VIDEO SURVEILLANCE SYSTEM BASED ON ACTION AND EVENT RECOGNITION WITH MOVING OBJECT DETECTION AND TRACKING

A Thesis Submitted to
the Graduate School of Engineering and Science of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Electronics and Communication Engineering

by
Tuğçe ELÇİ

July 2024
İZMİR

We approve the thesis of **Tuğçe ELÇİ**

Examining Committee Members:

Assist. Prof. Dr. M. Zübeyir ÜNLÜ
Department of Electrical and Electronics Engineering,
Izmir Institute of Technology

Prof. Dr. Mustafa A. ALTINKAYA
Department of Electrical and Electronics Engineering,
Izmir Institute of Technology

Assist. Prof. Dr. Başak Esin KÖKTÜRK GÜZEL
Department of Electrical and Electronics Engineering,
Izmir Democracy University

12 July 2024

Assist. Prof. Dr. M. Zübeyir ÜNLÜ
Supervisor, Department of Electrical
and Electronics Engineering,
Izmir Institute of Technology

Prof. Dr. Mustafa A. ALTINKAYA
Head of Department of Electrical and
Electronics Engineering

Prof. Dr. Mehtap EANES
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGEMENTS

First of all, I would like to thank my family, who have always been there for me and whose support I have always felt. I would like to thank my father, Güngör Elçi, who made a great contribution to my career and academic goals, and whose presence I will feel with me forever. I would like to thank my dear mother, Süreyya Elçi, who played a very important role in helping me overcome the difficulties I experienced in my academic life and in my normal life and who I know is always by my side. I would like to thank my brothers Bülent Tuğberk Elçi and Haktan Utku Elçi, whose support I have always felt. I would also like to thank Orhan Balotu for his support.

Human life has always had its challenging aspects. Knowing that a person contributes to himself, society, and science in these difficult times causes people not to give up on life and even to hold on to life more tightly. I would like to thank my company, Borusan Lojistik, who I know stood by me during my difficult process and supported me in continuing my academic life, and our general manager, deputy general manager, and valuable managers who took part in this organization, especially our Advanced Analytics and Artificial Intelligence department manager, Deniz Kantar, and my colleagues.

I would like to thank my advisor, Assist. Prof. Dr. M. Zübeyir ÜNLÜ, to whom I am grateful for his support and understanding regarding my graduate education, who broadened my horizons in line with my studies, and from whom I learned new things every day. I would also like to thank Prof. Dr. Bilge Karaçalı, who inspired me to enter graduate education and helped shape my academic journey.

Finally, I would like to thank every woman who contributes to science for helping me on this path, reminding me where I should be, and showing me what a woman can do.

ABSTRACT

VIDEO SURVEILLANCE SYSTEM BASED ON ACTION AND EVENT RECOGNITION WITH MOVING OBJECT DETECTION AND TRACKING

The rapid growth of the logistics sector in recent years has led to the expansion of warehouse areas and an increase in the number of equipment used, resulting in an increase in work accidents. Work accidents that occur in warehouses are mostly caused by carelessness, fatigue, intense work tempo, individual behavior, lack of experience, inadequate training, and negligence of employees. Therefore, a system that predicts person-equipment interaction in real time is needed to ensure in-warehouse reliability. Within the scope of the thesis, a comprehensive video surveillance system consisting of object detection, object tracking, action detection, and alarm classification components that will increase occupational safety in warehouse environments is proposed.

YOLOv7, used as the object detection methodology in this system, is a deep learning model that detects objects quickly and accurately in a single network pass. Deep SORT is a computer vision tracking procedure that assigns a unique identifier to each tracked object and uses deep learning during tracking. The action detection part of the system analyzes identifies actions and movements and recognizes anomalies and potential risks. Then, various alarm levels are estimated using the speed, tag, movement direction, and coordinate information of the person and equipment, and different alarm levels are generated depending on these estimated alarm levels.

Through this system, which has been tested to provide technological competencies such as real-time response and to work with a high success rate, accidents in warehouses will be predicted, alarms will be generated, and possible occupational accidents can be prevented to a large extent.

ÖZET

HAREKETLİ NESNE ALGILAMA VE İZLEME İLE EYLEM VE OLAY TANIMAYA DAYALI VİDEO GÖZETİM SİSTEMİ

Lojistik sektörünün son yıllarda hızla büyümesi, depo alanlarının genişlemesine ve kullanılan ekipman sayısının artmasına neden olarak iş kazalarının artmasına neden olmuştur. Depolarda meydana gelen iş kazaları çoğunlukla dikkatsizlik, yorgunluk, yoğun iş temposu, bireysel davranışlar, deneyim eksikliği, yetersiz eğitim ve çalışanların ihmalinden kaynaklanmaktadır. Bu nedenle depo içi emniyetin sağlanması için insan ve ekipman etkileşimini gerçek zamanlı olarak tahmin eden bir sisteme ihtiyaç vardır. Tez kapsamında depo ortamlarında iş güvenliğini artıracak nesne algılama, nesne izleme, eylem algılama ve alarm sınıflandırma bileşenlerinden oluşan kapsamlı bir video gözetim sistemi önerilmektedir.

Bu sistemde nesne tespit metodolojisi olarak kullanılan YOLOv7, nesneleri tek bir ağ geçişinde hızlı ve doğru bir şekilde tespit eden bir derin öğrenme modelidir. Deep SORT ise izlenen her nesneye benzersiz bir tanımlayıcı atayan ve izleme sırasında derin öğrenmeyi kullanan bir bilgisayarlı görme izleme teknolojisidir. Sistemin eylem algılama kısmı, anormallikleri ve potansiyel riskleri tanıyarak eylemleri ve hareketleri tanımlamak ve analiz etmek için tasarlanmıştır. Bu bölümde insan ve ekipmanların hız, etiket, hareket yönü ve koordinat bilgileri kullanılarak çeşitli alarm seviyeleri tahmin edilmekte ve bu tahmini alarm seviyelerine bağlı olarak da farklı alarm seviyeleri üretilmektedir.

Gerçek zamanlı müdahale ve yüksek başarı oranıyla çalışabilme gibi teknolojik yeterlilikleri sağlaması test edilen bu sistem sayesinde depolardaki kazalar tahmin edilecek, alarmlar üretilecