges in the Warehouse management:

Warehousing plays an important part in the entire Supply chain management of an organization, both in terms of

a) Storing the raw material components which are received from the suppliers.
b) Storing the finished goods which will be delivered to the customers in the form of trade units like pallets.
c) Contract warehousing, which is handled by a third-party vendor.

But there are some challenges in the warehouse operations. Scenarios like not getting the right data at the right time, occurrence of human error, as the manpower is a crucial part in the warehouse operations are one of the challenges (Dewa et al., 2017). If these challenges are not dealt properly, there might be a risk of losing the customer satisfaction, which is the top priority for any business.

Due to the Industry 4.0 revolution, there is a huge availability of solutions to tackle these challenges, to achieve the efficient automated warehouse management (Christoph Jan Bartodziej, 2017). The technologies such as sensors, computer vision, augmented reality (AR), drones, Internet of things (IoT) and robotics provide many opportunities to automate and optimize the warehousing, working along with the manpower. This also enables us to predict the appearance of bottlenecks in advance, thus ensuring the effective warehouse management (Lototsky et al., 2019).

Some of the global organizations have already started deploying the technologies like Internet of things (IOT) and Big Data analytics in their Warehouse management systems to achieve efficient operations.An IOT based Warehouse management system, which completely utilizes the Radio Frequency Identification or RFID technology, and the wireless sensors to track and trace the raw materials and finished goods (Lee et al., 2018). This enables the real time information sharing and ensures the availability of the right data at the right time, which in turn enables to be in sync with the changing customer requirements. However, there are some challenges with the above mentioned IOT based methodology. As the Barcodes and the RFID tags are the typical ways to identify the items in a warehouse, there's a possibility of occurrence of damage to barcodes and the RFID tags have a limited lifetime, which involve huge costs in assigning new barcodes/RFID tags and thus affects the inventory tracking.

1.2.2 Computer vision and Deep learning-based solution:

One of the solutions to this problem, is to use Computer vision and Deep learning-based approach. One research has proposed that the items entering a warehouse can be classified using convolutional neural networks with greater accuracy and helps in sorting the items according to their respective classes (Patel and Chowdhury, 2020).

In addition to the above CNN based approach, Object Detection and tracking also becoming one of the popular approaches to carry out the operations like object identification/classification, detection, and tracking. The necessity of combining the classical neural network models and the computer vision methods yields an accurate result (Yang et al., 2021). The proposed methodology in one of the research papers, shows the Faster R-CNN cascaded with the CNN based classifier can be used for object detections while the Kalman filters are used for tracking the objects (Mohamed et al., 2020).

In this research, we try to solve the problem of detecting, counting, and tracking the pallet boxes using the object detection and tracking algorithms. We use YOLO versions for object detection, compare their performances and choose the best one among them. Then we develop and use our customized object tracking algorithm for counting and tracking the objects, with the help of OpenCV library in python. The solution that we propose will be implemented using the python programming language and aims to achieve an efficient warehousing system that reduces the human effort**.**

1.2.3 Why YOLO for Object detection?

There are many popular object detection algorithms that are widely used. Algorithms like R-CNN generates regional proposals for each image and those are passed to convolutional network for extracting feature maps (Girshick et al., n.d.). The extracted feature maps are passed to the fully connected layers, from which the output will be branched into two parts – one outputs the class of the object detected and the other one outputs the values of the bounding box by applying regression. The generation of region proposals in the R-CNN and training them is a more time-consuming task.

YOLO is a single stage object detector algorithm (Redmon et al., 2015). Unlike the algorithms like R-CNN where the object detection is done by the classifiers and the bounding boxes by the regressors, YOLO framed the entire object detection problem into a single regression problem, i.e., bounding boxes and their respective class probabilities are predicted in the form of continuous variables. To do so, it utilizes a single neural network architecture, which makes predictions directly from full images in a single evaluation. So, the name YOLO is You only look once.

There are many versions of YOLO since its initial release. Here's the brief description for each one of them:

- YOLOv2 is developed by making a lot of improvements on top of the initial YOLO and anchor boxes are one among them. It can detect over 9000 object categories (Redmon and Farhadi, 2016).
- YOLOv3 is a faster one compared to its previous version. Built upon previous versions with the additional features like predicting the objectiveness score for each bounding box, it improved the performance even on the smaller objects (Redmon and Farhadi, 2018).
- YOLOv4 is still being used by many to perform object detections as it makes real time detection as priority and conducts training on a single GPU. It introduced other improvements like improved feature aggregation, mish activation and more (Bochkovskiy et al., 2020).
- YOLOv5 by Glenn Jocher released shortly after the release of YOLOv4 in the same year 2020. It was developed in Pytorch framework and many versions of YOLOv5 are being developed, examples are YOLOv5n, YOLOv5s, YOLOvm, YOLOv5l and YOLOv5x, which work on the 640x640 size of pixels. YOLOv5x is the fastest among all the above-mentioned versions (Glenn Jocher, 2020).

Despite YOLO makes localization errors, it is faster and makes accurate predictions. There are other single shot detectors like SSD (Liu et al., 2015). A comparative study can be performed by experimenting with all the state-of-the-art object detection algorithms, but due to the time limitation and considering the available computational power, in this research only YOLO will be used for object detection. we are going to make two models, one with YOLOv4 using darknet architecture, and the other one with the YOLOv5x one of the versions of YOLOv5 in Pytorch, then choosing the best one among them by using the evaluation metrics.

1.3 Aim and Objectives:

Artificial intelligence and data science are being applied in many domains and Supply chain management is not an exception. Due to Industry 4.0 revolution, there are a lot of use cases in this domain, which can be automated to achieve the efficient and low error operations. One of them is the warehouse management, in which the traditional approaches like manual counting of the incoming and outgoing items. Despite having unique RFID tags or barcodes on each item, manpower is utilized to scan the items by holding the scanning machines. And there are certain challenges like, damage of barcode labels, RFID tags and human errors can lead to inefficient warehousing. To solve this problem, we will be using the computer vision techniques like object detection and tracking, to count the pallets in the warehouse and the dataset we will be using is provided by iNeuron.ai, which is an open-source contribution. Figure 1.1 shows the sample image belonging to the dataset, containing pallet boxes. The pallet boxes in these images will be labelled by drawing bounding boxes around them and labelled as “pallet_box” before they are used for training the object detection models.

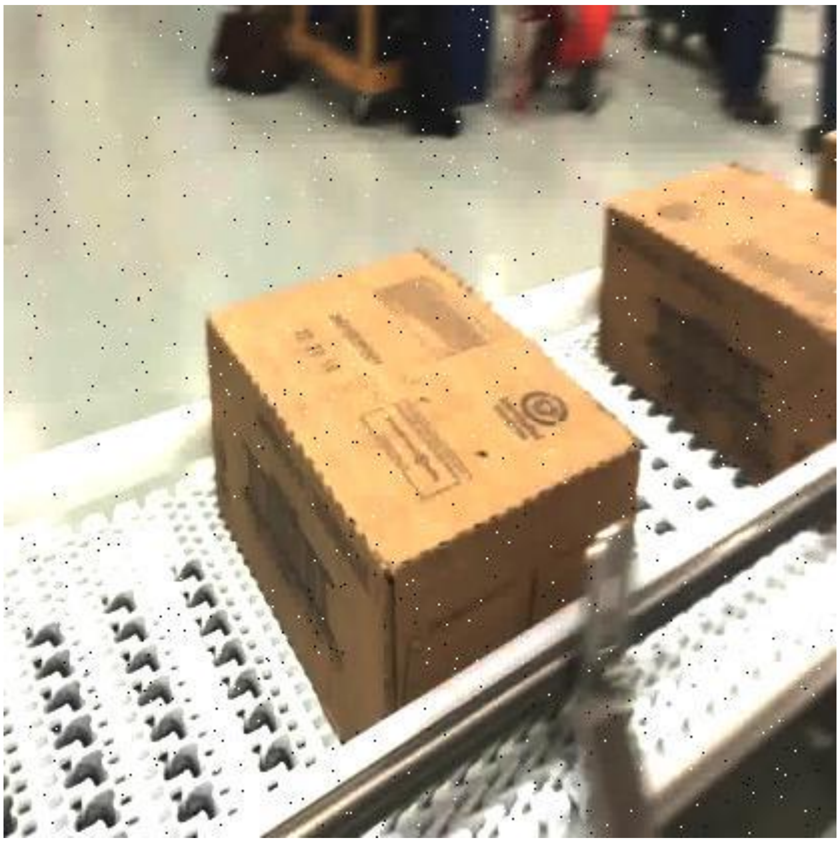


Figure 1.1 Sample image from the dataset containing pallet boxes.

Following objectives are formulated based on our study:

a. Data preparation for analysis.

b. Experiment with Yolov4 and Yolov5 object detection algorithms on the dataset, compare their results based on IOU, precision, recall and mean average precision metrics, and based on that choose the best one.

c. Then we will be developing a custom object tracker, which uses the best model among the two, to perform object detection, and then performs multiple objects tracking.

The reason for proposing a customized tracking algorithm is to experiment with the identity switches problem, occurrence of false positives and false negatives, and try to minimize them.

1.4 Scope of the Study:

Due to the time limitations and with the available computational power, the scope of our study will be limited to:

- Using only Yolov4 and Yolov5 models for object detection tasks and choosing the best among them by comparing their respective performance metrics.
- Our focus will be on developing a customized object tracker.
- Using only the dataset provided by iNeuron.ai for training and testing our models.

1.5 Significance of the Study:

This research enables us to develop a tracking algorithm, which can couple with the detector to detect, track, and count the items in the warehouse and stores the data into the database. We will be making open-source contribution of our code so that, anyone can use it, can make their own versions of it.

Cameras can be augmented with this real time object detection and tracking system, so that whatever goods enter or leave the warehouse, they can capture it effectively. This reduces the human errors and increases the efficiency, so that the supply chain management can have the right data at the right time to take the right decisions and ultimately leads to profits.

1.6 Structure of the Study:

The structure of study is split into the chapters as follows:

Chapter 1: - This chapter introduces the reader about the background of the research and the problem statement. It then explains the aim of this research and the objectives to accomplish.

Chapter 2: - This chapter then discusses the existing literature on the Supply chain management, SCOR model, warehouse operations like receiving, storage, order picking, sorting, and transporting, the application of Industry 4.0 – i.e., IOT based approaches, Big Data analytics, machine learning, and computer vision technologies can make warehousing a cost efficient and more profitable.

Chapter 3: - This chapter explains the computer vision methodology in solving the problem and the path to accomplish the objectives of this research.

Chapter 4: - This chapter discusses the implementation of the proposed computer vision methodology in detail.

Chapter 5: - This chapter discusses the results of the implemented computer vision methodology.

Chapter 6: - This chapter concludes the entire study and discusses the future recommendations.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction:

This entire literature review follows the funnel down approach, i.e., this review starts from the evolution of the supply chain management followed by its significance, the Supply chain operations reference model or the SCOR model and then delves into the Warehousing review, which explains the existing technologies that are being applied to achieve the cost-efficient and profitable warehousing nowadays, and this is the core component and foundation to carry out this research. All the popular and widely used computer vision techniques have been reviewed and their application in warehouse management is discussed.

This work is an inspiration from the figure 2.1.

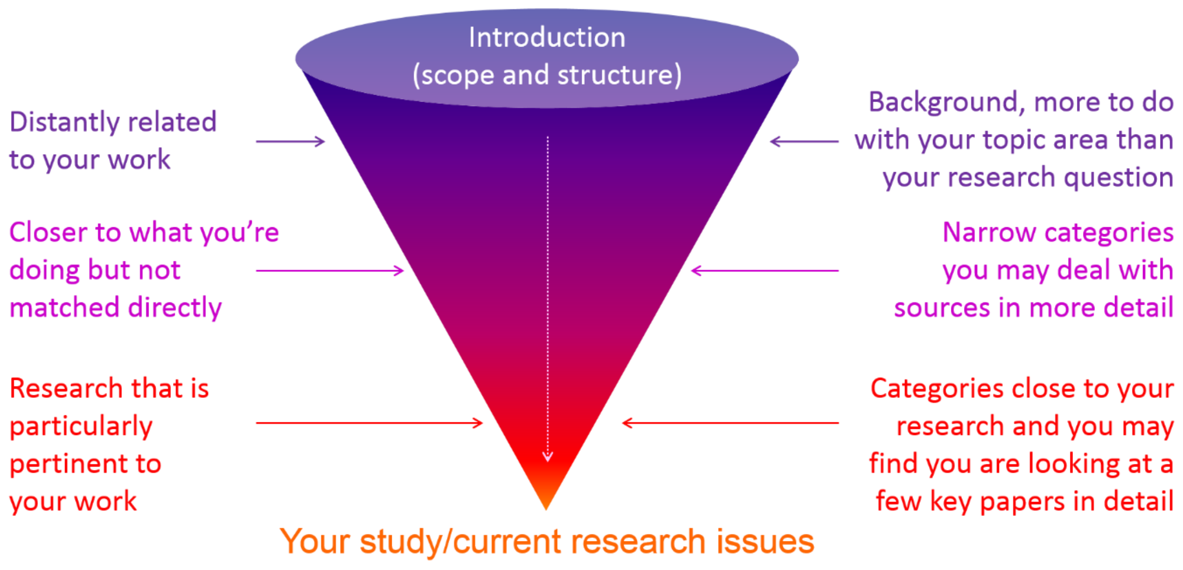

Introduction
(scope and structure)
Distantly related to your work
Background, more to do with your topic area than your research question
Closer to what you're doing but not matched directly
Narrow categories you may deal with sources in more detail
Research that is particularly pertinent to your work
Categories close to your research and you may find you are looking at a few key papers in detail
Your study/current research issues

Figure 2.1 Funnel technique that narrows down our views about the area of research.

Source: - https://canvas.hull.ac.uk/courses/779/pages/writing-the-review

2.2 Supply chain management:

Supply chain management deals with the coordination and the integration of every process of a business, thereby achieving an efficient product/service flow, information flow and the money flow among the suppliers, manufacturers, distributors, and the end customers (Mabert and Venkataramanan, n.d.). The goal of efficient supply chain management is to make sure the availability of the right products/services at the right place, at the right time, to achieve the right level of profitability and customer satisfaction. Over the past century, there have been three major revolutions in supply chain management, which transformed the supply chains from vertically integrated to the virtually integrated (Janat Shah, 2016). In terms of products, it had transformed from generating mass production to mass customization.

2.2.1 Evolution of Supply chain management:

The first major revolution took place in 1900s, talks about the vertical integration i.e., tightly integrating each process of the business. At that time, Ford Motor company owned every part of their supply chain say from the production of the raw material components to the final product "Model T", and it could manage the entire process from the iron ore to the finished automobile in 81 hours. Like Ford Motor company, many other automobile firms were also the highly integrated firms. Indeed, this vertical integration strategy was highly efficient at that time, as the firms used to offer low variety of products.

But due to the changes with respect to the time, like a trend towards a wide product variety, this vertical integration strategy became inflexible. To deal with this challenge, companies had to re-engineer their supply chains to make it efficient as well as flexible by offering a variety of a product, like different flavours of an Ice cream. This marked the second major revolution in 1960s. Toyota motor company came up with an idea of carrying out only the manufacturing of key components and the final assembling operations, in house while the other raw materials were sourced from many suppliers, who were located very close to their assembling plants. Toyota maintained long term relationships with their suppliers. This went fine until Toyota tried to expand their business by setting up assembly plants in different parts of the world, where it became tough to take their suppliers along with them. At that time some of their suppliers became no longer cost competitive, so having permanent relations with them became a liability over a period. With the advancements in the Information technology, paved way for the third revolution.

Third revolution started during the 1990s, enabled the customers to get more customized products and services. This became possible as most of the firms started adopting virtually integrated global supply chain strategies. A well-known example is Apple Inc. Apple doesn't have its own manufacturing plant instead it has outsourced all its operations across the globe. Apple focuses more on providing a better user experience by providing the unique personalized services to each customer. Also, Apple encourages its consumers to contribute to a core areas like app development, which enables it consumers to become partners and provide more wide customized products and services by developing applications as per the unique user experiences. The key driver for Apple’s progress is the utilization of the information technology and the internet.

There are many other firms in today’s world, where the mass customization is the key component. From this we can say that organizations now a days are more focussed on providing the better user experience to the customers unlike in the early years of the last century, where the mass production of a product was the driver.

2.2.2 SCOR model:

To provide customized products and services to the customers, firms put a lot of efforts starting from sourcing the raw materials, manufacturing, and distribution. Ideally this looked simple, but practically firms suffered with some issues like product recalls. The main reason for happening this issue is failing to meet the quality of the products, despite quality assurance is the top priority for any firm, that helps in achieving the customer satisfaction (Li et al., 2011). To solve this problem, Supply chain operations reference or SCOR model is used as a diagnostic tool.

SCOR model was developed by the Supply chain council. It is a strategic planning tool that can be used by the senior supply chain managers to simplify the complexity of the supply chains and thus can make strategic decisions (Huan et al., 2004). The SCOR model is described as shown in the figure 2.2

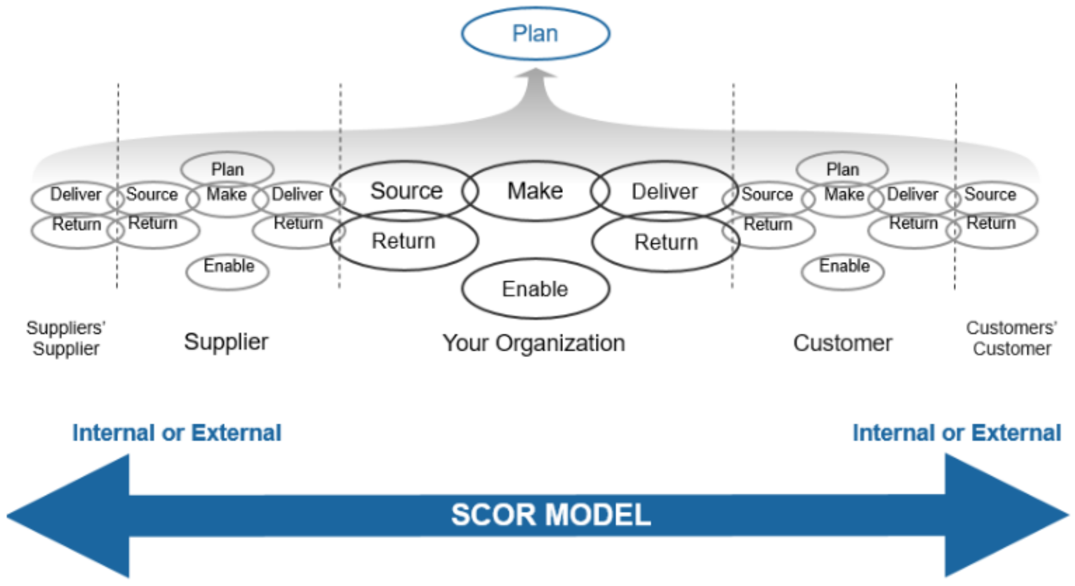


Figure 2.2 Supply chain operations reference model (APICS Supply Chain Operations Reference Model SCOR Version 12.0, 2017)

Many organizations may take part in a single supply chain and each one of them implements the following processes.

1. Plan: Planning is the first step, as it involves determining the requirements, resources, deciding on the inventory storage capacity, transportation and logistics, assets, etc., and the goal is to make sure all these things are in line with the business goals, like ensuring the supplies are with respect to the customer demand, by implementing an efficient supply chain strategy.
2. Source: This process deals with purchasing fixed assets and the supplier sourcing. Once the supplier is sourced, the organization makes a contract with them to procure the raw materials or products from them. This involves purchase requisitions, purchase orders, supplier invoices, and then the supplier delivers their goods to the organization, in return they get their payment. The purchased raw materials or products from the suppliers are stored in warehouses. This Source component also involves supplier performance monitoring, which enables the procurement managers to decide whether to continue relations with the suppliers or not.
3. Make: This process deals with the manufacturing of the final product. Depending on the type of the products or services that the organization provides, the manufacturing can be a discrete manufacturing or a process manufacturing. In this phase, an organization utilizes the raw materials or the ingredients that were purchased from the suppliers, operates their manufacturing equipment which can be their assets and finally produces finished goods, which are then subjected to the quality check and then they will be sent to the warehouses for making deliveries.
4. Deliver: This phase deals with the warehouse management, order management and transport management. Once an order is received from a customer, the ordered product will be picked up from the warehouse. Then, a lead time (the time by which the order reaches its customer) will be decided. To make on time deliveries, transport and logistics team decides an optimal mode of transport that provides a fast delivery in a cost-efficient method, thus fulfilling the customer demand.
5. Return: This phase is also called as reverse supply chain. When customers are not satisfied with the product they received, they return it. So, the organization must accept the returned goods, and inspect the faults within it. This happens mostly in the case of electronic gadgets and clothing. This phase also involves returning raw materials to the suppliers if they are not up to the product’s standards. So, in a nutshell, it is a flow of

products from the customer to the organization or organization to its suppliers while the money flows in the reverse direction.

6. Enable: This phase deals with the risk management of the supply chain, compliances and other business rules that enables the organization to make profits while cutting down the redundant costs.

All the above 6 phases of the supply chain vary across the various organizations, and this ensures integrity among the all the organizations in the supply chain, ultimately achieving customer satisfaction and making profits.

In this thesis, our area of research is on the Warehouse management belonging to the "Deliver" part of the SCOR model.

2.3 Warehousing - Review:

2.3.1 Evolution:

Warehouse management system, part of the supply chain management, deals with the material management. The transformation journey of the warehouse started from being just a storeroom for storing the materials to the smart warehousing system, which works faster enough to meet the supply and demand cycles of the entire supply chain (Kumar et al., 2021a).

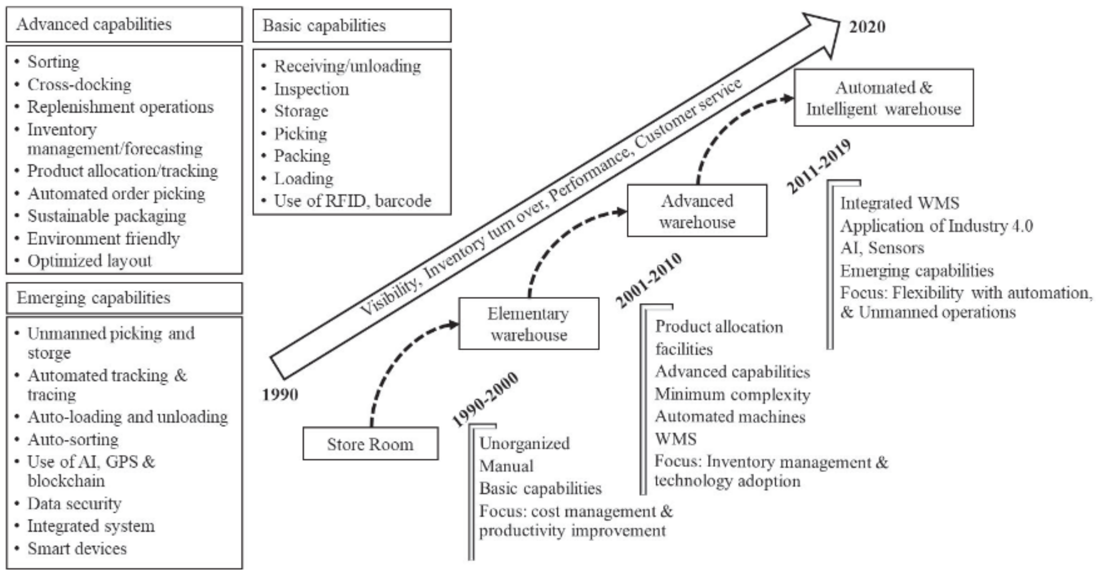

Advanced capabilities
• Sorting
• Cross-docking
• Replenishment operations
• Inventory management/forecasting
• Product allocation/tracking
• Automated order picking
• Sustainable packaging
• Environment friendly
• Optimized layout
Basic capabilities
• Receiving/unloading
• Inspection
• Storage
• Picking
• Packing
• Loading
• Use of RFID, barcode
Emerging capabilities
• Unmanned picking and storge
• Automated tracking & tracing
• Auto-loading and unloading
• Auto-sorting
• Use of AI, GPS & blockchain
• Data security
• Integrated system
• Smart devices
Visibility, Inventory turn over, Performance, Customer service
1990
2020
Store Room
Elementary warehouse
Advanced warehouse
Automated & Intelligent warehouse
1990-2000
Unorganized
Manual
Basic capabilities
Focus: cost management & productivity improvement
2001-2010
Product allocation facilities
Advanced capabilities
Minimum complexity
Automated machines
WMS
Focus: Inventory management & technology adoption
2011-2019
Integrated WMS
Application of Industry 4.0
AI, Sensors
Emerging capabilities
Focus: Flexibility with automation, & Unmanned operations

Figure 2.3 Evolution of warehousing (Kumar et al., 2021a)

One of the studies has discussed about the designing of warehouses to meet the storage requirements and the optimization of the throughput capacities (Cormier and Gunn, 1992). Because carrying out warehousing operations do involve certain costs like transportation/shipping costs, labour costs, Electrical energy, etc., so, the goal was to achieve an efficient warehousing with minimal cost. This study has discussed the various warehouse models and their respective operations, and classified the types of warehouses as follows:

1. Manual warehouses (picker to product systems)
2. Automated warehouses (product to picker systems)
3. Automatic warehouse.

Warehousing activities involve receiving, storage, picking and shipping. Among these four activities, picking is the highest contributor for the operating cost in at typical warehouse. So, depending on the use case there are many picking strategies like single order picking, batch picking, zone wise picking, pick and sort, etc., with respect to the manual warehouses. This involves a lot of human effort and there's a possibility of occurrence of errors. One of the studies has proposed a methodology to analyse the human errors in the warehouse activities, which helps in increasing the productivity of the warehouses (Dewa et al., 2017). Humans are one of the key assets of the warehouses and while dealing with the fast-moving consumer goods, despite of the variety increases with respect to the time, manpower must be accurately utilized by minimizing the errors.

2.3.2 Role of Industry 4.0 in warehouse automation:

The role played by the technology in the warehousing is very crucial. It takes care of some of the tedious and important tasks, while the manpower can be utilized in other innovative activities. In this e-commerce era, where there is a volatile demand, automating the warehouses helps in maintaining the supplies appropriately (Boysen et al., 2019). For example, deploying robots in order picking process, increases the speed and less error prone.

Considering this volatile demand, changing customer behaviour and the huge competition among the firms, German government launched the initiative called, "Industry 4.0" in 2011 (Christoph Jan Bartodziej, 2017). This gave birth to the new concept called, cyber physical machines, where the machines communicate among themselves giving rise to the smart factory. One of the studies explains that the industry 4.0 enables the firms to introduce new products and services into the market, mass customization and increases flexibility (Tjahjono et al., 2017).

An application of Internet of things in the warehouse management system results in data sharing among all the physical devices (Lee et al., 2018). This approach uses RFID tags, which will be pasted on the goods (unique RFID tag to each item) and the RFID scanners scan the items, transmit their information to the server, from there it can be transmitted to the warehouse managers, to trace the items.

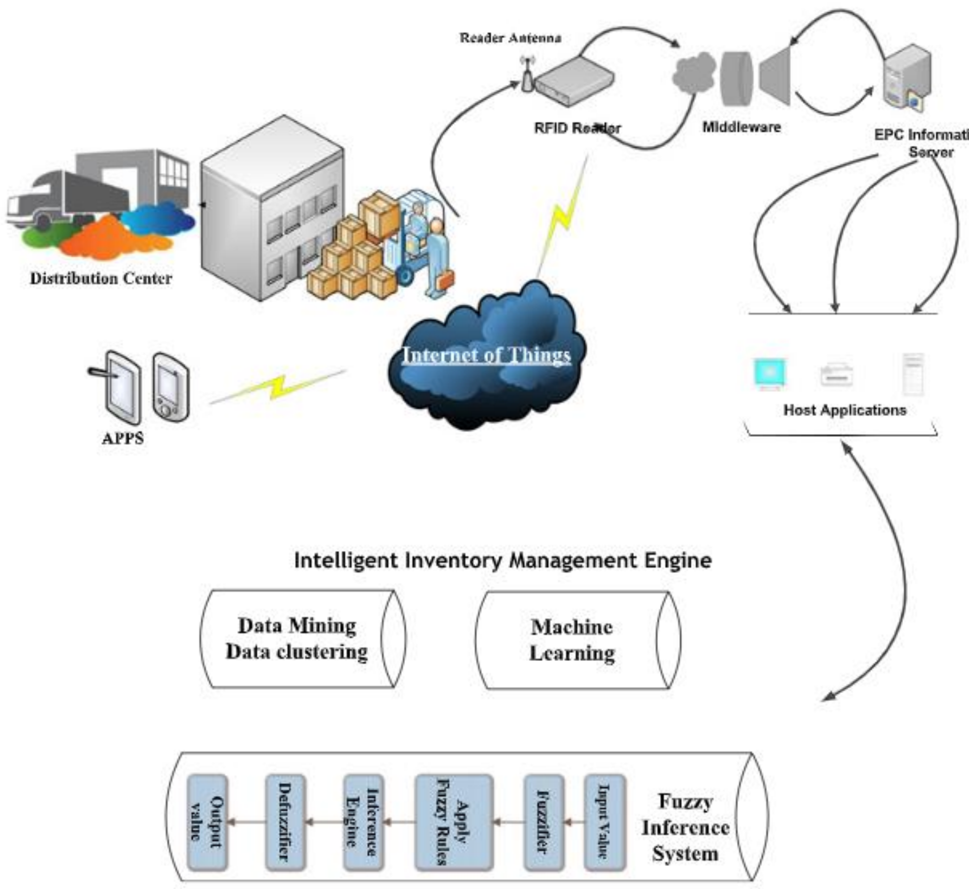


Figure 2.4 An IOT based warehouse management framework (Lee et al., 2018)

Applications of the wireless technologies like Zigbee, GPRS – 3G/4G, not only benefited in achieving the smart warehousing but also in achieving the smart freight transportation and the smart distribution (Ding et al., 2021). For instance, we can track the trucks in transit and can estimate the time of delivery, to make an advanced decisions One of the studies has proposed the joint application of RFID and the ultra-wide band mobile terminal to gather information about the goods, their location with the help of GPS and sense the goods to take the decision of loading or unloading them (Zhao et al., 2020). The RFID reader and antennas are installed to the forklift, which can be drove in the aisles of the warehouse and scans the RFID tags on the goods.

Despite having many benefits of using the IOT based approaches – like real time data sharing, there are some challenges.

1. Internet connectivity issues
2. Possibility of the damage to the RFID tags like barcodes and QR codes.

If any one of the above happens, we may get inaccurate results and may lose track on any item.

2.3.3 Data analytics in WMS:

The implementation of IOT based technologies also enabled to generate huge amount of data. The data can be either in the form of structured or unstructured (like images, videos, or anything). Organizations like Amazon, Microsoft which are having data centres across the globe, are providing the cloud-based services with an affordable price and many companies also shifting towards the cloud for storing their massive data instead of relying on the traditional server-based approaches. The application of big data analytics on the data generated derives the hidden insights, which can be used to optimize the businesses (Ghaouta et al., 2018).

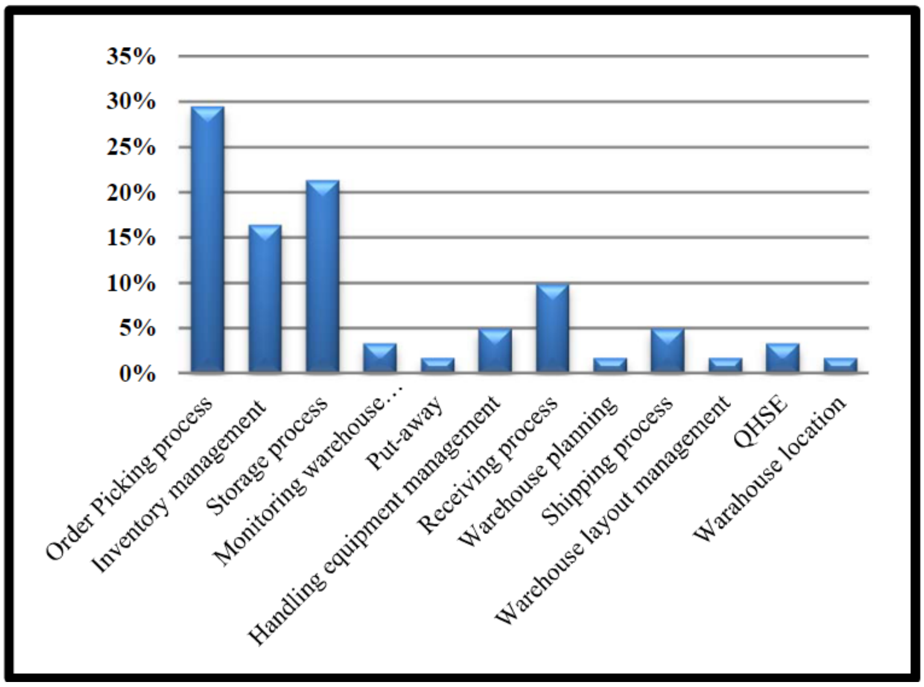


Figure 2.5 Application of Big data analytics in warehousing activities (Ghaouta et al., 2018)

As shown in the figure 2.5, there's a huge application of big data analytics in the order picking process. Likewise, the applications of predictive analytics can provide an interesting pattern. Machine learning technology can be applied in WMS to identify the hidden patterns in the inventory management data and the features that are responsible for the costs (Tirkolaee et al., 2021). Based on the results obtained, the warehouse managers can take an appropriate decision to optimize the operations. Another application is quality check and the damage inspection. The products or the raw materials can be classified accurately whether they are good for utility. Machine learning is widely used in generating the better delivery routes, so that the delivery trucks can follow them to provide the timely deliveries. AI and machine learning also works well to achieve the resilient supply chains (Toorajipour et al., 2021). This helps in maintaining the stability of the business even in the times of pandemic. It also helps in dealing with the bull whip effect by accurately forecasting the demand thereby helps in reducing the production costs, transportation and logistics costs and the wastage (Hoque et al., 2021).

With the advancement in Deep learning, the application of artificial neural networks or ANNs in the sales forecasting gives even more accurate results than the regression and the time series algorithms. A study has proposed the use of neural networks in the sales forecasting and in addition to that it explains how to interpret the neural network model using the SHAP techniques (Chen et al., 2021). So, the availability of large amounts of data enables to apply deep learning techniques as well to get the desired results.

2.4 Computer vision techniques:

2.4.1 Computer vision techniques – Review:

Computer vision is an area of study that deals with the extraction and analysis of information obtained from the images. In other words, it is a technique that enables the computer to simulate human visualization by pre-processing the images, extracting features from them, segmenting or detecting the objects and decision making. This entire framework is made up of two groups, one is 3D morphological analysis, and the other is pixel optimization (Wiley and Lucas, 2018). Image classification is one of the important applications of the computer vision. Before the advent of Deep learning methods, researchers used to put a lot of effort in designing scale invariant features like SIFT - Scale Invariant Feature Transform and HOG - Histogram of Oriented Gradients, feature representations like Bag of features and Fisher Kernel, and classifiers such as Support Vector Machines or SVM to carryout Image classification (Feng et al., 2019). However, these methodologies faced challenges while classifying the objects in natural images with complex background, illumination, and ever-changing poses. The development of Convolutional neural networks (CNNs) indeed brought a significant change in the field of computer vision. Example, at the ImageNet Large Scale Visual Recognition Challenge or ILSVRC that took place in 2012, AlexNET - a CNN architectutre won the first prize surpassing the conventional computer vision techniques by a significant margin.

CNNs are derived from the traditional ANNs or Artificial neural networks. Initially, to deal with the smaller datasets like MNIST, which are having sizes of just 28 × 28, ANNs were sufficient to carryout image classification tasks. But to deal with the more substantial-coloured images, ANNs requires a greater number of weights on just a single neuron, making it a very complex to train. And one more drawback of using ANN for image classification task is, it can take the one-dimensional data, which means, the feature maps need to be transformed into 1-D data, which may lead to the loss of some of the features. To solve this problem, CNNs were designed. One of the main differences between ANN and CNN is, the former one can deal with the one-dimensional data while the latter one can deal with the three-dimensional data i.e., having height, weight, and depth.

CNNs are made up of three types of layers – Convolutional layers, pooling layers, and fully connected layers, which are stacked to get a CNN architecture (O'Shea and Nash, 2015).

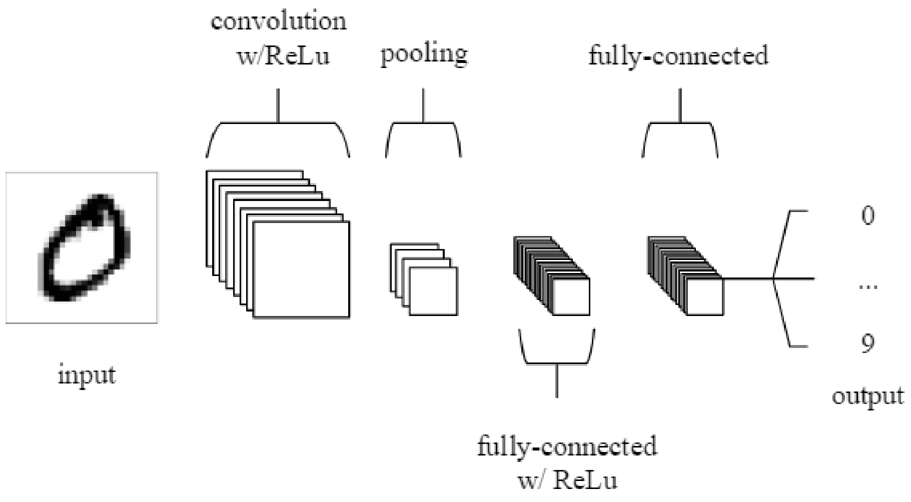


Figure 2.6 A Simple CNN architecture with 5 layers (O'Shea and Nash, 2015)

The CNN architecture shown in figure 2.6 works on the MNIST dataset to classify the handwritten digits. It consists of an input layer which holds the pixel values of an input image. The convolution layers extract features from the input image by calculating the scalar product between their weights and the corresponding region connected to the input image. The activation function ReLu, is applied on the output obtained from the layers, which decides whether a neuron should be activated or not. Then the pooling layer down samples the spatial dimensions of the feature maps obtained from the convolutional layers, which further reduces the number of parameters within that activation. Then the output from the pooling layer of the last convolutional layer will be taken as an input by the fully connected layers, which are nothing but the standard ANNs, which provides the final output in the form of class probabilities. The class having highest probability score, is the predicted class of the given image. This is the brief explanation of how the CNN works.

There are many CNN architectures, which are trained on the huge image datasets and are available for open source, so that anyone can use them for the custom use cases through transfer learning. The figure 2.7 shows the comparison of the popular CNN architectures based on their respective accuracies and the no. of operations required to classify an image. The size of each circle represents the number of parameters in its respective network.

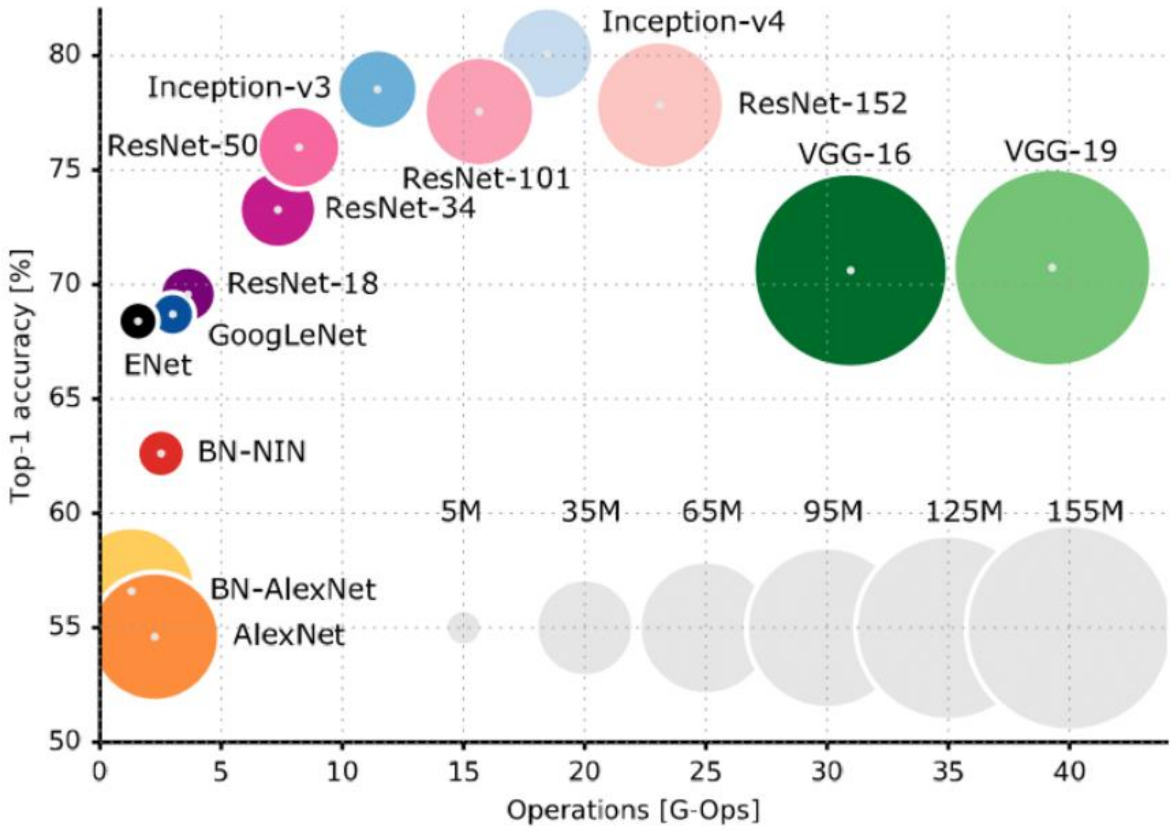


Figure 2.7 Comparison of the popular CNN architectures. Source: - https://www.researchgate.net/figure/Comparison-of-popular-CNN-architectures-The-vertical-axis-shows-top-1-accuracy-on_fig2_320084139

Apart from the Image classification, the other important applications of the computer vision are:

1. Object detection (detecting objects in an image or video)
2. Image segmentation (a pixel level classification technique which divides an image into meaningful regions by classifying each pixel into a specific entity (Feng et al., 2019)).

Our research problem is completely related to the objected detection problem. So, from the above two applications, we will be focussing only on the former one i.e., Object detection.

2.4.2 Object detection architectures:

Object detection is a methodology that determines and locates the object in an image or a video, and classifies it based on the predefined categories (Feng et al., 2019). Compared to the Image classification problem, object detection problem is a challenging one, as the algorithm must locate the object in an image accurately and must handle many highly variable objects. Following is the summary of the popular object detection architectures.

- R-CNN:

  It is a 2-stage detector, where it uses Selective search algorithm to generate region proposals during the first stage. AlexNet CNN architecture extracts features from those region proposals and each one of them is classified by a set of category specific linear SVMs in the second stage. This architecture achieved a mean average precision (mAP) of 53.7% on PASCAL VOC 2010 (Girshick et al., n.d.). However, it is inefficient due to its complex 2 stage pipeline.

- Fast R-CNN:

  This architecture, inspired from the SPPNets or Spatial Pyramid Pooling networks, performs a single stage training, i.e., performs only the object detection operation on the region proposals that are generated externally by a separate algorithms like Selective search or Edge box. This network consists of a Region of interest or ROI pooling layer before the fully connected layer, which produces the fixed length feature vector for each region proposal. During the training, all the network layers get updated in a single convolution operation and uses multitask loss. It utilizes deep VGG16 CNN architecture for training, and it is 9 times faster than R-CNN and achieved the mAP of 66% on PASCAL VOC 2012 (Girshick, n.d.). Although this is better than R-CNN, it requires expensive computation for generating the region proposals externally.

- Faster R-CNN:

  This architecture is obtained by merging the Fast R-CNN with a Region proposal network or RPN, which utilizes a CNN architecture to generate region proposals. Here both RPN and Fast R-CNN share most of the convolution layers and the features from the last shared layer are used for both region proposal generation and region classification. This architecture is highly efficient as compared to the R-CNN and Fast R-CNN as it utilizes GPU for both proposal generation and training and it achieved a mAP of 42.7% on MS COCO dataset, and a mAP of 75.9% on the COCO+VOC07++12 dataset (Ren et al., n.d.).

- Mask R-CNN:

  Based on the Faster R-CNN, Mask R-CNN was proposed, which performs the image segmentation task along with the object detection task, by using the ResNet101-FPN network. This architecture achieved a mAP of 39.8% on MS COCO dataset in 2017, greater than that of the results obtained by the Fast R-CNN and Faster R-CNN respectively (He et al., 2017).

- YOLO:

  YOLO or You Only Look Once, is the first single stage object detection architecture, that considers the object detection problem as a regression problem. This network is inspired by the GoogLeNet architecture, used in image classification. Compared to the two stage R-CNN architectures, YOLO is faster to train as it doesn't perform the proposal generations, instead it divides the entire image into several grids and proposes bounding boxes for each grid. Then using the features obtained by CNNs from the input image, it predicts the bounding box coordinates around an object and the class probabilities for that object. YOLO can work with the speed of 45 frames per second, faster than all the R-CNN architectures (Redmon et al., 2015). In terms of accuracy, YOLO lags the R-CNN architectures, as it struggles in detecting the small objects which are in groups, and it imposes strong spatial constraints on the bounding box predictions. But still, due to its advantages, it is widely used for solving the real-world object detection problems and there are many upgraded versions of YOLO available – YOLOv2, YOLOv3, YOLOv4 and YOLOv5, and new versions are being developed in addition to the aforementioned.

- SSD:

  SSD or Single shot multi-box detector, follows as similar singe stage detection strategy like YOLO, but it outperforms the YOLO architecture in terms of accuracy. After extracting the features from the feature maps generated by the CNNs, SSD produces a fixed size collection of bounding boxes and predicts the scores for the presence of objects in those boxes. Then non maximum suppression is performed to produce the final detections by removing the duplicate predictions i.e., bounding boxes. SSD513 architecture achieved a mAP of 76.8% on PASCAL VOC 2007 dataset surpassing the YOLO (VGG16) which achieved a mAP of 66.4% (Liu et al., 2015).

2.4.3 Multiple objects tracking algorithms:

Object detection algorithms or architectures detect objects in an image or in a video. Multiple Object tracking algorithms track multiple objects in a video by assigning unique Ids to each object. One of the prominent use cases of object tracking are visual surveillance, virtual reality, and human computer interaction. Multiple objects tracking algorithms can be divided into two groups – Detection based tracking and Detection free tracking, where the former one utilizes the trained object detector to detect objects and then those objects are linked into the trajectories, while the latter one requires manual initialization of a fixed number of objects in the first frames, then localizes these objects in the subsequent frames (Luo et al., 2021).

Multiple object tracking can be further classified into Online tracking and Offline tracking, where in the former approach, the image sequences or frames are handled in a sequential manner, i.e., the object or observation tracked in the current frame is dependent on its state in the previous frames. While the latter approach utilizes a batch of frames, and obtains observations from them, jointly analyses, and estimates the final output. SORT or Simple Online and Realtime Tracking is one of the popular online tracking algorithms which uses Kalman filters on the images to estimate the state of the objects, and the Hungarian algorithm to establish association among the similar objects in the subsequent frames by measuring the bounding box overlap (Bewley et al., 2016). In this algorithm, the IOU distance of the bounding boxes handles the short-term occlusion, when any other object covers the target object. This algorithm creates a unique tracking id for the objects and keeps them until they become undetected. In other words, if an object appears in a frame, an id will be assigned to it. If it disappears in the subsequent frame, its id will be terminated. In case it got occluded by another object in the previous frames i.e., in case of a long-term occlusions, and re appears in the current frame, a new id gets assigned to it, even though the object remains same. This problem is called as Identity switches.

DeepSORT is another popular object tracking algorithm and an extension to the SORT, solves the identity switches problem by using the more robust metric that combines both motion and appearance information by using the CNN network, that was trained to discriminate pedestrians on a large-scale person re identification dataset (Wojke et al., 2017). This algorithm achieved a Multiple object tracking accuracy or MOTA score of 61.4% in the MOT16 challenge, while the SORT achieved a MOTA score of 59.8%.

Despite having different approaches to do the multiple objects tracking task, still there are challenges to deal with the more severe occlusions. So, the research on the multiple objects tracking problem is still on going to design even more robust methodologies.

2.4.4 Applications of Computer vision in Warehouse management:

Computer vision has huge applications in the various industrial sectors. The ongoing development of the autonomous or self-driving cars is a well-known example. Likewise in the warehouse management also, it has huge applications. The methodology to accurately detect and identify the automobile parts using object detection is one of the use cases in the automobile warehouse (Gray et al., 2021). In this use case, the proposed computer vision model detects the vehicle parts and reads their corresponding barcode information using the OCR or optical character recognition technique, thus very much helpful for the just in time automobile manufacturers.

Another work proposed a heterogeneous unmanned aerial vehicle (UAV) that moves in the aisles of the warehouse and scans the barcodes of the pallet that are placed on a rack (Kalinov et al., 2020). This proposed UAV has a global shutter camera and an on-board computer to process the images. This utilizes the CNN- a U-net based architecture, that is trained on the pallet box image dataset, by applying the certain augmentation techniques on the images. The output of the CNN is processed with the standard OpenCV morphological transformations to erase noise and applied contours detection to get the coordinates of areas with barcodes. Then all the detected regions from CNN go to the factor-graph. The UAV updates and specifies its localization relative to each detected barcode region in every UAV camera image even before the laser scanner's verification. Although the proposed methodology reduces the human effort in scanning each pallet box, there are many challenges in operating the UAVs, like the occurrence of false positives in scanning the barcodes, the optimum height that drone has to fly to capture correctly the barcodes, the practical arrangement of pallets in the warehouse, etc.

One more use case explains the classification of items in the inventory using the convolutional neural network ResNet 50 (Patel and Chowdhury, 2020). But this works only on the image data and the response time of the model sometimes cause the inefficiencies in recording the data. Another study proposed an object detection framework along with the application of robotics in sorting the parcels quickly and accurately (Han et al., 2020). In addition to the detecting the objects, tracking them also extremely important as the security is a matter of concern. Deploying an object tracking application using DeepSORT algorithm along with the object detection, using the Yolov4 pretrained weights to detect and track people gathered at a place during the Covid pandemic time and violating the restrictions imposed by the government, can alert the concerned authorities to take appropriate actions (Kumar et al., 2021b). This study has inspired us to develop the methodology to detect the pallet boxes that are moving on a conveyor belt and track them with the unique tracking IDs by customizing the DeepSort algorithm, as this is a research gap in the warehouse management system identified by us.

2.5 Discussion:

Supply chain management is the crucial component of any business. One supply chain may consist of many organizations, playing different roles in satisfying the end customer’s demand. As the name suggests, it is a chain that connects various organizations to provide quality products and services. So, having an efficient supply chain is nothing but connecting the organizations with a stronger chain. There are many components in the supply chain, one of them is warehouse management. Warehouses are used to store both the supplies/raw materials that were sourced from the suppliers, and the finished products that are ready to be delivered to the retailers, so that end consumers can purchase from them. But an efficient warehousing requires a lot of human effort, to ensure smooth movement of goods inbound and outbound, and storing them in their respective racks. The possibility of occurrences of human errors may have the impact on the entire supply chain. Say, a man wants to purchase a smart phone, so visited an e-commerce website and it displayed that the product is out of stock. Then that man chooses an alternative e-commerce platform or goes out to a nearest retail store to purchase the desired product. But one quantity of that product still exists in the e-commerce company’s warehouse and the information related to it has not been updated in the system by the concerned employee. This resulted the e-commerce company to lose a sale. So, to reduce the human errors, researchers are working on achieving smart and automated warehouses. In this data driven world, application of Data science, AI and computer vision techniques have a huge scope in achieving the efficient warehouse management system, that in turn contributes to achieve an efficient supply chain.

2.6 Summary:

The literature review follows the funnel down approach, i.e., narrowing down from the Supply chain management concepts till the applications of the computer vision technologies in the warehouse management.

First, the evolution of the supply chain from the past century has been discussed with respect to the three major revolutions. This shows the transformation from the vertically integrated supply chain to the virtually integrated supply chain, where the former one worked well when there is only one product type and mass production was the only approach, while the latter one is the present situation, indicating the vast number of customized products fulfilling the demands of different groups of customers. SCOR or Supply chain operations reference model has been discussed to achieve the efficient supply chains, that helps in satisfying the customer demands and in turn brings the profits to the organizations.

Next, discussed about the warehousing concepts, and the existing literature on it, which starts with its evolution. Then discussed about the 4th revolution i.e., Industry 4.0 in the warehousing, indicating the applications of cyber physical systems – IOT based approaches to achieve the cost-efficient warehouse operations and the challenges of using RFID tags on the items. Then, discussed about the applications of the Big data technologies, machine learning in the WMS. And finally discussed about the computer vision applications – image classification, object detection and tracking, and how these techniques can be leveraged in achieving the smart and automated warehousing.

This review is the foundation to develop the methodology to detect and track the pallets in the warehouses, which provides an accurate data about the inbound and outbound pallets inside the warehouse. This also reduces a lot of human effort.

CHAPTER 3

RESEARCH METHODOLOGY

This chapter discusses about the proposed research methodology to solve the problem statement that has been defined.

3.1 Introduction:

This chapter starts with exploring the data set and making annotations on the training images. Then we customize the object detection algorithms Yolov4 and Yolov5 separately and train the two models. Once the models are trained, they are evaluated on the validation images and based on the performance metrics – Intersection over union (IOU), Precision, Recall and Mean Average Precision, we will compare the two models and choose the best one.

The weights of the best object models are then saved. Next comes the object tracking part, where a custom object tracker is developed. The performance of the tracking algorithm is measured by using the multiple objects tracking accuracy (MOTA) score. By using the best detector model weights and this newly developed object tracker algorithm, we can perform the object detection, object tracking and counting operations.

3.2 About the Dataset:

This dataset is provided by iNeuron.ai and it is made open source by them as a part of their open internship program, where anyone can sign up for free and start working on the projects. This dataset consists of different images of pallets, all belonging to single pallet class. Labelling part has already been done by someone, but for our research we will manually do the annotation for each image so that we can have all the annotations in a format suitable for analyses with zero errors. Along with the training images, we were also given videos, on which we can implement the tracker algorithm, which tracks and counts the number of pallet boxes in a frame.

3.3 Annotating the training images:

To train our detection models, we need to let them know what the objects in each image are. Just like for a regular tabular data, after dividing it column wise into the independent or feature variables and the dependent or target variable, for predicting the latter one based on the learning the relationship with the former one, here we need to annotate the objects in each image, so that the model can learn about those objects and can make predictions on the unseen images.

To do this, we draw a bounding box around each object that we want the detector to see, and we label each box with the object class, that we would like our detector to predict. "LabelImg" is one of the free labelling tools that can be used to accomplish the annotation task. All the annotations are to be made in YOLO format, i.e., in the form of (x, y, width, height) where x and y indicate the centre point coordinates of the bounding box, while the width and height are self-explanatory, indicate the width and height of the bounding box.

3.4 Training the custom object detection models:

Since we are having our pallet dataset and it is of single class type i.e., all the pallets come under the single pallet class, we will be customizing our YOLOv4 and YOLOv5 models accordingly. Here's the brief explanation about how our implementation is going to be. This procedure is just a proposal and may vary slightly at the time of implementation, depending on the software dependencies.

- **<u>Yolov4 model:</u>**

  We will implement it using Darknet framework in python.

  a) First, we will be downloading the Yolov4 pre trained weights from the Darknet Github repository (https://github.com/AlexeyAB/darknet) to our local system or google drive in case if we use google colab notebook. Then we will install the dependencies as per the requirements.

  b) Next, we need to create a folder or a directory with the name as 'yolov4' for easy identification and inside that create another directory 'training' where the model weights will be saved during the training.

  c) Download the 'yolov4-custom.cfg' configuration file and set the batch size, image size, learning rate and the number of convolution filters before each YOLO layer, with respect to the number of classes. Similarly, we create our obj.names and obj.labels files with the details like number of classes and labels respectively.

d) After customizing the configurations, we will start training our model on our labelled dataset. The trained weights get stored in a separate folder called “training”.

- **Yolov5 model:**

Since Yolov5 is written in pytorch framework, we will do model training using the pytorch library, by importing it in python. At the end, weights will be stored.

a) We will be Downloading or cloning the Yolov5 Github repository (https://github.com/ultralytics/yolov5) and install the dependencies.

b) Then we will install Torch GPU version capable to our Cuda version.

c) Next, we will create YAML file for training. This file contains the file paths of our custom dataset i.e., the training images, annotations for training images (done in YOLO format) and testing images respectively.

d) Then run the train.py file which contains the training code by customizing the batch size based on the capacity of the memory we have. Here we will be using Yolov5x weights as it is the best version in terms of both mAP and speed, as mentioned in the repository compared to other versions like v5n, v5s, v5m and v5l.

3.5 Evaluation of the object detection models:

Our use case is single class object detection problem. To check the performance metrics, we need to measure by comparing the predicted bounding boxes with the ground truth box, by classifying whether our model correctly detected the object (True Positive), not detected the object (False Negative) and incorrectly detected the object (False positive).

True Negative is not used in object detection like in classification problems, because there can be an infinite no. of possibilities of why a particular object is not available in a frame, which adds no benefit.

Briefly about the most widely used metrics for the object detection (Padilla, 2021):

- **Intersection over union (IOU):**

  This metric evaluates the overlap between the predicted bounding box of an object and its ground truth bounding box as shown in the figure 3.1. Based on the threshold value (generally set to 50%, 75% or 95 %, depending on the use case), we can conclude whether the detection is valid one (True positive) or invalid one (False positive).
- **Precision:**

  It is the ratio of correct positive predictions to the total no. of predictions, i.e., the ratio of True positives and (True positives + False positives).
- **Recall:**

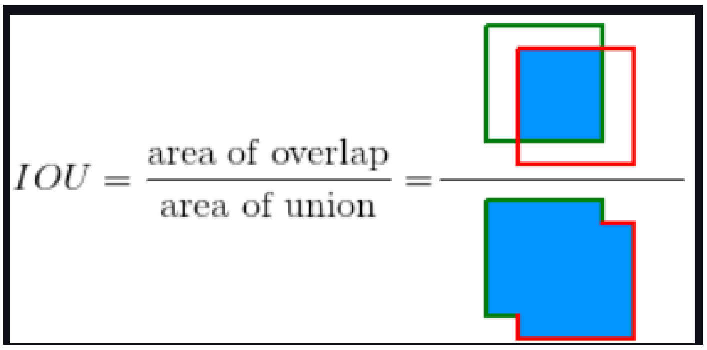
  It is the ratio of correct positive predictions to the total no. of ground truth values, i.e., the ratio of True positives and (True positives + False negatives).
- **Average Precision (AP):**

  AP is obtained by averaging precision across all recall values between 0 and 1. This metric can be used to compare different detectors by plotting their precision × recall curves in a single plot and performing interpolation, so that we can easily determine the area under curve, as shown in the figure 3.2.

  There are two interpolation methods, one is 11 – point interpolation, in which we try to summarize the shape of a precision x recall curve, by averaging the precision across 11equally spaced recall levels. The other interpolation method is averaging precision across all the recall levels. Then we divide the AUC into small rectangular parts, calculating all the areas and will sum up to get the total area under curve to get the average precision value.

- **Mean Average Precision (mAP):**
  It is one of the popular metrics used to evaluate the performance of an object detection model. It is calculated by computing the mean of the average precision (AP) over the number of classes, as shown in the equation 3.1 In our use case, the value of N will be 1, as there's only one class i.e., "pallet_box".

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \qquad \text{Eqn. 3.1}$$



$IOU = \frac{\text{area of overlap}}{\text{area of union}} =$

Figure 3.1 Intersection over union explanation (Padilla, 2021)

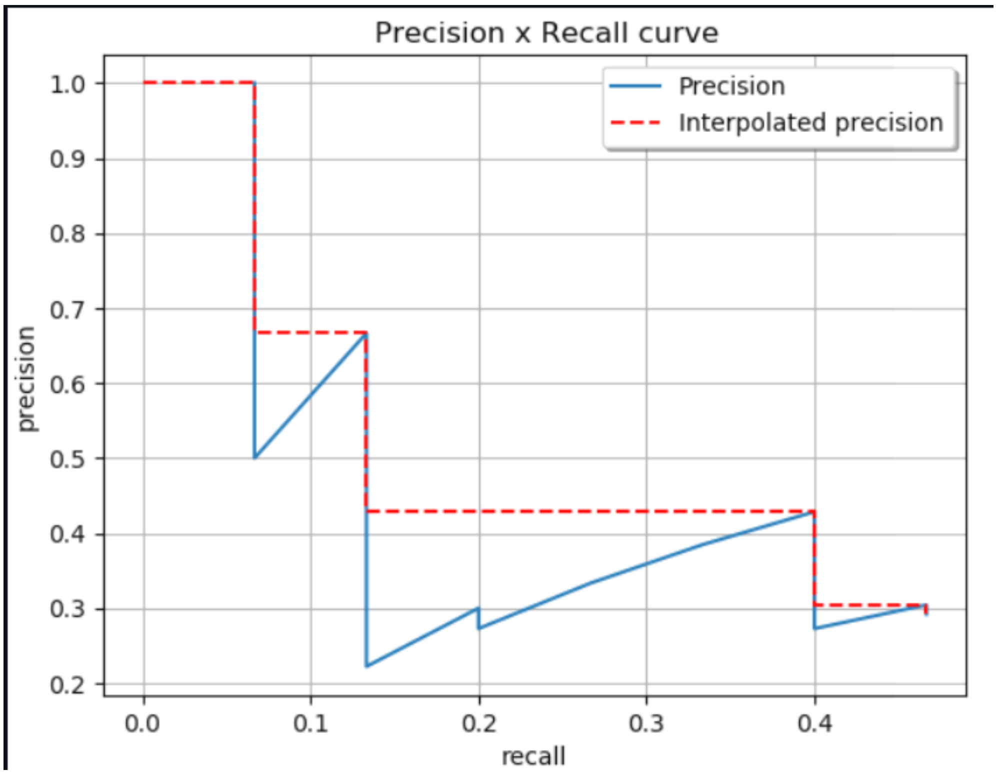


Figure 3.2 Calculating average precision using Precision x Recall curve (Padilla, 2021)

We evaluate our two object detection models using the above-mentioned metrics and choose the best one, which can be used to deploy in the object tracker algorithm.

3.6 Custom object tracking methodology:

In this study, we propose an object tracking methodology, which can be used in tracking and counting the pallets in the videos by assigning unique ID to each object. In practice, we use opencv-python library to execute this task. For an easy explanation, considered only one object, but this methodology can be applied on multiple objects as well.

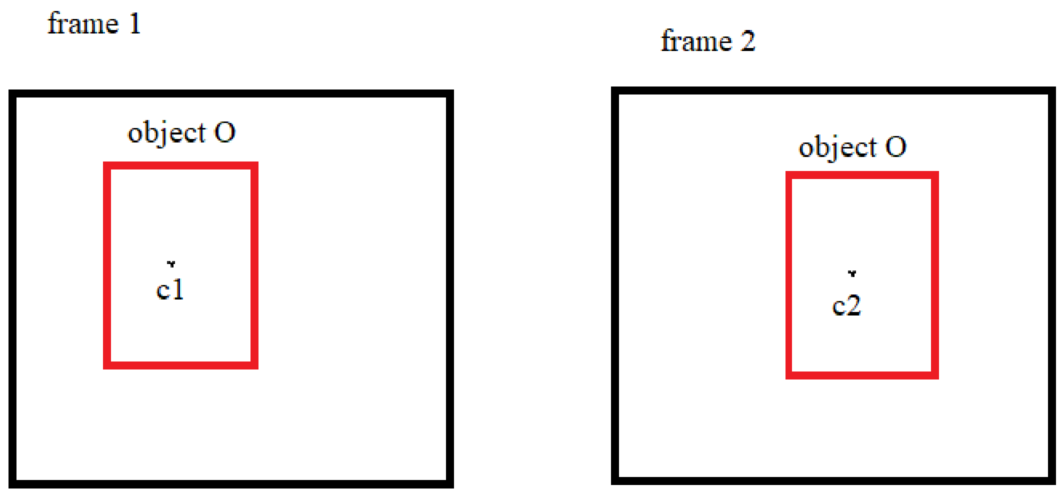


Figure 3.3 Illustration of the object tracking methodology

The methodology is as follows:

1) In each image, make sure the coordinates of bounding boxes are in this format (x,y,w,h) Where (x,y) indicates centre point, w and h indicate the width and height of the bounding box respectively.
2) Then, consider two successive frames in a video, detect the object O, in each one of them.
3) Assign a tracking ID for that object O, say i1
4) Fetch the centre points of the objects in the two frames and store them in two separate variables say c1 and c2 as shown in the figure 6.3.
5) Calculate the Euclidean distance between c1 and c2 say, d.
6) Define a threshold distance t.
7) If d is less than t, then keep the tracking ID i1.
8) If d is greater than t, then assign a new ID say id2 to the object O in the latter frame. So, the object in the latter frame is treated as a separate object.

9) If the object disappears in the latter frame, then delete the id1 which was assigned to it in the former frame.
10) Then, store all the unique tracking IDs of all the objects along with their respective centre points, as a key value pairs in a python dictionary and count them to get the number of items tracked.

This entire iteration runs on all the frames one by one, captured from the given video.

Here's the Pseudo code,

```
c1 = (x1, y1)    # centre of object O in frame 1
# Assign tracking Id as id1 for object O in both the frames
f1 = id1
f2 = id1
track_ids = {}  # an empty python dictionary or a container that stores the tracking ids.
c2 = (x2, y2)   # centre of object O in frame 2
while object O in frame 2:
    # Calculate the Euclidean distance between c1 and c2
    $d = \sqrt{((x1-x2)^2 – (y1-y2)^2)}$
    #Defining a threshold distance t
    t = 3 units
    # here t = 3 is taken just as an example. We need to define it by performing  experiments.
    # Condition to check
    if object_exists == True:
        if d is less than t:
            f1 = f2
            track_ids[f2] = c2  # updating the track_ids dictionary
        else:
            #Assign new id to the f2
            f2 = id2
            track_ids[f2] = c2
    else:
        track_ids.pop(f1)  # if object doesn't exist remove the track id from the track_ids
```

3.7 Implementation of our object tracking methodology and evaluation:

The methodology that we proposed, will be implemented and its result will be compared with another object tracker, which will be developed by using DeepSORT algorithm, on the same video. Our intention of proposing this kind of algorithm for tracking is to experiment with the identity switches, the number of false positives and the false negatives, based on that we can improve its performance.

We will use the best detectors weights, and couple with our custom tracker to perform the object detection, tracking and counting operations. We use the metric called multiple objects tracking accuracy (MOTA) to evaluate our tracker's performance and make necessary changes to improve its performance (Kumar et al., 2021b).

3.8 Summary:

To summarize the methodology,

1) Prepare the image dataset suitable for the analyses, by labelling the objects with the bounding boxes with “pallet_box” as a label. Use labelImg tool to execute this task.
2) Object detection task: Train the models using the Yolov4 and Yolov5 one by one on the training dataset. Evaluate the performance metrics like Intersection over union, Average precision, of the two models, then choose the best one out of them.
3) Object tracking task: Using the proposed object tracking methodology, implement it on the videos and check the multiple objects tracking accuracy or MOTA score to evaluate the tracker’s performance, compare its performance with another object tracker developed using DeepSORT algorithm, and make improvements to it if necessary.

CHAPTER 4

IMPLEMENTATION

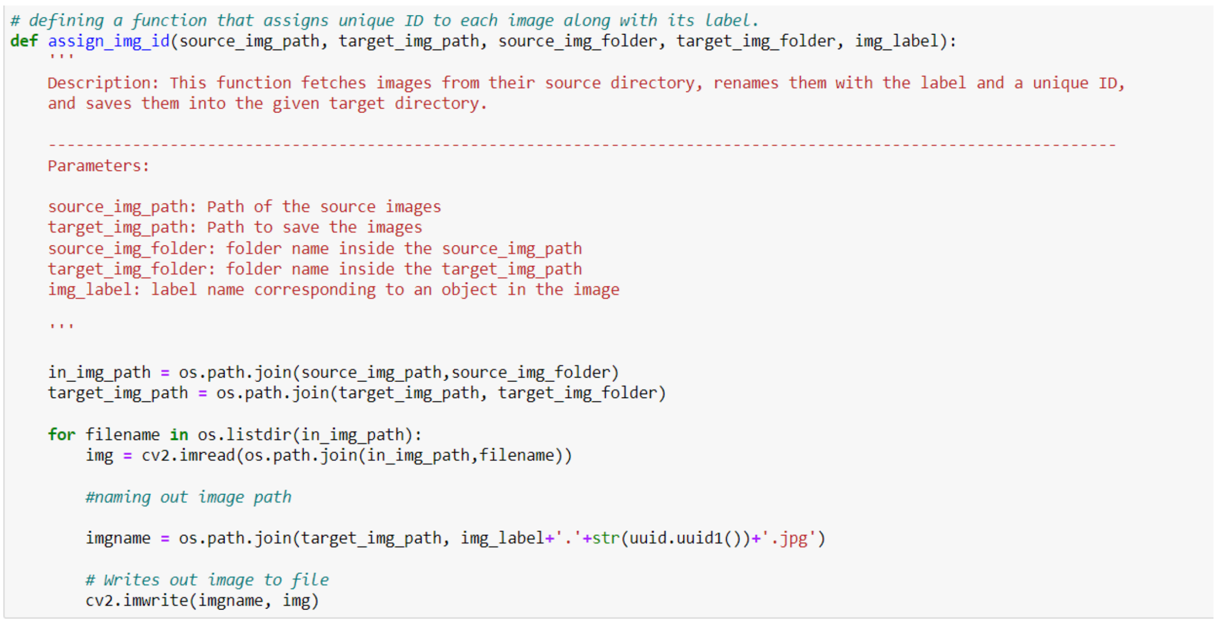
4.1 Dataset understanding:

The dataset used in this research is an open-source contribution by iNeuron.ai. This dataset consists of image and video data of the pallet boxes. All the images are having 416 × 416 dimensions and are present in the folder "train_data", in which they were already split into training (containing 639 images) and validation sets (containing 18 images) along with their respective label or ground truth data, hence making the overall count to 657. All these images were pre-processed i.e., every image is having its copy images which are blurred and in dark contrast. These images are used to train the object detection models and the videos are used for performing the object detection, tracking, and counting tasks.

4.2 Image data preparation:

Training an object detection model requires a lot more data. In the raw dataset, the number of images is low such that the model may be overfit on it. First, all the training and validation sets are merged into one single folder. Then, image re sampling operation is performed, to increase the size of the dataset. In this research, the size of the image data doubled, i.e., size increased from 657 to 1314. Then, all these images are randomly split into the train and validation sets respectively, by moving 97% of them into a "train" folder while the rest of them are moved to "val" folder.

Then, by using a library called "uuid" in python, all the images in both the "train" and "val" folders were renamed with unique ids and saved. The format of each image file name looks like this "<class_name>. <unique_id>.jpg". To make it clearer, one of the images was saved as "pallet.6c4f6159-cb01-11ec-b2aa-8045ddecc722.jpg". OpenCV's python library known as "opencv-python" is used to save the images to the target directory. The python function that was defined to perform this operation is shown in the figure 4.1. To perform this operation, this function must be called by specifying the following parameters: source_img_path, target_img_path, source_img_folder, target_img_folder, and img_label.



```python
# defining a function that assigns unique ID to each image along with its label.
def assign_img_id(source_img_path, target_img_path, source_img_folder, target_img_folder, img_label):
    '''
    Description: This function fetches images from their source directory, renames them with the label and a unique ID,
    and saves them into the given target directory.

    ---------------------------------------------------------------------------------------------------------------
    Parameters:

    source_img_path: Path of the source images
    target_img_path: Path to save the images
    source_img_folder: folder name inside the source_img_path
    target_img_folder: folder name inside the target_img_path
    img_label: label name corresponding to an object in the image

    '''

    in_img_path = os.path.join(source_img_path,source_img_folder)
    target_img_path = os.path.join(target_img_path, target_img_folder)

    for filename in os.listdir(in_img_path):
        img = cv2.imread(os.path.join(in_img_path,filename))

        #naming out image path

        imgname = os.path.join(target_img_path, img_label+'.'+str(uuid.uuid1())+'.jpg')

        # Writes out image to file
        cv2.imwrite(imgname, img)
```

Figure 4.1 A python function that assigns unique id to each image.

4.3 Image data labelling:

In a supervised machine learning approach, a model learns the patterns in a data by comparing its predictions with the ground truth data. Then it computes the error or loss and tries to minimize it to provide the accurate predictions. In solving the object detection problem, a model requires the ground truth information, i.e., the coordinates of all the objects present in an image, to learn the patterns in the given region of interest, make detections and compare the results with the corresponding ground truth values to compute loss and thereby trying to reduce the loss to provide accurate object detections. To label the images, there are many open-source tools available, and “labelImg” is one among them. It can be either downloaded by using “pip install labelImg” or can be cloned from the github repository using the following command- “git clone https://github.com/tzutalin/labelImg”. Once it is downloaded, it can be used by running the python file “labelImg.py” which opens an application. Then, the source directory where the images are available, and the target directory where the corresponding label data of the images must be saved, are to be specified by clicking on the “Open Dir” and “Change Save Dir” buttons respectively, and the labelling format must be set to “YOLO”. Following this, the image loads into the window from the source directory, and it can be labelled as shown in the figure 4.2.

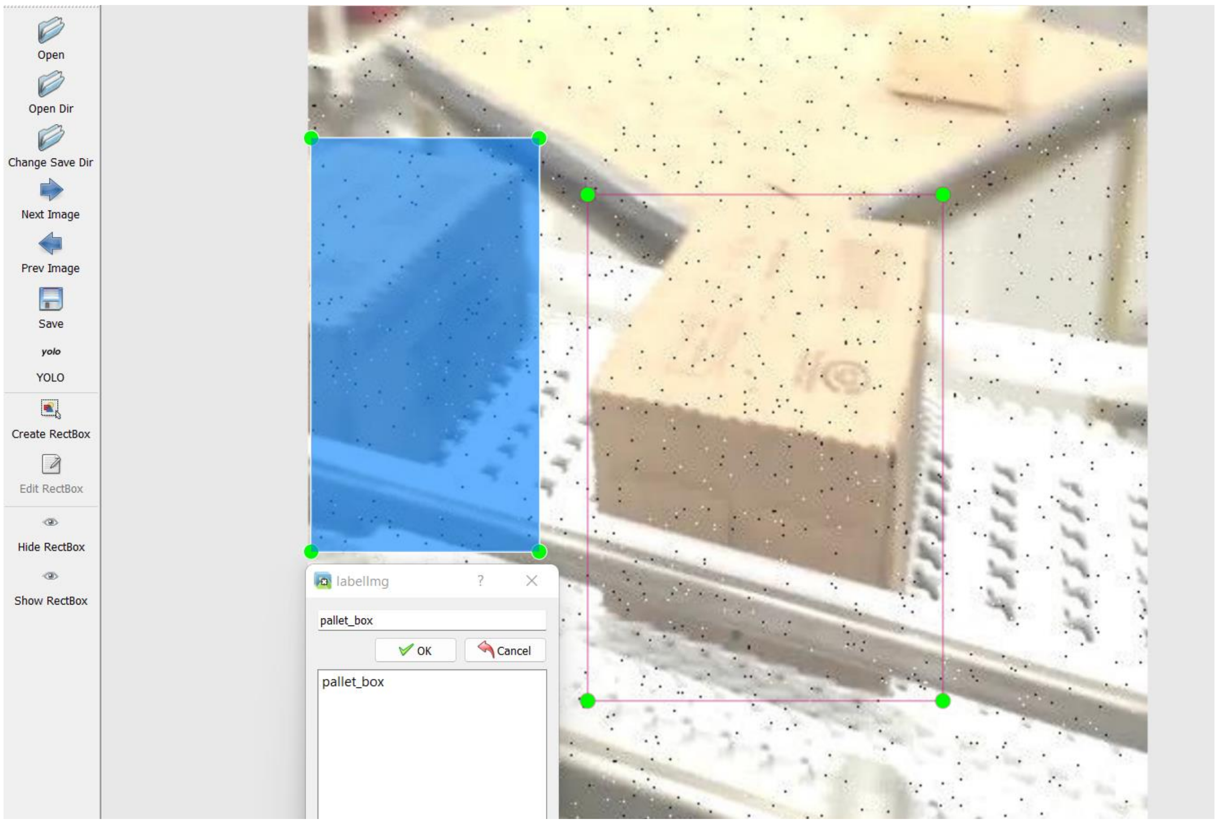


Figure 4.2 Labelling the images using labelImg tool.

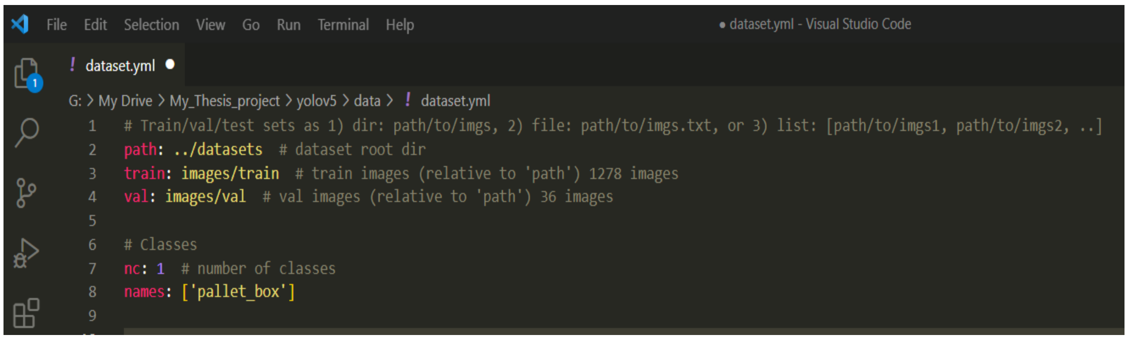
The label data of an image will be in the form of (<class_id> <x> <y> <w> <h>) where id is an identification number of a particular class, (x, y) is the centre point of the ground truth bounding box while w and h are its width and height values respectively. This labelling operation must be performed on all the training and validation images. Although, there is a labelling data available with the raw dataset, the labelling operation has been explicitly performed to make sure that each image is annotated correctly without any errors.

Next phase is training the object detection models. We trained two different models using Google colab.

4.4 Training YOLOv5x model:

We started this training by cloning the official YOLOv5's repository from GitHub and installing the requirements. We used Pytorch library to train and evaluate the model. The following are the steps that we followed to train the custom YOLOv5 model:

1. Prepared the "dataset.yml" file, which contains the information of paths to the dataset and the number of classes, then copied that file inside the data directory of the yolov5 directory as shown in the figure 4.3. This file will be used to get the data from both the train and validation directories during the training process.



File Edit Selection View Go Run Terminal Help
● dataset.yml - Visual Studio Code
! dataset.yml ●
G: > My Drive > My_Thesis_project > yolov5 > data > ! dataset.yml

```yaml
# Train/val/test sets as 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) list: [path/to/imgs1, path/to/imgs2, ..]
path: ../datasets  # dataset root dir
train: images/train  # train images (relative to 'path') 1278 images
val: images/val  # val images (relative to 'path') 36 images

# Classes
nc: 1  # number of classes
names: ['pallet_box']
```

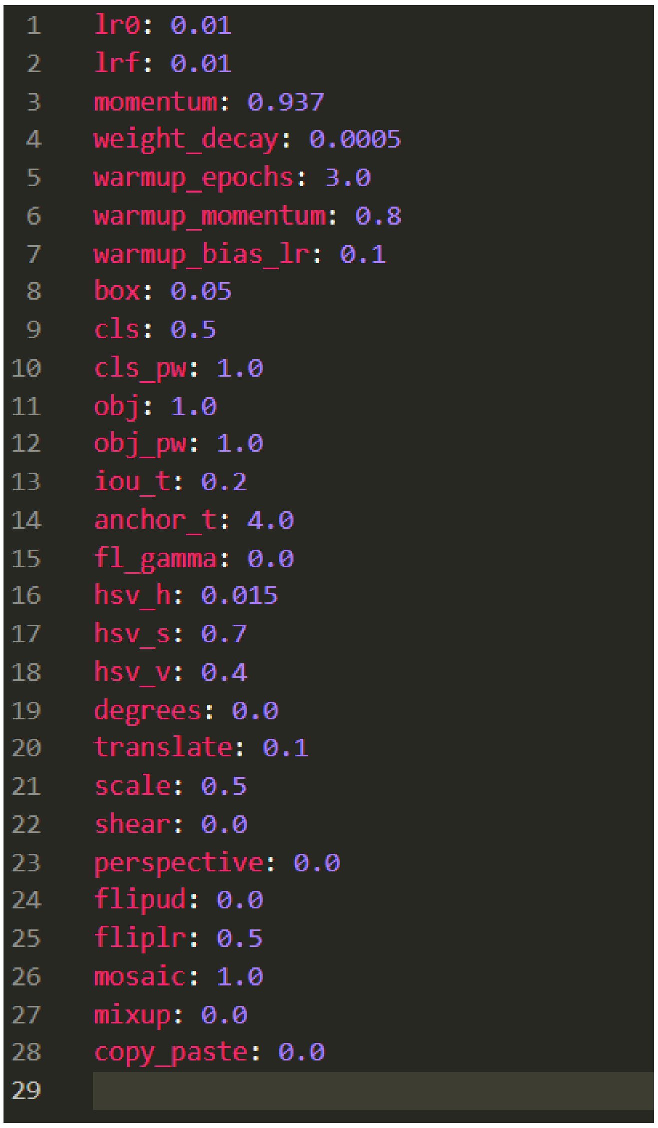
Figure 4.3 dataset.yml file for training the YOLOv5 model.

2. Then we set the following parameters - image size, batch size and number of epochs in the train.py file, specified the dataset.yml file and chose "yolov5x.pt" as the weights file, then we ran it to start training the model. This process followed a series of experiments as shown in table 4.1 and finally set the batch size to 12 and number of epochs to 20, while maintaining the image size 640. The trained model weights and results are saved in the runs folder in YOLOv5 folder

Table 4.1 YOLOv5x model training experiments

Exp No.	**Batch size**	**No. of epochs**	**Outcome**
1	16	50	Error: Cuda out of memory
2	8	50	Training took more time, but it was interrupted as the it exceeded the GPU usage limits.
3	12	20	Training successful

3. We used default hyper parameters to train the model as shown in the figure 4.4, in which, the learning rate set to 0.01, IOU training threshold set to 0.2, box loss gain set to 0.05, class loss gain set to 0.5, and the probability of fliplr set to 0.5 while the probability of flipud set to 0 as flipping the images upside down is not required for our use case.



```
lr0: 0.01
lrf: 0.01
momentum: 0.937
weight_decay: 0.0005
warmup_epochs: 3.0
warmup_momentum: 0.8
warmup_bias_lr: 0.1
box: 0.05
cls: 0.5
cls_pw: 1.0
obj: 1.0
obj_pw: 1.0
iou_t: 0.2
anchor_t: 4.0
fl_gamma: 0.0
hsv_h: 0.015
hsv_s: 0.7
hsv_v: 0.4
degrees: 0.0
translate: 0.1
scale: 0.5
shear: 0.0
perspective: 0.0
flipud: 0.0
fliplr: 0.5
mosaic: 1.0
mixup: 0.0
copy_paste: 0.0

```

Figure 4.4 Hyperparameters set to train the YOLOv5x model.

At this point, we successfully trained our first model.

4.5 Training YOLOv4 model:

We started this training by cloning the Darknet repository from the GitHub and the following steps are followed to train the custom YOLOv4 model:

1. We merged all the images along with the respective label files from train and validation datasets into one folder called, “obj” and copied that folder in the data directory inside the darknet folder as shown in the figure 4.5.

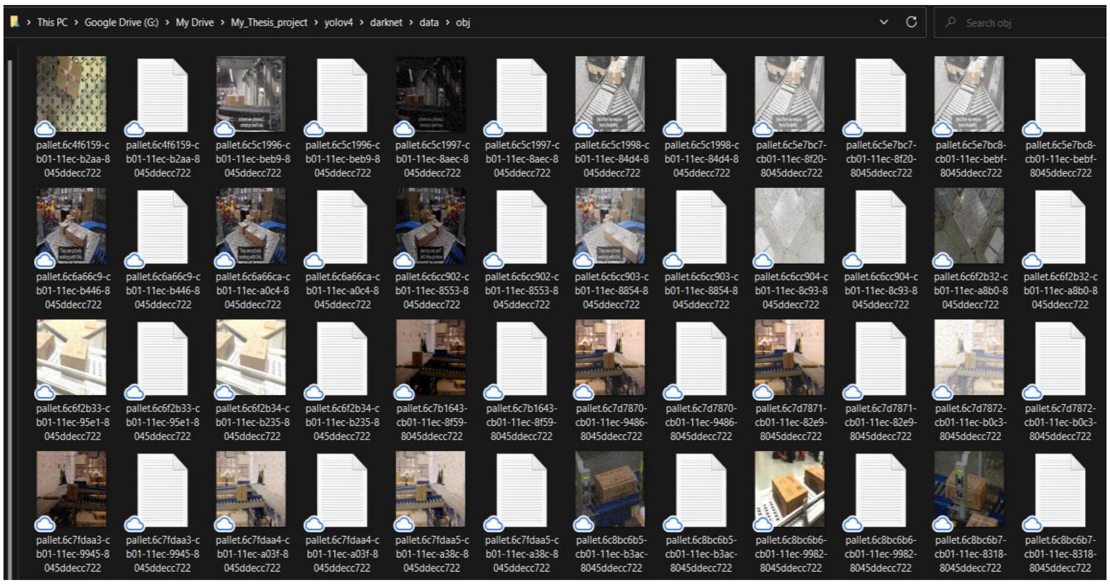

This PC › Google Drive (G:) › My Drive › My_Thesis_project › yolov4 › darknet › data › obj
Search obj
pallet.6c4f6159-cb01-11ec-b2aa-8045ddecc722
pallet.6c4f6159-cb01-11ec-b2aa-8045ddecc722
pallet.6c5c1996-cb01-11ec-beb9-8045ddecc722
pallet.6c5c1996-cb01-11ec-beb9-8045ddecc722
pallet.6c5c1997-cb01-11ec-8aec-8045ddecc722
pallet.6c5c1997-cb01-11ec-8aec-8045ddecc722
pallet.6c5c1998-cb01-11ec-84d4-8045ddecc722
pallet.6c5c1998-cb01-11ec-84d4-8045ddecc722
pallet.6c5e7bc7-cb01-11ec-8f20-8045ddecc722
pallet.6c5e7bc7-cb01-11ec-8f20-8045ddecc722
pallet.6c5e7bc8-cb01-11ec-bebf-8045ddecc722
pallet.6c5e7bc8-cb01-11ec-bebf-8045ddecc722
pallet.6c6a66c9-cb01-11ec-b446-8045ddecc722
pallet.6c6a66c9-cb01-11ec-b446-8045ddecc722
pallet.6c6a66ca-cb01-11ec-a0c4-8045ddecc722
pallet.6c6a66ca-cb01-11ec-a0c4-8045ddecc722
pallet.6c6cc902-cb01-11ec-8553-8045ddecc722
pallet.6c6cc902-cb01-11ec-8553-8045ddecc722
pallet.6c6cc903-cb01-11ec-8854-8045ddecc722
pallet.6c6cc903-cb01-11ec-8854-8045ddecc722
pallet.6c6cc904-cb01-11ec-8c93-8045ddecc722
pallet.6c6cc904-cb01-11ec-8c93-8045ddecc722
pallet.6c6f2b32-cb01-11ec-a8b0-8045ddecc722
pallet.6c6f2b32-cb01-11ec-a8b0-8045ddecc722
pallet.6c6f2b33-cb01-11ec-95e1-8045ddecc722
pallet.6c6f2b33-cb01-11ec-95e1-8045ddecc722
pallet.6c6f2b34-cb01-11ec-b235-8045ddecc722
pallet.6c6f2b34-cb01-11ec-b235-8045ddecc722
pallet.6c7b1643-cb01-11ec-8f59-8045ddecc722
pallet.6c7b1643-cb01-11ec-8f59-8045ddecc722
pallet.6c7d7870-cb01-11ec-9486-8045ddecc722
pallet.6c7d7870-cb01-11ec-9486-8045ddecc722
pallet.6c7d7871-cb01-11ec-82e9-8045ddecc722
pallet.6c7d7871-cb01-11ec-82e9-8045ddecc722
pallet.6c7d7872-cb01-11ec-b0c3-8045ddecc722
pallet.6c7d7872-cb01-11ec-b0c3-8045ddecc722
pallet.6c7fdaa3-cb01-11ec-9945-8045ddecc722
pallet.6c7fdaa3-cb01-11ec-9945-8045ddecc722
pallet.6c7fdaa4-cb01-11ec-a03f-8045ddecc722
pallet.6c7fdaa4-cb01-11ec-a03f-8045ddecc722
pallet.6c7fdaa5-cb01-11ec-a38c-8045ddecc722
pallet.6c7fdaa5-cb01-11ec-a38c-8045ddecc722
pallet.6c8bc6b5-cb01-11ec-b3ac-8045ddecc722
pallet.6c8bc6b5-cb01-11ec-b3ac-8045ddecc722
pallet.6c8bc6b6-cb01-11ec-9982-8045ddecc722
pallet.6c8bc6b6-cb01-11ec-9982-8045ddecc722
pallet.6c8bc6b7-cb01-11ec-8318-8045ddecc722
pallet.6c8bc6b7-cb01-11ec-8318-8045ddecc722

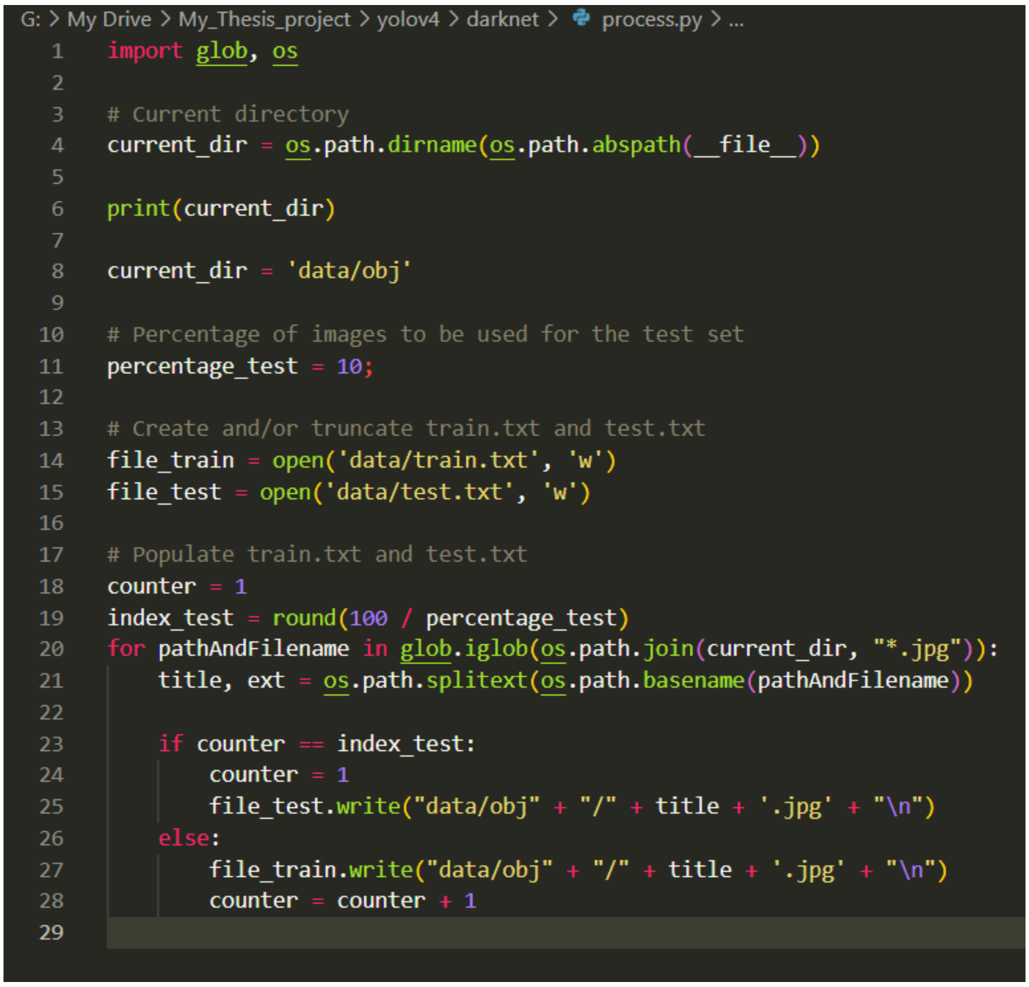
Figure 4.5 Organizing the data for training the YOLOv4 model

2. Next step is to organize the image files into training and testing groups with in 9:1 ratio. We did this by using a python code “process.py”, which created two files – train.txt and test.txt, containing the image paths. In other words, train.txt contains the image paths which will be used for training and test.txt contains the image paths which will be used for validation purposes. These two files are copied in the data directory inside the darknet folder The python code inside the process.py file, is as shown in the figure 4.6.

3. We created two files – obj.names and obj.data. The former file contains an information related to the classes while the latter file contains an information about the number of classes and paths, as shown in the figure 4.7. These two files are also copied in in the data directory inside the darknet folder.

4. In the data directory, till this point we had 5 items – obj folder that stores all the images with respective label files, obj.names, obj.data, train.txt and test.txt. Then we created an

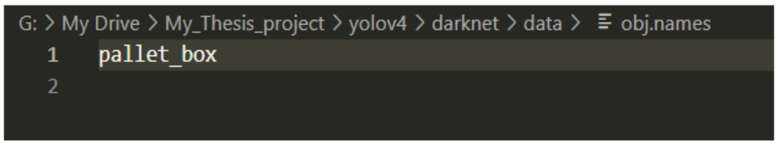
empty directory, “labels” which will be used to store the image files by the algorithm during the training.



G: > My Drive > My_Thesis_project > yolov4 > darknet > process.py > ...

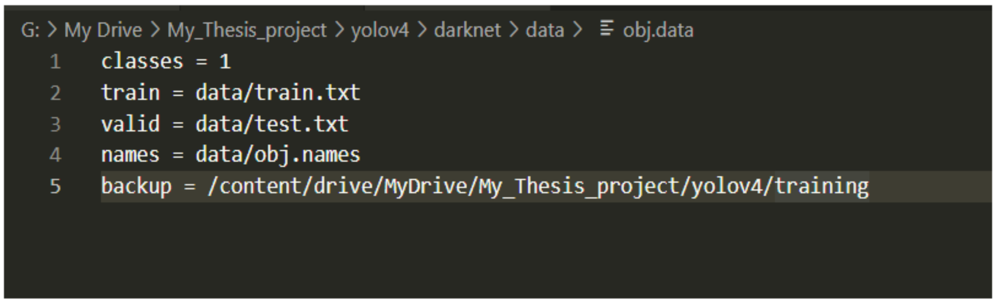
```python
import glob, os

# Current directory
current_dir = os.path.dirname(os.path.abspath(__file__))

print(current_dir)

current_dir = 'data/obj'

# Percentage of images to be used for the test set
percentage_test = 10;

# Create and/or truncate train.txt and test.txt
file_train = open('data/train.txt', 'w')
file_test = open('data/test.txt', 'w')

# Populate train.txt and test.txt
counter = 1
index_test = round(100 / percentage_test)
for pathAndFilename in glob.iglob(os.path.join(current_dir, "*.jpg")):
    title, ext = os.path.splitext(os.path.basename(pathAndFilename))

    if counter == index_test:
        counter = 1
        file_test.write("data/obj" + "/" + title + '.jpg' + "\n")
    else:
        file_train.write("data/obj" + "/" + title + '.jpg' + "\n")
        counter = counter + 1

```

Figure 4.6 Python code that organizes the images into train and test sets.



G: > My Drive > My_Thesis_project > yolov4 > darknet > data > obj.names

```
pallet_box

```



G: > My Drive > My_Thesis_project > yolov4 > darknet > data > obj.data

```
classes = 1
train = data/train.txt
valid = data/test.txt
names = data/obj.names
backup = /content/drive/MyDrive/My_Thesis_project/yolov4/training
```

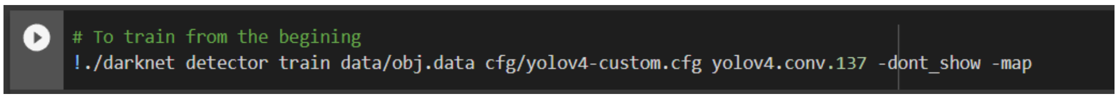
Figure 4.7 obj.names and obj.data files for training the YOLOv4 model.

5. Then, we made changes to the “yolov4_custom.cfg” file, present inside the cfg folder in the darknet directory, by setting the learning rate to 0.001, batch size to 64, subdivisions to 16, width to 416, height to 416 and max batches to 2000. Since we are dealing with the single class object detection problem, we set the classes to 1 in all the 3 yolo layers and set the number of filters to 18 in every convolution layer before each yolo layer. This was calculated using the formula (classes + 5) × 3.

6. We made changes to the “make” file in the darknet directory to OpenCV, GPU and CUDNN. Then we ran make file, which builds darknet.

7. Instead of training the entire yolov4 model from scratch, we followed transfer learning approach. We downloaded the pretrained yolov4 weights file - “yolov4.conv.137”, which was trained up to 137 convolutional layers, useful to train our custom object detection model.

8. Then we started training our model by running the code as shown in the figure 4.8. All the trained weights are saved in the backup folder called “training”, which can be used to make detections on the images, while the training results are stored in the darknet folder.



```
# To train from the begining
!./darknet detector train data/obj.data cfg/yolov4-custom.cfg yolov4.conv.137 -dont_show -map
```

Figure 4.8 Code to train the yolov4 model.

9. In case if the model training gets interrupted, we can continue it by using the last saved weights file.

At this point, we successfully trained our second model. The mean average precisions of both YOLOv5x and YOLOv4 models are in the range between 0.95 and 1. So, both are performing equally well. The more details on the results will be discussed in the Chapter 5.

4.6 Object tracking and counting using custom object tracker:

We implemented our proposed methodology to develop custom object tracker with the help of opencv-python library in Pycharm and jupyter notebook, in our local Windows machine. First, we need to input a video, either a live one or a recorded one using opencv method called "cv2.VideoCapture()" Then the entire process runs by iterating through each frame. This process is comprised of two parts:

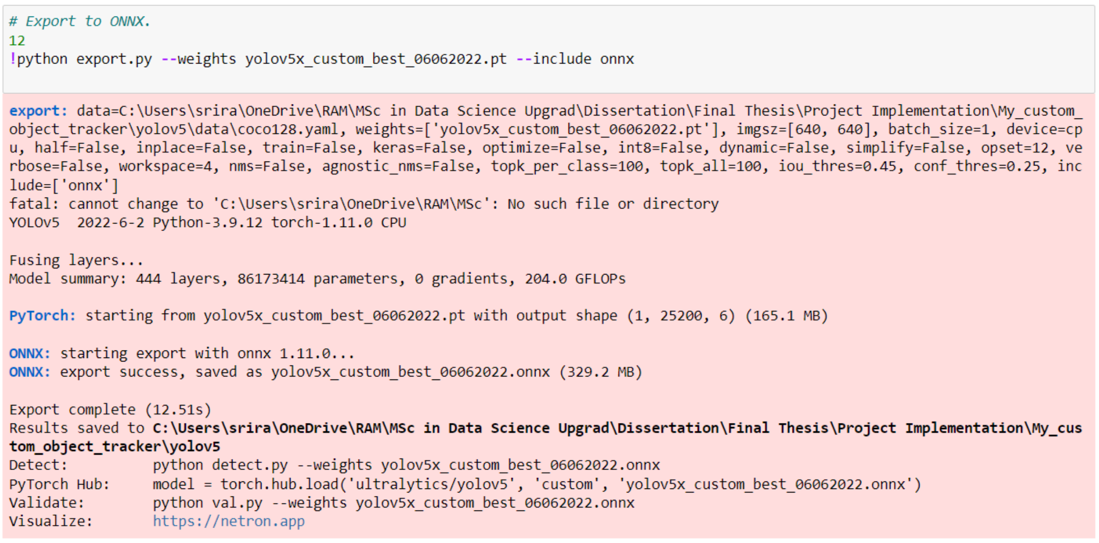
1. Detecting objects in a frame.
2. Tracking and counting the objects in a frame.

4.6.1 Detecting objects in a frame:

The objects in a frame are detected by the object detection model. As discussed earlier, the two object detection models developed by us are performing equally well. Out of them we chose YOLOv5x model because the weights are saved in Pytorch form i.e.,with ".pt" extension, it can be easily used to make detections on the images. But we can't directly deploy it on OpenCV by using the torch.hub.load() method, as we tried, the model could not make the detections properly due to the difference in the interpretation between Pytorch and OpenCV libraries. So, we did as follows by taking one tutorial as a reference (link: - https://learnopencv.com/object-detection-using-yolov5-and-opencv-dnn-in-c-and-python/) :

1. OpenCV has a DNN module, that can be used to perform inference in Deep learning. We can pass our trained object detection model weights into this module which returns the model as an output. But DNN module in OpenCV do not support for .pt weights. So, we need to convert our YOLOv5x model weights file, which is in .pt format into the ONNX format, as shown in the figure 4.9. Then, we call the method, "cv2.dnn.readNet()" by passing the path related to the model weights in ONNX format, which returns the model.
2. We developed a module called "object_detection.py" and inside that defined a class called "ObjectDetection". This class is defined with some methods as illustrated in the figure 4.10, and by calling those methods, the detections made by the model a get processed.

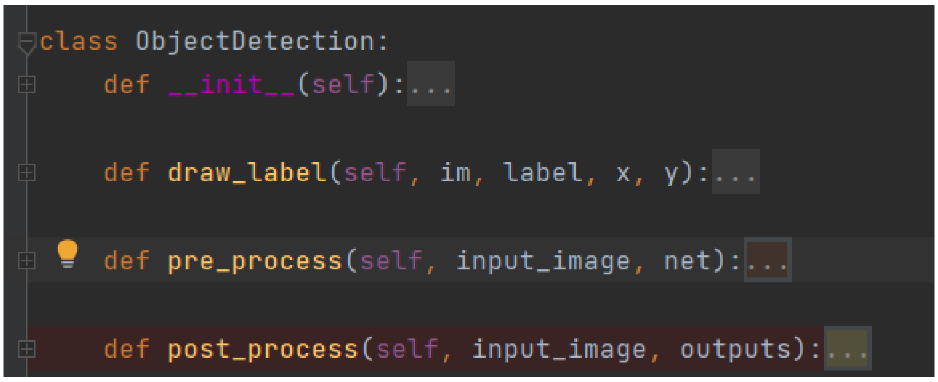
3. This class is initialized with some constant parameters – input_width = 640, input_height = 640, score_threshold = 0.5, NMS_threshold = 0.45 and confidence_threshold = 0.45. These parameters will be used by the methods to process the given input image.



```
# Export to ONNX.
12
!python export.py --weights yolov5x_custom_best_06062022.pt --include onnx
```

```
export: data=C:\Users\srira\OneDrive\RAM\MSc in Data Science Upgrad\Dissertation\Final Thesis\Project Implementation\My_custom_
object_tracker\yolov5\data\coco128.yaml, weights=['yolov5x_custom_best_06062022.pt'], imgsz=[640, 640], batch_size=1, device=cp
u, half=False, inplace=False, train=False, keras=False, optimize=False, int8=False, dynamic=False, simplify=False, opset=12, ve
rbose=False, workspace=4, nms=False, agnostic_nms=False, topk_per_class=100, topk_all=100, iou_thres=0.45, conf_thres=0.25, inc
lude=['onnx']
fatal: cannot change to 'C:\Users\srira\OneDrive\RAM\MSc': No such file or directory
YOLOv5  2022-6-2 Python-3.9.12 torch-1.11.0 CPU

Fusing layers...
Model summary: 444 layers, 86173414 parameters, 0 gradients, 204.0 GFLOPs

PyTorch: starting from yolov5x_custom_best_06062022.pt with output shape (1, 25200, 6) (165.1 MB)

ONNX: starting export with onnx 1.11.0...
ONNX: export success, saved as yolov5x_custom_best_06062022.onnx (329.2 MB)

Export complete (12.51s)
Results saved to C:\Users\srira\OneDrive\RAM\MSc in Data Science Upgrad\Dissertation\Final Thesis\Project Implementation\My_cus
tom_object_tracker\yolov5
Detect:          python detect.py --weights yolov5x_custom_best_06062022.onnx
PyTorch Hub:     model = torch.hub.load('ultralytics/yolov5', 'custom', 'yolov5x_custom_best_06062022.onnx')
Validate:        python val.py --weights yolov5x_custom_best_06062022.onnx
Visualize:       https://netron.app
```

Figure 4.9 Conversion of YOLOv5x weights to ONNX format.



```
class ObjectDetection:
    def __init__(self):...

    def draw_label(self, im, label, x, y):...

    def pre_process(self, input_image, net):...

    def post_process(self, input_image, outputs):...
```

Figure 4.10 Structure of the ObjectDetection class

4. The draw_label method takes the frame, label text and the coordinates as input parameters, and draws the text box on to the frame at a location determined by the coordinates.

5. The pre_process method, takes the frame and the model as parameters. It first creates a 4D blob from the frame by normalizing the pixels and resizing the image to 640 × 640 dimensions. The obtained blob is used to set the inputs to the model, and when the model is run by using the method .getUnconnectedOutLayersNames(), it gets the features of

all the output layers through which the blob is forward propagated, and ultimately returns the detections.

6. The post_process method, takes the frame and the outputs or detections returned by the pre_process method as parameters. This method first resizes the frame by diving its height and weight with 640. Then by iterating through the detections, this method discards the detections whose confidence score and class score are less than the predefined thresholds confidence score and class score i.e., 0.45 and 0.5 respectively. For the good detections, it generates the bounding box coordinates. Then by performing the Non-Maximum Suppression, the redundant and overlapping boxes with lower confidence scores get eliminated, so that we will be left with the valid bounding box coordinates. From them the centre points are calculated. Then using those coordinates, bounding boxes are drawn on the frame indicating the object detections along with their corresponding confidence scores and label names using the draw_label method. Ultimately, this method returns the frame with object detections on it, and the centre point coordinates of those objects in the form of python list. These will be further used to run the custom object tracker.

4.6.2 Tracking and counting the objects in a frame:

This section is all about the working of our custom object tracker. At the beginning, i.e., for the first two frames, it runs as follows:

1. After detecting objects in a frame and getting the centre point coordinates of each object, we store them in a list called “centre_points_cur_frame”. Unique ids will be assigned to each object in a frame Before moving to the second frame, we copy the items in the centre_points_cur_frame list to another list called “centre_points_prev_frame”. So, after moving to the second frame, we calculate the Euclidean distance between the corresponding items present in the centre_points_cur_frame and centre_points_prev_frame.

2. If the calculated distance is less than the predefined threshold distance, which is 50 pixels (it is defined after a series of experiments), then it updates the ids of the corresponding objects into a dictionary “tracking_objects” as {“track_id”: “centre_point_of_an_object”}.

Then from the third frame onwards, it runs as follows:

1. It calculates the Euclidean distance between the centre points of the objects which are detected in the current frame, and the corresponding centre points of the objects in previous frame. Here the previous frame's data will be taken from the tracking_objects dictionary.
2. If the calculated distance is less than the predefined threshold distance, an object continues to have the same id, and the object's centre point coordinates gets updated in the tracking_objects dictionary with respect to its id.
3. Or else, an object gets a new id assigned and it will be updated in the tracking_objects dictionary along with its centre point by replacing its old id key, value pair.
4. If an object which was present in previous frame and is absent in current frame, then its id gets discarded from the tracking_objects dictionary.
5. If a new object appears in the current frame, it will be assigned a unique id and it will be updated in the tracking_objects dictionary.

In each frame, after tracking the objects, their corresponding tracking ids will be indicated on the frame as a center point, by using the cv2.circle() and cv2.putText() methods. The objects are counted in each frame by calculating the length of the tracking_objects dictionary, and will be displayed using the cv2,putText() method. Figure 4.11 shows the object tracking and counting operation performed on a video. The performance of this custom object tracker will be discussed in the chapter 5.

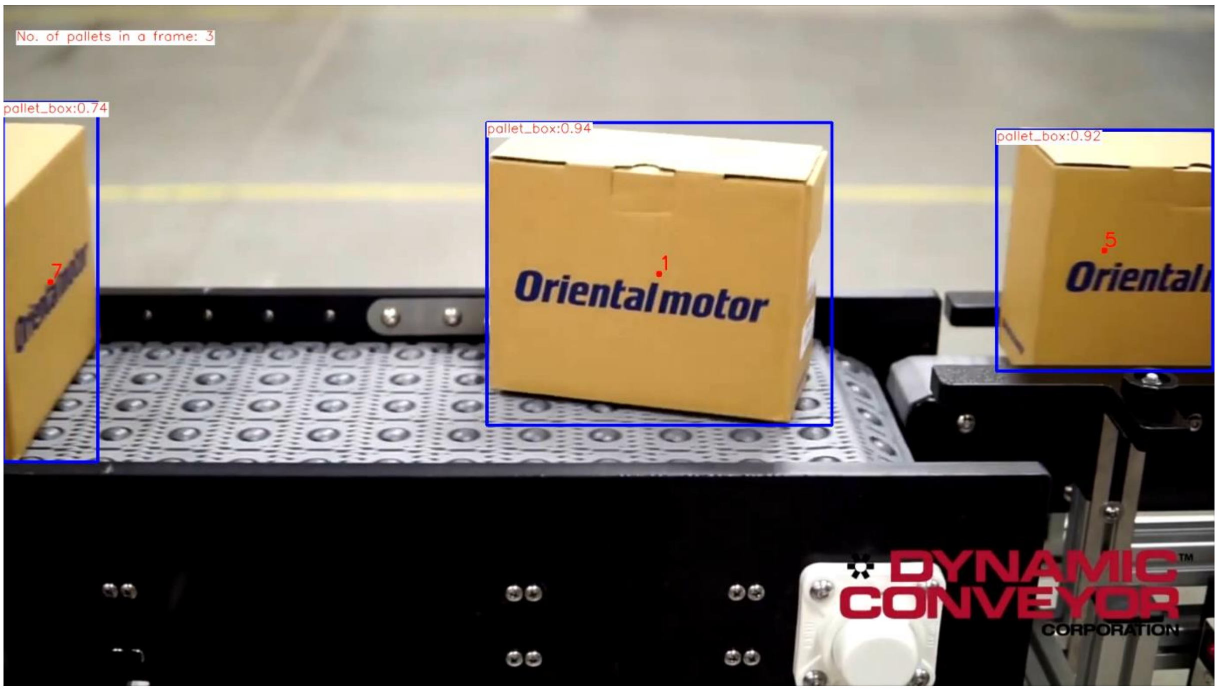


Figure 4.11 Performance of our custom object tracker.

This completes our project implementation part. A summary of what we did:

1. Image data preparation and labelling for our analysis.
2. Training custom YOLOv5x and custom YOLOv4 models.
3. Converting the YOLOv5x model into ONNX format, to deploy it by using OpenCV dnn module.
4. Using that model to make object detections on a given video, which was split into a series of frames, tracking and counting the objects using our custom object tracker.

CHAPTER 5

RESULTS AND DISCUSSION

This chapter discusses the results of the various tasks that were performed during the Implementation. This comprises of two parts:

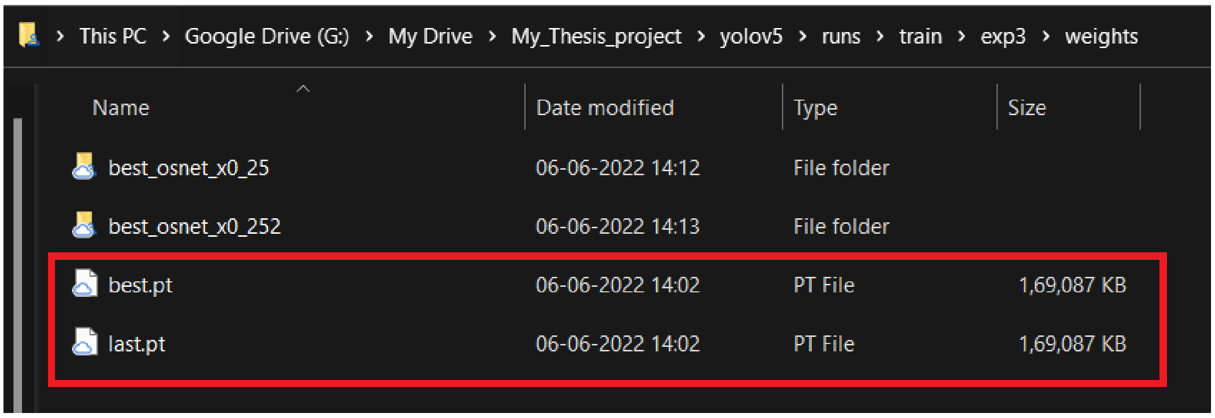
1. Results of the object detection models
2. Results of the custom object tracker

5.1 Results of the object detection models:

We trained two different object detection models, i.e., YOLOv5x model and the YOLOv4 model. These two models are analysed as mentioned below and we used Google colab:

5.1.1 YOLOv5x model:

After training the model by specifying the batch size as 12 and running for 20 epochs, its weights are stored in the storage as highlighted in the figure 5.1. We converted the "best.pt" weights file into the model by using the pytorch method torch.hub.load(). Then, defined a python helper function as shown in the figure 5.2, that can be used directly to make detections on any image.

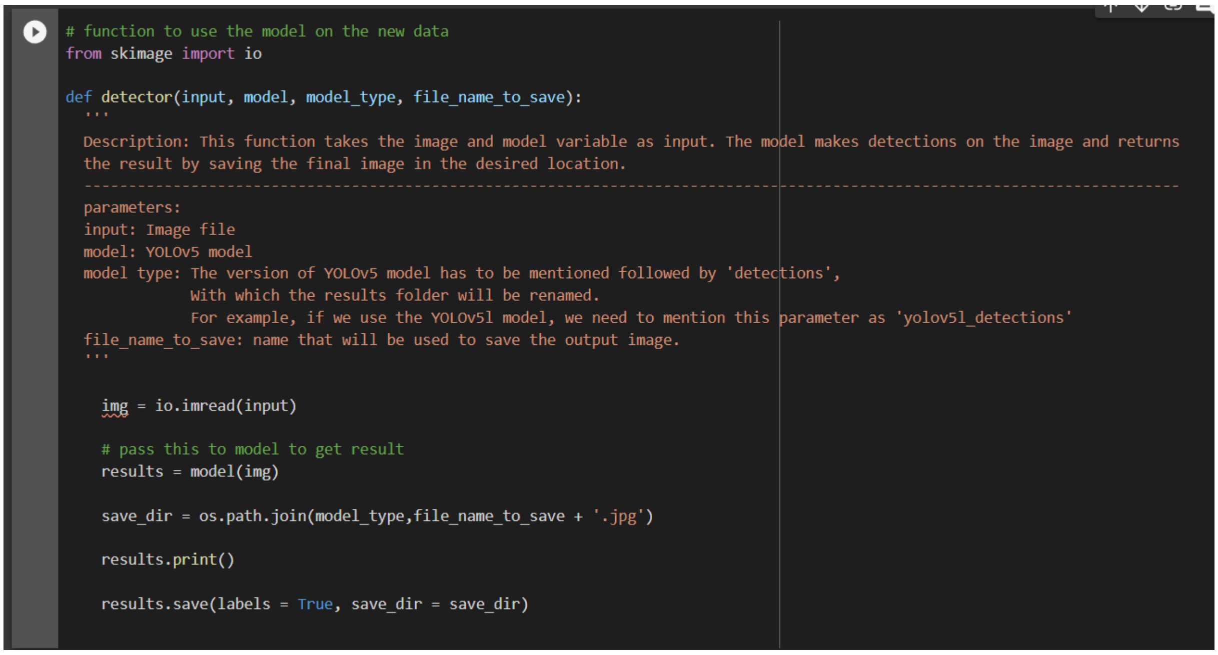


This PC › Google Drive (G:) › My Drive › My_Thesis_project › yolov5 › runs › train › exp3 › weights

Name	Date modified	Type	Size
best_osnet_x0_25	06-06-2022 14:12	File folder	
best_osnet_x0_252	06-06-2022 14:13	File folder	
best.pt	06-06-2022 14:02	PT File	1,69,087 KB
last.pt	06-06-2022 14:02	PT File	1,69,087 KB

Figure 5.1 Weight files of the YOLOv5x model

Then this function is used to make detections on a random test image. Figure 5.3 shows both the original image (on left) and the resultant image (on right), which is having object detection with a confidence score of 0.93.



```
# function to use the model on the new data
from skimage import io

def detector(input, model, model_type, file_name_to_save):
  '''
  Description: This function takes the image and model variable as input. The model makes detections on the image and returns
  the result by saving the final image in the desired location.
  ------------------------------------------------------------------------------------------------------------------------------
  parameters:
  input: Image file
  model: YOLOv5 model
  model type: The version of YOLOv5 model has to be mentioned followed by 'detections',
              With which the results folder will be renamed.
              For example, if we use the YOLOv5l model, we need to mention this parameter as 'yolov5l_detections'
  file_name_to_save: name that will be used to save the output image.
  '''

    img = io.imread(input)

    # pass this to model to get result
    results = model(img)

    save_dir = os.path.join(model_type,file_name_to_save + '.jpg')

    results.print()

    results.save(labels = True, save_dir = save_dir)
```

Figure 5.2 A python helper function to make detections using YOLOv5x model.

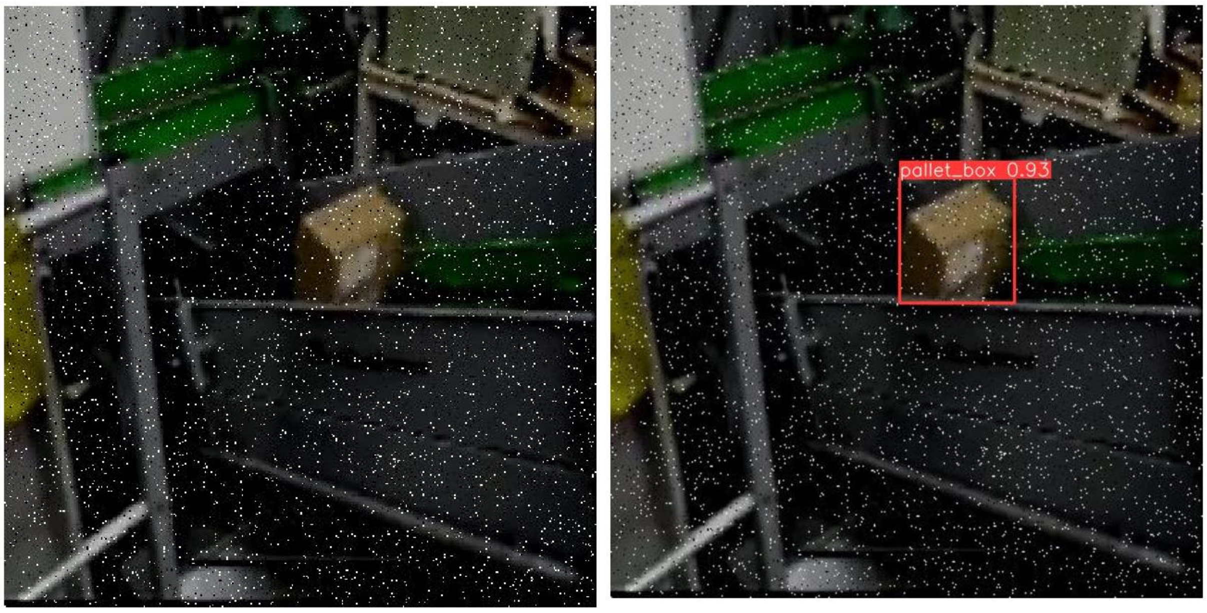


Figure 5.3 Object detection by YOLOv5x model on a random test image.

The overall performance of the model is illustrated as shown in the figure 5.4. Since we trained the model only on one class i.e., “pallet_box”, the class loss is 0. The box loss and object loss during both the training and validation processes, decreased gradually from the 1st epoch to the 20th epoch, while both the precision and recall increased within few epochs post the commencement of the training. The mean average precision at 0.5 IOU threshold ultimately settled at 0.995 while the mean average precision at 0.5:0.95 IOU threshold settled at 0.89. So, considering all these metrics, it is evident that our YOLOv5x model was trained well and can make accurate object detections on the pallet box images.

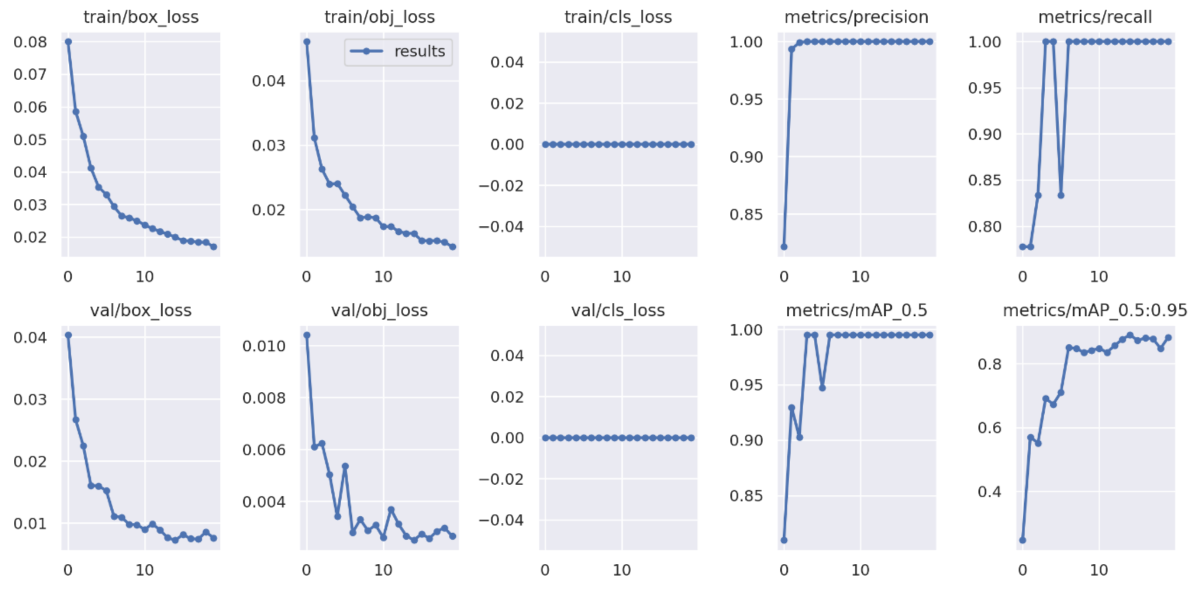


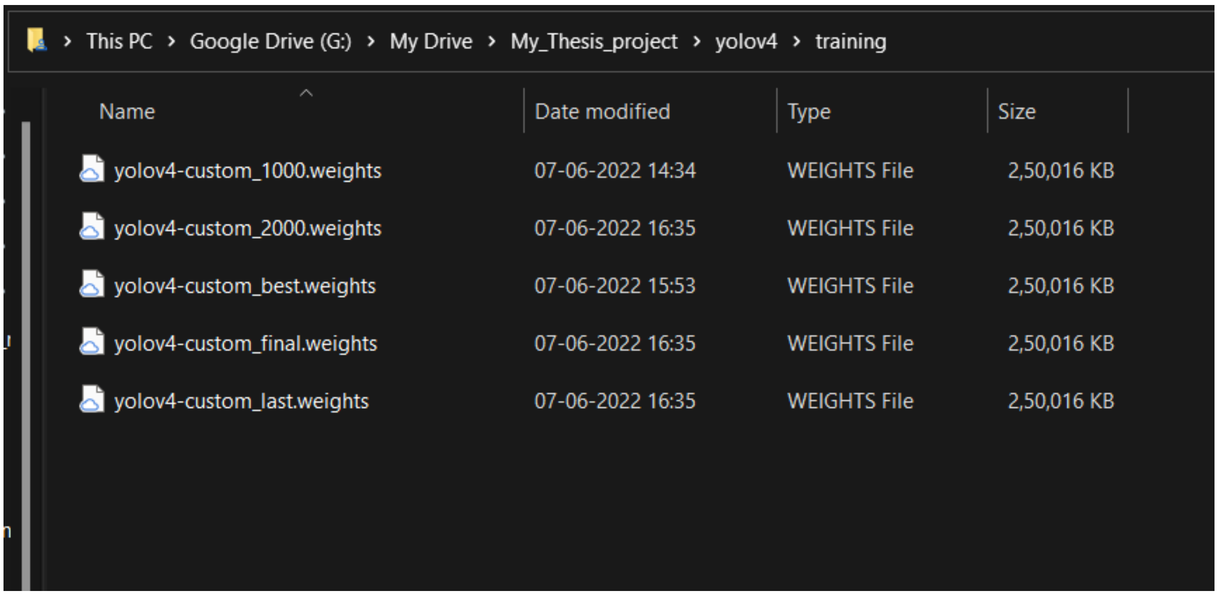
Figure 5.4 Overall performance of the YOLOv5x model

5.1.2 YOLOv4 model:

After training the model, the files containing the weights are stored in the storage as shown in the figure 5.5. Among them, we used “yolov4-custom_best.weights” for making detections. To do this, we made changes to the “yolov4_custom.cfg” file by setting the batch size to 1 and subdivisions to 1. This makes our model to work in test mode. Then we ran the model on the same test image, that we ran YOLOv5x model, in the Google colab by setting the IOU threshold as 0.5 and using the code as mentioned below:

```
!./darknet detector test data/obj.data cfg/yolov4_custom.cfg <file_path_to_the_ weights> <file_path_to_the_test_image> -thresh 0.5
```

Figure 5.6 shows both the original image (on left) and the resultant image (on right) which is obtained by running the model on the original image. The model detected the object with a confidence score of 1 i.e., with 100% accuracy. This shows that the YOLOv4 model is also performing well.


This PC › Google Drive (G:) › My Drive › My_Thesis_project › yolov4 › training

Name	Date modified	Type	Size
yolov4-custom_1000.weights	07-06-2022 14:34	WEIGHTS File	2,50,016 KB
yolov4-custom_2000.weights	07-06-2022 16:35	WEIGHTS File	2,50,016 KB
yolov4-custom_best.weights	07-06-2022 15:53	WEIGHTS File	2,50,016 KB
yolov4-custom_final.weights	07-06-2022 16:35	WEIGHTS File	2,50,016 KB
yolov4-custom_last.weights	07-06-2022 16:35	WEIGHTS File	2,50,016 KB

Figure 5.5 Weight files created during the training of YOLOv4 model.

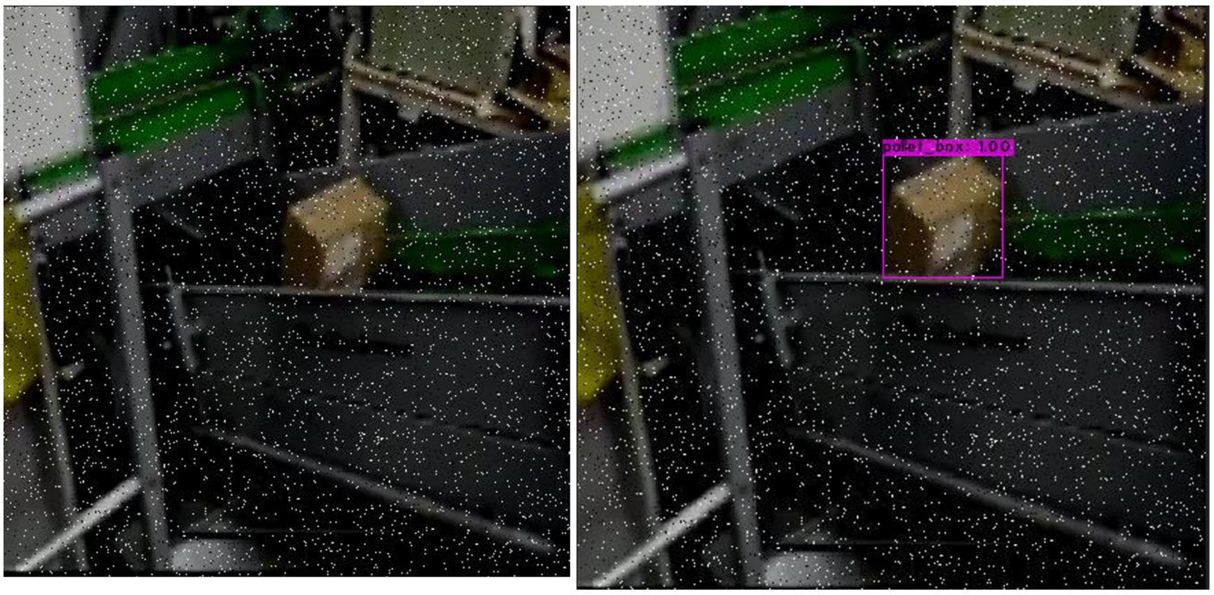


Figure 5.6 Object detection by YOLOv4 model on a random test image.

The variation of mAP and loss, when trained the model up to 2000 batches is shown in the figure 5.7. The mAP score achieved is 96% at 0.5 IOU threshold with 95.96% as an average precision.

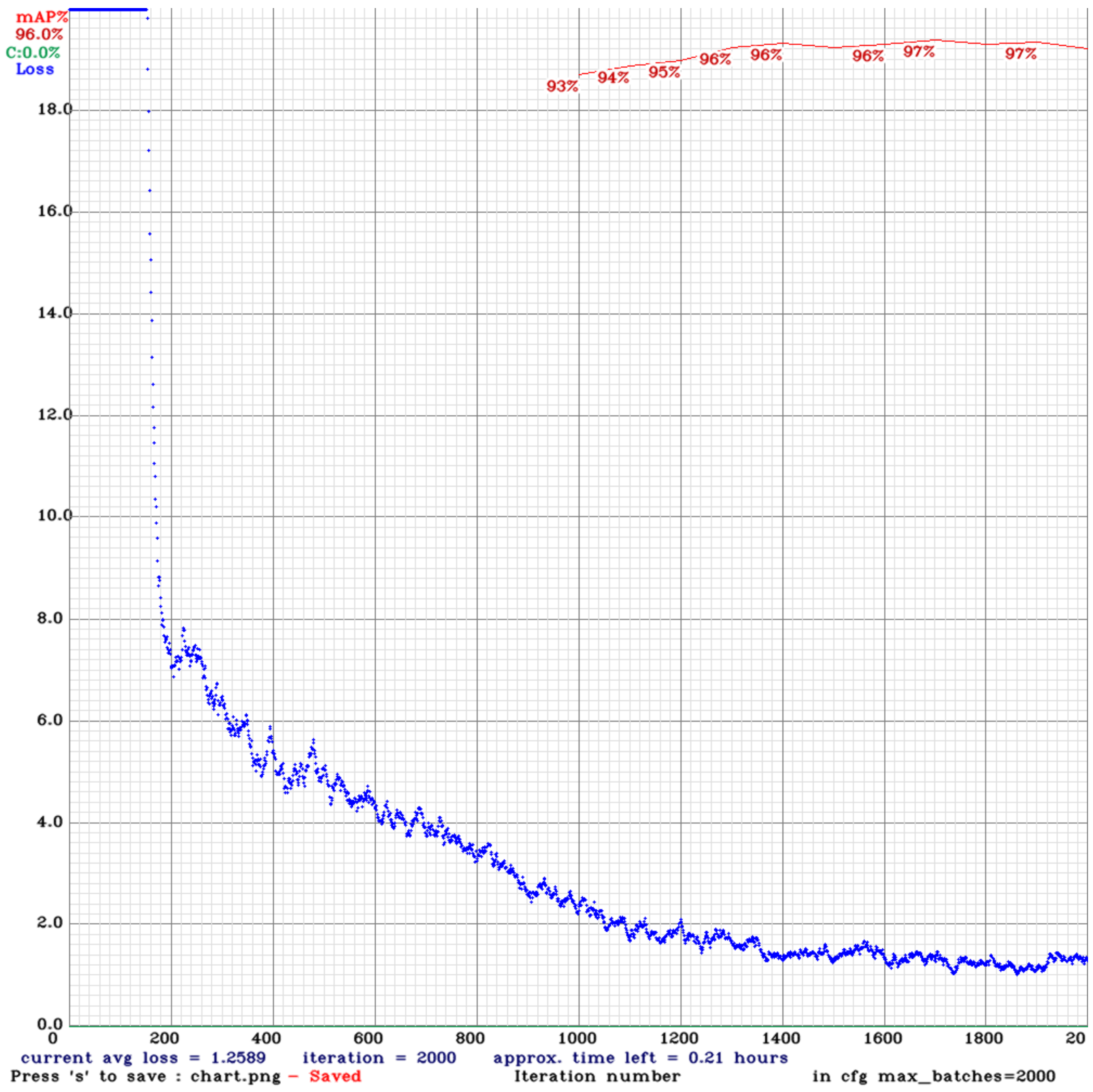


Figure 5.7 Performance of YOLOv4 model.

To Summarize, the results of YOLO models,

Table 5.1 Comparison of the mAPs of the object detection models

Model	**mAP at 0.5 IOU**
YOLOv5x	99.5%
YOLOv4	96.0%

As per the mAP, YOLOv5x model is performing better than the YOLOv4 model. But when compare the detections performed by these two models on a random image, i.e., comparing the figures 5.3 and 5.6, YOLOv4 model made detection with 100% confidence score. There will be a variety of results irrespective of mAP when these models are trained on the unseen or test images. So, we conclude that, both these models are performing equally.

5.2 Results of the custom object tracker:

Even though our custom object tracker tracked the objects correctly in few frames of the given pallet video, identity switches happened for some objects, false positives and false negatives also occurred. To prevent the occurrence of identity switches, experimented with the various values of the threshold distance and when it was at 50, we observed very little identity switches but still the occurrence of false positives and false negatives are inevitable. To measure how accurate our object tracker performed on the given video, which is having a duration of 12 seconds and contains 5 pallet boxes overall, we use a metric called "Multiple Object Tracking Accuracy", in short MOTA. It can be calculated as shown in the figure 5.8, where:

- t is the time frame
- FN is False Negative
- FP is False Positive
- IDS is the number of identity switches
- GT is the Ground truth object count.

We also performed object tracking using DeepSORT algorithm with our trained YOLOv5x model on the same video, to compare the MOTA scores of these two versions.

$$MOTA = 1 - \frac{\sum_t FN_t + FP_t + IDS_t}{\sum_t GT_t}$$

Figure 5.8 Equation to calculate the MOTA. Source - https://visailabs.com/evaluating-multiple-object-tracking-accuracy-and-performance-metrics-in-a-real-time-setting/

To do this we followed the below procedure:

1. Using opencv-python library, we divided the original video into 362 frames, which are labelled as a ground truth values.
2. We also divided the output videos obtained from the custom object tracker and the DeepSORT version respectively, into the same number of frames.
3. We noted down the number of False positives, False negatives, identity switches and the Ground truth objects with respect to each frame, then we calculated the MOTA scores for both the versions. The results are as shown in the table 5.2.

Table 5.2 MOTA scores of the object trackers

Object Tracker	**Total ID Switches**	**Total False positives**	**Total False Negatives**	**Total Ground Truth objects**	**MOTA Score**
Custom Object tracker	7	24	36	873	92%
Object tracker developed using Deep SORT	5	2	0	873	99%

Our custom object tracker performed well and achieved 92% MOTA score. But it was outperformed by the object tracker developed using the DeepSORT algorithm, which achieved 99% MOTA score. Ours recorded a greater number of False Negatives, i.e., it failed to detect the objects in some frames. DeepSORT algorithm, a robust and one of the most widely used object tracking algorithm, performed minimal ID switches, and recorded very less false positives. Our custom object tracker can be further improved by experimenting more with the threshold distance and implementing the Kalman filtering algorithm, or the Hungarian algorithm, which improves the associations of the object ids in the consecutive frames.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions:

The present study has discussed the importance of the efficient warehouse management system, which can be achieved by implementing the automation techniques. The study proposed a computer vision methodology, that detects, tracks, and counts the pallet boxes, aiming to maintain the accurate data of the number of pallet boxes entering and leaving the warehouse, to achieve an efficient warehouse management system, and ultimately contributing to maintain an efficient supply chain. A detailed literature review, which was done by following the funnel down approach, has discussed about the various developments in the areas of Supply chain, the need of an automating warehouse operations, and how the trending technologies in this Industry 4.0 revolution phase, like Internet of things and Artificial Intelligence, are adding value to accomplish the automation. This literature review laid a solid foundation to develop this computer vision methodology.

The proposed computer vision methodology was implemented in two phases:

1. Object detection.
2. Object tracking and counting.

In the Object detection phase, first, the pallet box image dataset was labelled with the ground truth bounding box coordinates in the YOLO format. Then trained two object detection models using the YOLOv5x and YOLOv4 algorithms respectively. The YOLOv5x model achieved a mAP of 99.5% while the YOLOv4 model achieved a mAP of 96%. We chose YOLOv5x model to deploy it in the object trackers. In the object tracking and counting phase, two object trackers were developed, one is a custom object tracker, and the other is developed by using DeepSORT algorithm. Both utilizes the YOLOv5x model, which was converted into ONNX weights and further converted to the DNN model by OpenCV dnn module, to make object detections on the frames. To track the objects in the consecutive frame, custom object tracker uses the Euclidean distance approach, which maintains the same tracking id assigned to an object, when the distance between the centre points of an object in two consecutive frames is less than the predefined threshold, i.e., 50 pixels, which was determined after performing the series of experiments. This custom object tracker also counts the number of pallet boxes occur in the frame and displays the count, which can be used for monitoring purposes. Both these trackers are tested on the same video. Custom object tracker achieved 92% MOTA score due to the greater number of false negatives, false positives, and identity switches, while the DeepSORT based tracker achieve 99% with smaller number of the false positves and identity switches. This custom object tracker can be further improved by implementing the Kalman filtering algorithm, or using the Hungarian algorithm, which improves the association of the objects in the consecutive frames, to achieve even better MOTA score.

The other state of the art object detection algorithms like YOLOR, SSD, and RetinaNet are also can be experimented with the pallet box detection use case, and the comparative study can be performed. But due to the limited time scope, this study has confined for training the custom object detection models only with the YOLOv5x and the YOLOv4 algorithms respectively. The study concludes that there is a lot of scope of research in the field of warehouse management, and when it combines with the AI research, it may contribute a lot in achieving an efficient supply chain, which benefits not only the individual organizations or companies taking part in that chain, but the whole global community as well.

6.2 Future Recommendations:

This study has combined the two different domains, one is Warehouse management system, and the other is Computer vision, and developed a methodology that accurately tracks and counts the number of pallet boxes, which are entering and exiting the warehouse. It has a greater scope in the research because, it has a vast potential to achieve the efficient warehouse management system. This methodology can be customized to use on other storage containers like cable drums and barrels. Artificial Intelligence has been evolving with respect to the time in such a way that, it can solve any real time problem. With the availability of the high-performance faster GPUs, the availability of the high-speed internet, and the cloud technologies, it is recommended to develop a robust system by combining the Internet of Things and the Artificial Intelligence techniques such as Computer vision, Deep learning, and reinforcement learning, which can solve the pallet tracking and counting problems with a better accuracy. As this enables the machines to transfer the information among themselves and work accordingly, this would reduce the human intervention to a great extent thereby reducing a lot of human errors, and the data that is generated can be stored in cloud and can be utilized to draw the business-driven insights. This study can be further enhanced by the future researchers and scholars, in supply chain management domain, the crucial component of any business in today’s world.

References:

Anon (2017) *APICS Supply Chain Operations Reference Model SCOR Version 12.0*.

Bewley, A., Ge, Z., Ott, L., Ramos, F. and Upcroft, B., (2016) Simple Online and Realtime Tracking. [online] Available at: http://arxiv.org/abs/1602.00763.

Bochkovskiy, A., Wang, C.-Y. and Liao, H.-Y.M., (2020) YOLOv4: Optimal Speed and Accuracy of Object Detection. [online] Available at: http://arxiv.org/abs/2004.10934.

Boysen, N., de Koster, R. and Weidinger, F., (2019) Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*, 2772, pp.396–411.

Chen, J., Koju, W., Xu, S. and Liu, Z., (2021) Sales Forecasting Using Deep Neural Network and SHAP techniques. In: *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering, ICBAIE 2021*. Institute of Electrical and Electronics Engineers Inc., pp.135–138.

Christoph Jan Bartodziej, (2017) *The Concept Industry 4.0*. [online] Available at: https://link.springer.com/book/10.1007/978-3-658-16502-4 [Accessed 8 Feb. 2022].

Cormier, G. and Gunn, E.A., (1992) *A review of warehouse models*. *European Journal of Operational Research*, .

Dewa, P.K., Pujawan, I.N. and Vanany, I., (2017) Human errors in warehouse operations: An improvement model. *International Journal of Logistics Systems and Management*, 273, pp.298–317.

Ding, Y., Jin, M., Li, S. and Feng, D., (2021) Smart logistics based on the internet of things technology: an overview. *International Journal of Logistics Research and Applications*, 244, pp.323–345.

Feng, X., Jiang, Y., Yang, X., Du, M. and Li, X., (2019) *Computer vision algorithms and hardware implementations: A survey*. *Integration*, .

Ghaouta, A., El, A. and Okar, C., (2018) *Big Data Analytics Adoption in Warehouse Management: A Systematic Review; Big Data Analytics Adoption in Warehouse Management: A Systematic Review*. *2018 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD)*.

Girshick, R., (n.d.) Fast R-CNN. [online] Available at: https://github.com/rbgirshick/ [Accessed 14 Feb. 2022].

Girshick, R., Donahue, J., Darrell, T. and Malik, J., (n.d.) Rich feature hierarchies for accurate object detection and semantic segmentation Tech report (v5). [online] Available at: http://www.cs.berkeley.edu/˜rbg/rcnn. [Accessed 14 Feb. 2022].

Górtowski, S., (2019) Supply chain modelling using data science. In: *Lecture Notes in Business Information Processing*. Springer Verlag, pp.634–645.

Gray, J., Hobbs, J., Gregory, S. and Singh, U., (2021) A Computer Vision Pipeline for Automatic Large-scale Inventory Tracking. [online] 82021. Available at: https://doi.org/10.1145/3409334.3452063 [Accessed 27 Jan. 2022].

Han, S., Liu, X., Han, X., Wang, G. and Wu, S., (2020) Visual Sorting of Express Parcels Based on Multi-Task Deep Learning. *Sensors 2020, Vol. 20, Page 6785*, [online] 2023, p.6785. Available at: https://www.mdpi.com/1424-8220/20/23/6785/htm [Accessed 23 Jan. 2022].

He, K., Gkioxari, G., Dollár, P. and Girshick, R., (2017) Mask R-CNN. [online] Available at: http://arxiv.org/abs/1703.06870.

Hoque, M.E., Thavaneswaran, A., Appadoo, S.S., Thulasiram, R.K. and Banitalebi, B., (2021) A novel dynamic demand forecasting model for resilient supply chains using machine learning. In: *Proceedings - 2021 IEEE 45th Annual Computers, Software, and Applications Conference, COMPSAC 2021*. Institute of Electrical and Electronics Engineers Inc., pp.218–227.

Huan, S.H., Sheoran, S.K. and Wan, G., (2004) *A review and analysis of supply chain operations reference (SCOR) model. Supply Chain Management*, .

Janat Shah, (2016) *Supply Chain Management: Text and Cases, 2nd Edition*. [online] Available at: https://www.oreilly.com/library/view/supply-chain-management/9789353062538/ [Accessed 5 Feb. 2022].

Kalinov, I., Petrovsky, A., Ilin, V., Pristanskiy, E., Kurenkov, M., Ramzhaev, V., Idrisov, I. and Tsetserukou, D., (2020) WareVision: CNN Barcode Detection-Based UAV Trajectory Optimization for Autonomous Warehouse Stocktaking. *IEEE Robotics and Automation Letters*, 54, pp.6647–6653.

Kumar, S., Narkhede, B.E. and Jain, K., (2021a) *Revisiting the warehouse research through an evolutionary lens: a review from 1990 to 2019. International Journal of Production Research*, .

Kumar, S., Vishal, Sharma, P. and Pal, N., (2021b) Object tracking and counting in a zone using YOLOv4, DeepSORT and TensorFlow. In: *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*. Institute of Electrical and Electronics Engineers Inc., pp.1017–1022.

Lee, C.K.M., Lv, Y., Ng, K.K.H., Ho, W. and Choy, K.L., (2018) Design and application of internet of things-based warehouse management system for smart logistics. *International Journal of Production Research*, 568, pp.2753–2768.

Li, L., Su, Q. and Chen, X., (2011) Ensuring supply chain quality performance through applying the SCOR model. *International Journal of Production Research*, 491, pp.33–57.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A.C., (2015) SSD: Single Shot MultiBox Detector. [online] Available at: http://arxiv.org/abs/1512.02325.

Lototsky, V., Sabitov, R., Smirnova, G., Sirazetdinov, B., Elizarova, N. and Sabitov, S., (2019) Model of the Automated Warehouse Management and Forecasting System in the Conditions of Transition to Industry 4.0. *IFAC-PapersOnLine*, 5213, pp.78–82.

Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W. and Kim, T.K., (2021) *Multiple object tracking: A literature review*. *Artificial Intelligence*, .

Mabert, V.A. and Venkataramanan, M.A., (n.d.) *Special Research Focus on Supply Chain Linkages: Challenges for Design and Management in the 21st Century* Challenges for Design and Management in the 21st Century*. *Decision Sciences*, .

Mohamed, I.S., Capitanelli, A., Mastrogiovanni, F., Rovetta, S. and Zaccaria, R., (2020) Detection, localisation and tracking of pallets using machine learning techniques and 2D range data. *Neural Computing and Applications*, 3213, pp.8811–8828.

O'Shea, K. and Nash, R., (2015) An Introduction to Convolutional Neural Networks. [online] Available at: http://arxiv.org/abs/1511.08458.

Padilla, R. and P.W.L. and D.T.L.B. and N.S.L. and da S.E.A.B., (2021) A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. *Electronics*, [online] 10, pp.237–242. Available at: https://www.mdpi.com/2079-9292/10/3/279 [Accessed 15 Feb. 2022].

Patel, A.D. and Chowdhury, A.R., (2020) Vision-based object classification using deep learning for inventory tracking in automated warehouse environment. In: *International Conference on Control, Automation and Systems*. IEEE Computer Society, pp.145–150.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., (2015) You Only Look Once: Unified, Real-Time Object Detection. [online] Available at: http://arxiv.org/abs/1506.02640.

Redmon, J. and Farhadi, A., (2016) YOLO9000: Better, Faster, Stronger. [online] Available at: http://arxiv.org/abs/1612.08242.

Redmon, J. and Farhadi, A., (2018) YOLOv3: An Incremental Improvement. [online] Available at: http://arxiv.org/abs/1804.02767.

Ren, S., He, K., Girshick, R. and Sun, J., (n.d.) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. [online] Available at: http://image-net.org/challenges/LSVRC/2015/results [Accessed 27 Jan. 2022].

Taghizadeh student, E., (n.d.) *UTILIZING ARTIFICIAL NEURAL NETWORKS TO PREDICT DEMAND FOR WEATHER-SENSITIVE PRODUCTS AT RETAIL STORES*.

Tirkolaee, E.B., Sadeghi, S., Mooseloo, F.M., Vandchali, H.R. and Aeini, S., (2021) *Application of Machine Learning in Supply Chain Management: A Comprehensive Overview of the Main Areas*. *Mathematical Problems in Engineering*, .

Tjahjono, B., Esplugues, C., Ares, E. and Pelaez, G., (2017) What does Industry 4.0 mean to Supply Chain? *Procedia Manufacturing*, 13, pp.1175–1182.

Toorajipour, R., Sohrabpour, V., Nazarpour, A., Oghazi, P. and Fischl, M., (2021) Artificial intelligence in supply chain management: A systematic literature review. *Journal of Business Research*, 122, pp.502–517.

Wiley, V. and Lucas, T., (2018) Computer Vision and Image Processing: A Paper Review. *International Journal of Artificial Intelligence Research*, 21, p.22.

Wojke, N., Bewley, A. and Paulus, D., (2017) Simple Online and Realtime Tracking with a Deep Association Metric. [online] Available at: http://arxiv.org/abs/1703.07402.

Yang, J.X., Li, L.D. and Rasul, M.G., (2021) Warehouse Management Models Using Artificial Intelligence Technology with Application at Receiving Stage – A Review. *International Journal of Machine Learning and Computing*, [online] 113, pp.242–249. Available at: http://www.ijmlc.org/index.php?m=content&c=index&a=show&catid=114&id=1224.

Zhao, K., Zhu, M., Xiao, B., Yang, X., Gong, C. and Wu, J., (2020) Joint RFID and UWB Technologies in Intelligent Warehousing Management System. *IEEE Internet of Things Journal*, 712, pp.11640–11655.

APPENDIX A: RESEARCH PROPOSAL:

PALLET DETECTION, TRACKING AND COUNTING USING COMPUTER VISION TECHNIQUES TO ACHIEVE THE WAREHOUSE AUTOMATION

SESHA VENKATA SRIRAM ERRAMILLI

RESEARCH PROPOSAL

Liverpool John Moores University – MSc in Data Science

OCTOBER 2021

Abstract

The popularity of data science is showing an increasing trend, as all kinds of industry across the globe are adopting it for various use cases. Industries like healthcare, are utilizing the state-of-the-art deep learning and computer vision techniques like Convolution neural networks to detect the cancer, brain tumours and most recently in detecting the presence of COVID 19. The success of Data science methodologies in the health care industries, like predicting the diseases in advance and saving the lives, has become possible because of the vast availability of the data both in the structured and the unstructured form, and properly utilizing them to make analyses. Similarly, in the Supply chain management domain, large amounts of data are available. Supply chain managers strive hard in maintaining the stable and efficient supply chains so that their respective organizations can achieve and maintain the customer satisfaction and can earn more profits. The key component of the entire supply chain management is the warehouse management. Because the role of warehouses comes in the two areas, one is storing the raw materials that are purchased from the suppliers and the other area is storing the manufactured goods or the finished goods that are to be delivered to the customers. In both the areas of warehousing, labour plays a major role in picking the items, staging, putting them into storage, packing, labelling, scanning, and shipping. During the scanning process, the chances of getting errors like improper scanning of the barcodes or RFID tags are high. To solve this problem, in this research, we will train customized object detection models using Yolov4 and Yolov5 separately, based on our pallet dataset and we will also propose a object tracking methodology based on the DeepSORT algorithm, using which we can accurately detect, track and count the pallets in a video. This enables the proper tracking of items both incoming and outgoing, thus achieving the efficient warehouse management.

Table of Contents

LIST OF FIGURES

1. Background

Supply chain management plays a crucial role in every business. Aim of the supply chain management is to provide a product or service to a customer at the right place, right time, right quality and at the right price, and in this process make profits. Supply chain management deals with the material flow from the suppliers to the manufacturer, from the manufacturer to the wholesaler, from the wholesaler to the retailers, and finally from the retailers to the end customers. While the money flow follows the opposite direction quite often, i.e., from end customers to the suppliers and the information flow is bidirectional. So, in a nutshell, we can say that the supply chain management integrates each part in the entire product life cycle.

Due to the rapid increasing of the volatile demand and the products which have high customizability, German government launched a strategic initiative called the Industry 4.0 (Christoph Jan Bartodziej, 2017) in 2011. This initiative aims at integrating the cyber and physical systems to achieve more efficient supply chain management. Now a days, due to this Industry 4.0 revolution, Digital supply chain management systems are becoming popular which enables to integrate virtually all the components of the traditional supply chain. Some of the examples are, Enterprise resource management or ERP tool, Customer relationship management or CRM and Supplier relationship management or SRM.

Most companies are implementing their supply chain management strategies to achieve and maintain the satisfaction of their customers. But due to the complexity nature of the business processes in every organization, a large amount of data is being generated (Górtowski, 2019) at each department of the supply chain – say in supply, demand, sales, operations etc., Unable to process this huge amount of data, the traditional systems are becoming inefficient (Tirkolaee et al., 2021).

Due to the increasing popularity of Data science across all the domains, researchers started exploring its application in the Supply chain management as well and as a result, companies also started utilizing Data science methodologies. The well-known use case of Data science application in the supply chain management is demand and sales forecasting. By using the nonlinear machine learning techniques (Tirkolaee et al., 2021), the sales team can accurately predict and forecast the inventory, sales and demand which leads to the increase in profits. The research (Taghizadeh student, n.d.) proposed the application of multi-layer perceptron (Artificial neural networks) for solving the demand and sales forecasting problem of a retail store. Likewise, there are many use cases in the sales and demand forecasting.

Apart from the Sales and demand forecasting, there are other use cases of the supply chain like, supplier identification and evaluation, warehouse management, risk management, production planning, generating the optimum delivery routes, sentiment analysis of the customers, etc. Among them, warehouse management is very crucial use case, which involves a tremendous effort of manpower and hence there's a high possibility of occurrence of human errors (Dewa et al., 2017) in tracking the items that enter or leave the warehouse.

There is a lot of research going on in the warehouse management, one among them, (Kalinov et al., 2020) proposed an Unmanned aerial vehicle-based robot that moves in the warehouse and by utilizing the encoder decoder based convolutional neural networks architecture, detects, and scans the bar code, and thus reduces the time of stocking and the number of mistakes in scanning the barcode decreases.

One more research, (Gray et al., 2021) proposed a low-cost automated inventory tracking model for an automotive manufacturing company. The proposed model utilizes the object detection and tracking models (Using the SSDMobileNetV2 and the YOLOv4 tiny as separate object detection models) to identify the storage rack containing the vehicle parts and reads the barcode information using the Optical character recognition techniques. By augmenting this model to a high-definition camera, that is attached to a mechanically autonomous imaging module which is suspended aerially travels in the aisles of the inventory warehouse, reads the data about the inventory, and stores it in a data base thus enabling the access to the real time data.

The utilization of computer vision techniques is also increasing in many domains and warehouse management is not an exception as seen from the above two use cases. For example, YOLO series has many numbers of object detection models, that were trained on the standard data sets like MS COCO and Image net. These models can be customized as per our use case and can yield accurate results. Starting from the facial recognition attendance marking system to the latest driverless cars, computer vision methods are being applied.

In this research, we are also going to apply the computer vision techniques - object detection and tracking, to solve the item tracking problems in warehouses, by using our pallet images dataset.

2. Problem Statement

2.1 Challenges in the Warehouse management

Warehousing plays an important part in the entire Supply chain management of an organization, both in terms of

a. Storing the raw material components which are received from the suppliers. (Inbound warehousing)
b. Storing the finished goods which will be delivered to the customers in the form of trade units like pallets. (Outbound warehousing).
c. Contract warehousing, which is handled by a third-party vendor and can be applicable for both Inbound and Outbound warehousing types.

But there are some challenges in the warehouse operations. Scenarios like not getting the right data at the right time, occurrence of human error (as the manpower is a crucial part in the warehouse operations (Dewa et al., 2017)) are one of the challenges. If these challenges are not dealt properly, there might be a risk of losing the customer satisfaction, which is the top priority for any business.

Due to the Industry 4.0 revolution (Christoph Jan Bartodziej, 2017), there is a huge availability of solutions to tackle these challenges, to achieve the efficient automated warehouse management. The technologies such as sensors, computer vision, augmented reality (AR), drones, Internet of things (IoT) and robotics provide many opportunities to automate and optimize the warehousing, working along with the manpower. This also enables us to predict the appearance of bottlenecks in advance, thus ensuring the effective warehouse management (Lototsky et al., 2019).

Some of the global organizations have already started deploying the technologies like Internet of things (IOT) and Big Data analytics in their Warehouse management systems to achieve efficient operations. And the research is going on.

An IOT based Warehouse management system (Lee et al., 2018), which completely utilizes the Radio Frequency Identification or RFID technology, and the wireless sensors to track and trace the raw materials and finished goods. This enables the real time information sharing and ensures the availability of the right data at the right time, which in turn enables to be in sync with the changing customer requirements.

However, there are some challenges with the above mentioned IOT based methodology. As the Barcodes and the RFID tags are the typical ways to identify the items in a warehouse, there’s a possibility of occurrence of damage to barcodes and the RFID tags have a limited lifetime, which involve huge costs in assigning new barcodes/RFID tags and thus affects the inventory tracking.

2.2 Computer vision and Deep learning-based solution

One of the solutions to this problem, is to use Computer vision and Deep learning-based approach. As proposed by (Patel and Chowdhury, 2020), items entering a warehouse can be classified using convolutional neural networks with greater accuracy and helps in sorting the items according to their respective classes.

In addition to the above CNN based approach, Object Detection and tracking also becoming one of the popular approaches to carry out the operations like object identification/classification, detection, and tracking. The necessity of combining the classical neural network models and the computer vision methods, as proposed by the (Yang et al., 2021), gives an accurate result. The proposed methodology in the research (Mohamed et al., 2020) shows the Faster R-CNN cascaded with the CNN based classifier can be used for object detections while the Kalman filters are used for tracking the objects.

In this research, we will try to solve the problem of detecting, counting, and tracking the pallet boxes using the object detection and tracking algorithms. We use YOLO versions for object detection, compare their performances and choose the best one among them. Then we develop and use our customized object tracking algorithm for counting and tracking the objects, with the help of Tensorflow library in python. The solution that we propose will be implemented using the python programming language and aims to achieve an efficient warehousing and reduces the human effort.

2.3 Why YOLO for object detection?

There are many popular object detection algorithms that are widely used. Algorithms like R-CNN (Girshick et al., n.d.), generates regional proposals for each image and those are passed to convolutional network for extracting feature maps. The extracted feature maps are passed to the fully connected layers, from which the output will be branched into two parts – one outputs the class of the object detected and the other one outputs the values of the bounding box by applying regression. The generation of region proposals in the R-CNN and training them is a more time-consuming task.

But YOLO (Redmon et al., 2015) is a different case. Unlike the algorithms like R-CNN where the object detection is done by the classifiers and the bounding boxes by the regressors, YOLO framed the entire object detection problem into a single regression problem, i.e., bounding boxes and their respective class probabilities are predicted in the form of continuous variables. To do so, it utilizes a single neural network architecture, which makes predictions directly from full images in a single evaluation. So, the name YOLO is You only look once.

There are many versions of YOLO since its initial release. Here's the brief description for each one of them:

- YOLOv2 (Redmon and Farhadi, 2016) is developed by making a lot of improvements on top of the initial YOLO and anchor boxes are one among them. It can detect over 9000 object categories.
- YOLOv3 (Redmon and Farhadi, 2018) is a faster one compared to its previous version. Built upon previous versions with the additional features like predicting the objectiveness score for each bounding box, it improved the performance even on the smaller objects.
- YOLOv4 (Bochkovskiy et al., 2020) is still being used by many to perform object detections as it makes real time detection as priority and conducts training on a single GPU. It introduced other improvements like improved feature aggregation, mish activation and more.
- YOLOv5 by Glenn Jocher released shortly after the release of YOLOv4 in the same year 2020. It was developed in Pytorch framework and many versions of YOLOv5 are being developed, examples are YOLOv5n, YOLOv5s, YOLOvm, YOLOv5l and YOLOv5x, which work on the 640x640 size of pixels. YOLOv5x is the fastest among all the above-mentioned versions.

Despite YOLO makes localization errors, it is faster and makes accurate predictions. There are other single shot detectors like SSD (Liu et al., 2015) and we can perform the comparative study But due to the time limitation and considering the available computational power, we are going to use only YOLO for object detection.

So, in our research, we are going to make two models one with custom YOLOv4 in Tensorflow and other with the YOLOv5x in Pytorch , then choosing the best one among them by using the evaluation metrics.

3. Aim and objectives

Artificial intelligence and data science are being applied in many domains and Supply chain management is not an exception. Due to Industry 4.0 revolution, there are a lot of use cases in this domain, which can be automated to achieve the efficient and low error operations. One of them is the warehouse management, in which the traditional approaches like manual counting of the incoming and outgoing items. Despite having unique RFID tags or barcodes on each item, manpower is utilized to scan the items by holding the scanning machines. And there are certain challenges like, damage of barcode labels, RFID tags and human errors can lead to inefficient warehousing. To solve this problem, we will be using the computer vision techniques like object detection, tracking, and counting to detect pallets in the warehouse and the dataset we will be using is provided by iNeuron.ai, as I'm currently pursuing an open internship with them on the similar project called "Pallet loading, counting and tracking".

Following objectives are formulated based on our study:

a. Data preparation for analysis

b. Experiment with Yolov4 and Yolov5 (implement in Pytorch) object detection algorithms on the dataset, compare their results based on IOU, precision, recall and average precision metrics and choose the best one.

c. Then we will be developing a custom object tracker by taking DeepSORT algorithm as reference, which uses the best detector to perform object detection, and then performs multiple objects tracking and counting. The reason for proposing a customized tracking algorithm is to deal with the occlusion problem and to improve the MOTA (multiple object tracking accuracy) score.

4. Significance of the study

This research enables us to develop a tracking algorithm, which can couple with the detector to detect, track, and count the items in the warehouse and stores the data into the database. We will be making open-source contribution of our code so that, anyone can use it, can make their own versions of it.

Cameras can be augmented with this real time object detection and tracking system, so that whatever goods enter or leave the warehouse, they can capture it effectively. This reduces the human errors and increases the efficiency, so that the supply chain management can have the right data at the right time to take the right decisions and ultimately leads to profits.

5. Scope of the study

Due to the time limitations and with the available computational power, the scope of our study will be limited to:

- Using only Yolov4 and Yolov5 models for object detection tasks and comparing the two models using the performance evaluation metrics.
- Our focus will be on developing a customized object tracker.
- Using only the dataset provided by iNeuron.ai for training and testing our models.

6. Research Methodology

This part of our research proposal consists of the complete methodology of how we will be doing our research.

6.1 Introduction:

This research starts with exploring the data set and making annotations on the training images. Then we customize the object detection algorithms Yolov4 and Yolov5 separately and train the two models. Once the models are trained, they are evaluated on the test images and based on the performance metrics – Intersection over union (IOU), Precision, Recall and Average Precision, we will compare the two models and choose the best one.

The weights of the best object models are then saved. Next comes the object tracking part, where we will develop a customized object tracker based on the DeepSORT algorithm. The performance of the tracking algorithm is measured by using the multiple object tracking accuracy (MOTA) score. By using the best detector model weights and this newly developed object tracker algorithm, we can perform the object detection, object tracking and counting operations.

6.2 About the Dataset:

This dataset is provided by iNeuron.ai and it is made open source by them as a part of their open internship program, where anyone can sign up for free and start working on the projects. This dataset consists of different images of pallets, all belonging to single pallet class and there are two train data folders. We will be choosing one out of them which is having a greater number of training images. Labelling part has already been done by someone, but for our research we will manually do the annotation for each image so that we may get some idea.

Along with the training images, we were also given videos, on which we can implement the tracker algorithm, which will track and count the objects.

6.3 Annotating the training images:

To train our detection models, we need to let them know what the objects in each image are. Just like for a regular tabular data, after dividing it column wise into the independent or feature variables and the dependent or target variable, for predicting the latter one based on the learning the relationship with the former one, here we need to annotate the objects in each image, so that the model can learn about those objects and can make predictions on the unseen images.

To do this, we draw a bounding box around each object that we want the detector to see, and we label each box with the object class, that we would like our detector to predict. "LabelImg" is one of the free labelling tools that we will be using for accomplishing the annotation task. For yolov5, we will convert the annotations into the COCO format.

6.4 Custom object detection models training:

Since we are having our pallet dataset and it is of single class type i.e., all the pallets come under the single pallet class, we will be customizing our Yolov4 and Yolov5 models accordingly. Here's the brief explanation about how our implementation is going to be. This procedure is just a proposal and may vary slightly at the time of implementation, depending on the software dependencies.

- **Yolov4 model:**
    a. We will implement it using Tensorflow library in python.
    b. First, we will be downloading the Yolov4 pre trained weights from the Darknet Github repository (https://github.com/AlexeyAB/darknet) to our local system or google drive in case If we use google colab notebook. Then we will install the dependencies as the requirements.
    c. Next, we need to create a folder or a directory with the name as 'yolov4-custom' for easy identification and inside that create another directory 'weights' where we can save our trained weights.
    d. Download the 'yolov4-custom.cfg' configuration file and make the changes to it as per our use case. Similarly, we create our data.names and data.labels files with the details like number of classes and labels respectively.
    e. After customizing the configurations, we will start training our model on our labelled dataset. The trained weights get stored in a separate folder called "weights".

- **Yolov5 model:**
  a. Since Yolov5 is written in pytorch framework, we will do model training using the pytorch library in python. At the end, weights will be stored.
  b. We will be Downloading or cloning the Yolov5 Github repository (https://github.com/ultralytics/yolov5) and install the dependencies.
  c. Then we will install Torch GPU version capable to our Cuda version.
  d. Next, we will create YAML file for training. This file contains the file paths of our custom dataset i.e., the training images, annotations for training images (done in COCO format) and testing images respectively.
  e. Then run the train.py file which contains the training code by customizing the batch size based on the capacity of the memory we have. Here we will be using Yolov5x weights as it can be trained faster, as mentioned in the repository compared to other versions like v5n, v5s, v5m and v5l.

6.5 Evaluation of the object detection models:

Our use case is single class object detection problem. So, to check the performance metrics, we need to measure by comparing the predicted bounding boxes with the ground truth box, by classifying whether our model correctly detected the object (True Positive), not detected the object (False Negative) and incorrectly detected the object (False positive).

True Negative is not used in object detection like in classification problems, because there can be an infinite no. of possibilities of why a particular object is not available in a frame, which adds no benefit.

Briefly about the most widely used metrics for the object detection (Padilla, 2021):

- **Intersection over union (IOU):**
  This metric evaluates the overlap between the predicted bounding box of an object and its ground truth bounding box. Based on the threshold (generally set to 50%, 75% or 95 %, depending on the use case), we can conclude whether the detection is valid one (True positive) or invalid one (False positive).
- **Precision:**
  It is the ratio of correct positive predictions to the total no. of predictions, i.e., the ratio of True positives and (True positives + False positives).
- **Recall:**
  It is the ratio of correct positive predictions to the total no. of ground truth values, i.e., the ratio of True positives and (True positives + False negatives).
- **Average Precision (AP):**
  AP is obtained by averaging precision across all recall values between 0 and 1. This metric can be used to compare different detectors by plotting their precision x recall curves in a single plot and performing interpolation, so that we can easily determine the area under curve.
  There are two interpolation methods, one is 11 – point interpolation, in which we try to summarize the shape of a precision x recall curve, by averaging the precision across 11equally spaced recall levels.
  The other interpolation method is averaging precision across all the recall levels.
  Then we divide the AUC into small rectangular parts, calculating all the areas and will sum up to get the total area under curve to get the average precision value.

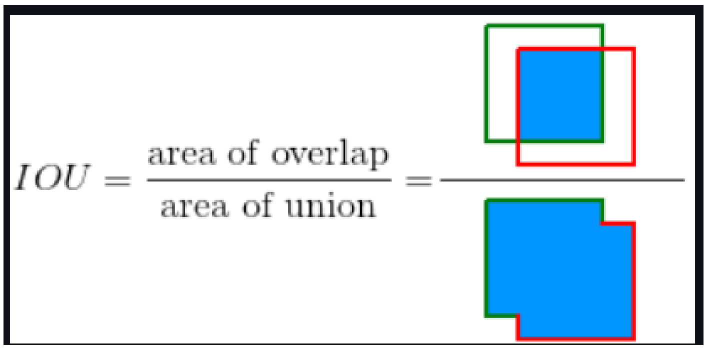

$IOU = \frac{\text{area of overlap}}{\text{area of union}} =$

Figure 6.1 Intersection over union explanation (Padilla, 2021)

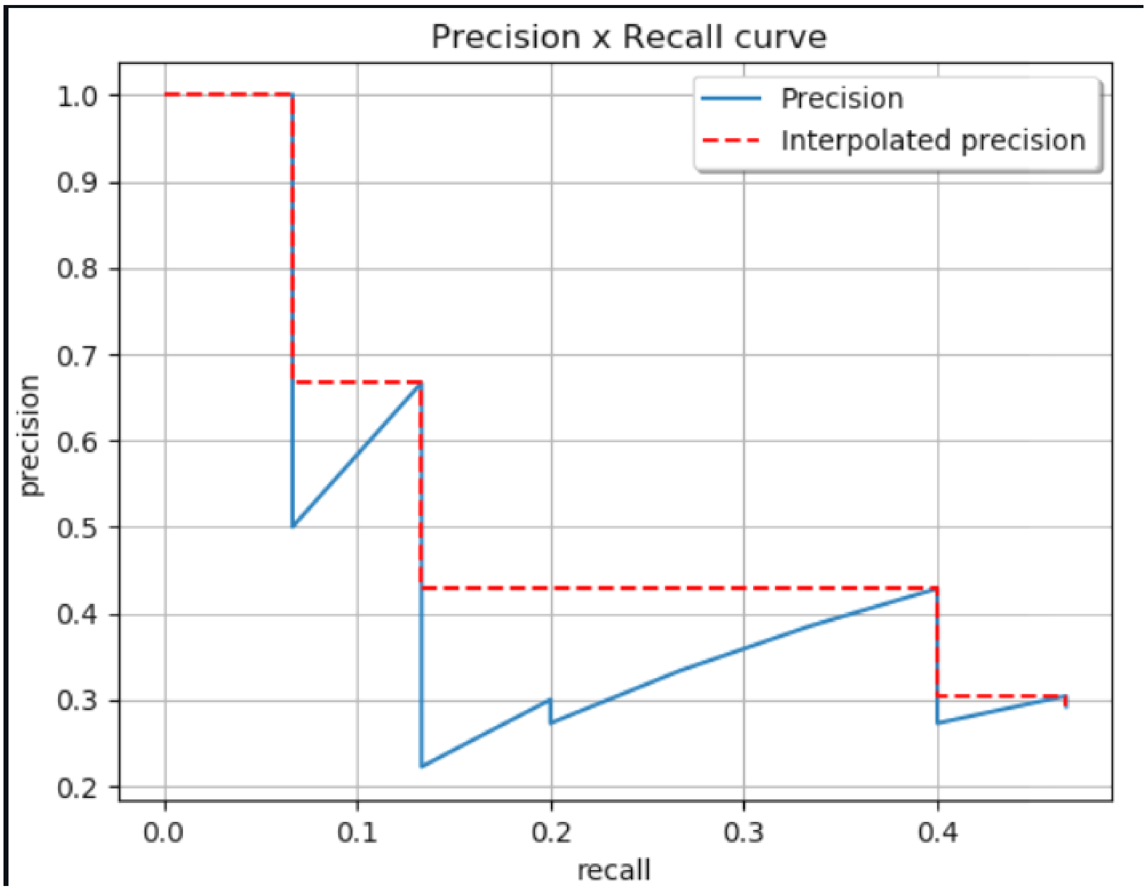


Figure 6.2 Calculating average precision using Precision x Recall curve (Padilla, 2021)

As shown in the figure 6.2, We evaluate our two models using the above-mentioned metrics and choose the best one, which can be used for detection and tracking purposes.

6.6 Custom object tracking methodology:

In this study, we propose an object tracking methodology, which can be used in tracking and counting the pallets in the videos by assigning unique ID to each object. For an easy explanation, considered only one object, but this methodology can be applied on multiple objects as well.

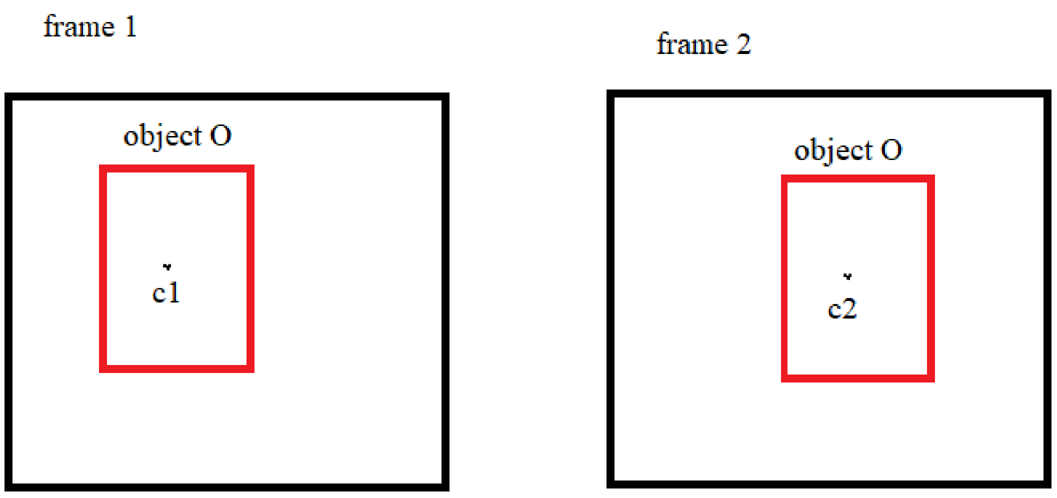


Figure 6.3 Illustration of the object tracking methodology

The methodology is as follows:

a. In each image, make sure the coordinates of bounding boxes are in this format (x,y,l,h)
b. Where (x,y) indicates centre, l and h indicate the length and breadth of the bounding box respectively.
c. Then, consider two successive frames in a video, detect the object O, in each one of them.
d. Assign a tracking ID for that object O, say i1
e. Fetch the centre points of the objects in the two frames and store them in two separate variables say c1 and c2 as shown in the figure 6.3.
f. Calculate the Euclidean distance between c1 and c2 say, d.
g. Define a threshold distance t.
h. If d is less than t, then keep the tracking ID i1.
i. If d is greater than t, then assign a new ID say id2 to the object O in the latter frame. So, the object in the latter frame is treated as a separate object.
j. Then, store all the unique tracking IDs of all the objects in a python list and count them to get the number of items tracked.

This entire iteration continues until the detected object O disappears from the subsequent frames.

Here’s the Pseudo code,

```
c1 = (x1, y1)   # centre of object O in frame 1
# Assign tracking Id as id1 for object O in both the frames
f1 = id1
f2 = id1
c2 = (x2, y2)   # centre of object O in frame 2
while object O in frame 2:
    # Calculate the Euclidean distance between c1 and c2
    d = $\sqrt{((x1-x2)^2 – (y1-y2)^2)}$
    #Defining a threshold distance t
    t = 3 units   # here 3 is taken just as an example. We need to define based on the pixels.
    # Condition to check
    if d is less than t:
        f1 = f2
    else:
        #Assign new id to the f2
        F2 = id2
```

6.7 Implementation of our object tracking methodology and evaluation:

The above methodology proposed will be implemented by taking the DeepSORT algorithm as a reference. Our intention of proposing this kind of algorithm for tracking is to experiment with the occlusion problem and based on that we can improve its performance.

We will use the best detectors weights, and couple with our custom tracker to perform the object detection, tracking and counting operations.

As mentioned in the study by (Kumar et al., 2021b), we use the metric called multiple object tracking accuracy (MOTA) to evaluate our tracker’s performance and make necessary changes to improve its performance.

7. Required Resources

7.1 Software Requirements:

We will be using the following software for our project implementation:

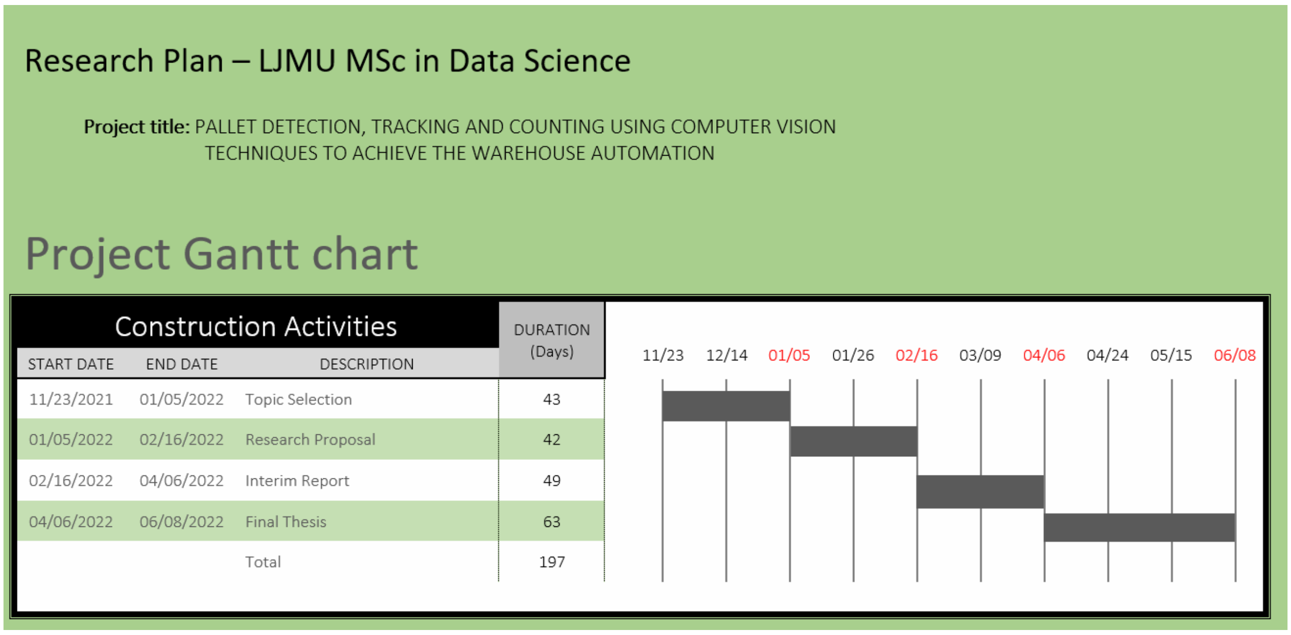
- **Operating System:** Windows 10 or 11
- **Python Package:** Conda version 4.11.0
- **Python version:** >=3.7
- **IDE:** Anaconda prompt, Jupyter notebook, PyCharm community edition
- **Python libraries:** Numpy, Tensorflow (GPU version 2.8.0), PyTorch>=1.7, Pandas
- **CUDA version:** 11.x
- MS Office, MS Paint, Mendeley Reference manager and Adobe Acrobat DC (PDF viewer)

7.2 Hardware Requirements:

Since our methodology follows Deep learning approach, we need more computational support for training the huge dataset that we are having. So, following are the hardware requirements:

- Intel Core i7 8 core CPU with 16GB RAM, NVIDIA GeForce RTX 3060 Laptop GPU (6GB GDDR6 dedicated).
- Google Colab.

8. Research plan



Research Plan – LJMU MSc in Data Science

Project title: PALLET DETECTION, TRACKING AND COUNTING USING COMPUTER VISION TECHNIQUES TO ACHIEVE THE WAREHOUSE AUTOMATION

Project Gantt chart

Construction Activities			DURATION (Days)
START DATE	END DATE	DESCRIPTION	
11/23/2021	01/05/2022	Topic Selection	43
01/05/2022	02/16/2022	Research Proposal	42
02/16/2022	04/06/2022	Interim Report	49
04/06/2022	06/08/2022	Final Thesis	63
		Total	197

References:

Dewa, P.K., Pujawan, I.N. and Vanany, I. (2017) ‘Human errors in warehouse operations: an improvement model’, Int. J. Logistics Systems and Management, Vol. 27, No. 3, pp.298–317.

Anon (2017) *APICS Supply Chain Operations Reference Model SCOR Version 12.0*.

Anon (n.d.) Effective Supply Chain Management.

Bewley, A., Ge, Z., Ott, L., Ramos, F. and Upcroft, B., (2016) Simple Online and Realtime Tracking. [online] Available at: http://arxiv.org/abs/1602.00763.

Bochkovskiy, A., Wang, C.-Y. and Liao, H.-Y.M., (2020) YOLOv4: Optimal Speed and Accuracy of Object Detection. [online] Available at: http://arxiv.org/abs/2004.10934.

Boysen, N., de Koster, R. and Weidinger, F., (2019) Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*, 2772, pp.396–411.

Chen, J., Koju, W., Xu, S. and Liu, Z., (2021) Sales Forecasting Using Deep Neural Network and SHAP techniques. In: *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering, ICBAIE 2021*. Institute of Electrical and Electronics Engineers Inc., pp.135–138.

Christoph Jan Bartodziej, (2017) *The Concept Industry 4.0*. [online] Available at: https://link.springer.com/book/10.1007/978-3-658-16502-4 [Accessed 8 Feb. 2022].

Cormier, G. and Gunn, E.A., (1992) *A review of warehouse models*. *European Journal of Operational Research*, .

Dewa, P.K., Pujawan, I.N. and Vanany, I., (2017) Human errors in warehouse operations: An improvement model. *International Journal of Logistics Systems and Management*, 273, pp.298–317.

Ding, Y., Jin, M., Li, S. and Feng, D., (2021) Smart logistics based on the internet of things technology: an overview. *International Journal of Logistics Research and Applications*, 244, pp.323–345.

Feng, X., Jiang, Y., Yang, X., Du, M. and Li, X., (2019) *Computer vision algorithms and hardware implementations: A survey*. *Integration*, .

Ghaouta, A., El, A. and Okar, C., (2018) *Big Data Analytics Adoption in Warehouse Management: A Systematic Review; Big Data Analytics Adoption in Warehouse Management: A Systematic Review*. *2018 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD)*.

Girshick, R., (n.d.) Fast R-CNN. [online] Available at: https://github.com/rbgirshick/ [Accessed 14 Feb. 2022].

Girshick, R., Donahue, J., Darrell, T. and Malik, J., (n.d.) Rich feature hierarchies for accurate object detection and semantic segmentation Tech report (v5). [online] Available at: http://www.cs.berkeley.edu/˜rbg/rcnn. [Accessed 14 Feb. 2022].

Glenn Jocher, (2020) *YOLOv5*. [online] Ultralytics. Available at: https://docs.ultralytics.com/ [Accessed 15 Feb. 2022].

Górtowski, S., (2019) Supply chain modelling using data science. In: *Lecture Notes in Business Information Processing*. Springer Verlag, pp.634–645.

Gray, J., Hobbs, J., Gregory, S. and Singh, U., (2021) A Computer Vision Pipeline for Automatic Large-scale Inventory Tracking. [online] 82021. Available at: https://doi.org/10.1145/3409334.3452063 [Accessed 27 Jan. 2022].

Han, S., Liu, X., Han, X., Wang, G. and Wu, S., (2020) Visual Sorting of Express Parcels Based on Multi-Task Deep Learning. *Sensors 2020, Vol. 20, Page 6785*, [online] 2023, p.6785. Available at: https://www.mdpi.com/1424-8220/20/23/6785/htm [Accessed 23 Jan. 2022].

He, K., Gkioxari, G., Dollár, P. and Girshick, R., (2017) Mask R-CNN. [online] Available at: http://arxiv.org/abs/1703.06870.

Hoque, M.E., Thavaneswaran, A., Appadoo, S.S., Thulasiram, R.K. and Banitalebi, B., (2021) A novel dynamic demand forecasting model for resilient supply chains using machine learning. In: *Proceedings - 2021 IEEE 45th Annual Computers, Software, and Applications Conference, COMPSAC 2021*. Institute of Electrical and Electronics Engineers Inc., pp.218–227.

Huan, S.H., Sheoran, S.K. and Wan, G., (2004) *A review and analysis of supply chain operations reference (SCOR) model*. *Supply Chain Management*, .

Janat Shah, (2016) *Supply Chain Management: Text and Cases, 2nd Edition*. [online] Available at: https://www.oreilly.com/library/view/supply-chain-management/9789353062538/ [Accessed 5 Feb. 2022].

Kalinov, I., Petrovsky, A., Ilin, V., Pristanskiy, E., Kurenkov, M., Ramzhaev, V., Idrisov, I. and Tsetserukou, D., (2020) WareVision: CNN Barcode Detection-Based UAV Trajectory Optimization for Autonomous Warehouse Stocktaking. *IEEE Robotics and Automation Letters*, 54, pp.6647–6653.

Kumar, S., Narkhede, B.E. and Jain, K., (2021a) *Revisiting the warehouse research through an evolutionary lens: a review from 1990 to 2019*. *International Journal of Production Research*, .

Kumar, S., Vishal, Sharma, P. and Pal, N., (2021b) Object tracking and counting in a zone using YOLOv4, DeepSORT and TensorFlow. In: *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*. Institute of Electrical and Electronics Engineers Inc., pp.1017–1022.

Lee, C.K.M., Lv, Y., Ng, K.K.H., Ho, W. and Choy, K.L., (2018) Design and application of internet of things-based warehouse management system for smart logistics. *International Journal of Production Research*, 568, pp.2753–2768.

Li, L., Su, Q. and Chen, X., (2011) Ensuring supply chain quality performance through applying the SCOR model. *International Journal of Production Research*, 491, pp.33–57.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A.C., (2015) SSD: Single Shot MultiBox Detector. [online] Available at: http://arxiv.org/abs/1512.02325.

Lototsky, V., Sabitov, R., Smirnova, G., Sirazetdinov, B., Elizarova, N. and Sabitov, S., (2019) Model of the Automated Warehouse Management and Forecasting System in the Conditions of Transition to Industry 4.0. *IFAC-PapersOnLine*, 5213, pp.78–82.

Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W. and Kim, T.K., (2021) *Multiple object tracking: A literature review*. *Artificial Intelligence*, .

Mabert, V.A. and Venkataramanan, M.A., (n.d.) *Special Research Focus on Supply Chain Linkages: Challenges for Design and Management in the 21st Century* Challenges for Design and Management in the 21st Century*. *Decision Sciences*, .

Mohamed, I.S., Capitanelli, A., Mastrogiovanni, F., Rovetta, S. and Zaccaria, R., (2020) Detection, localisation and tracking of pallets using machine learning techniques and 2D range data. *Neural Computing and Applications*, 3213, pp.8811–8828.

O’Shea, K. and Nash, R., (2015) An Introduction to Convolutional Neural Networks. [online] Available at: http://arxiv.org/abs/1511.08458.

Padilla, R. and P.W.L. and D.T.L.B. and N.S.L. and da S.E.A.B., (2021) A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. *Electronics*, [online] 10, pp.237–242. Available at: https://www.mdpi.com/2079-9292/10/3/279 [Accessed 15 Feb. 2022].

Patel, A.D. and Chowdhury, A.R., (2020) Vision-based object classification using deep learning for inventory tracking in automated warehouse environment. In: *International Conference on Control, Automation and Systems*. IEEE Computer Society, pp.145–150.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., (2015) You Only Look Once: Unified, Real-Time Object Detection. [online] Available at: http://arxiv.org/abs/1506.02640.

Redmon, J. and Farhadi, A., (2016) YOLO9000: Better, Faster, Stronger. [online] Available at: http://arxiv.org/abs/1612.08242.

Redmon, J. and Farhadi, A., (2018) YOLOv3: An Incremental Improvement. [online] Available at: http://arxiv.org/abs/1804.02767.

Ren, S., He, K., Girshick, R. and Sun, J., (n.d.) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. [online] Available at: http://image-net.org/challenges/LSVRC/2015/results [Accessed 27 Jan. 2022].

Taghizadeh student, E., (n.d.) *UTILIZING ARTIFICIAL NEURAL NETWORKS TO PREDICT DEMAND FOR WEATHER-SENSITIVE PRODUCTS AT RETAIL STORES*.

Tirkolaee, E.B., Sadeghi, S., Mooseloo, F.M., Vandchali, H.R. and Aeini, S., (2021) *Application of Machine Learning in Supply Chain Management: A Comprehensive Overview of the Main Areas*. *Mathematical Problems in Engineering*, .

Tjahjono, B., Esplugues, C., Ares, E. and Pelaez, G., (2017) What does Industry 4.0 mean to Supply Chain? *Procedia Manufacturing*, 13, pp.1175–1182.

Toorajipour, R., Sohrabpour, V., Nazarpour, A., Oghazi, P. and Fischl, M., (2021) Artificial intelligence in supply chain management: A systematic literature review. *Journal of Business Research*, 122, pp.502–517.

Wiley, V. and Lucas, T., (2018) Computer Vision and Image Processing: A Paper Review. *International Journal of Artificial Intelligence Research*, 21, p.22.

Wojke, N., Bewley, A. and Paulus, D., (2017) Simple Online and Realtime Tracking with a Deep Association Metric. [online] Available at: http://arxiv.org/abs/1703.07402.

Yang, J.X., Li, L.D. and Rasul, M.G., (2021) Warehouse Management Models Using Artificial Intelligence Technology with Application at Receiving Stage – A Review. *International Journal of Machine Learning and Computing*, [online] 113, pp.242–249. Available at: http://www.ijmlc.org/index.php?m=content&c=index&a=show&catid=114&id=1224.

Zhao, K., Zhu, M., Xiao, B., Yang, X., Gong, C. and Wu, J., (2020) Joint RFID and UWB Technologies in Intelligent Warehousing Management System. *IEEE Internet of Things Journal*, 712, pp.11640–11655.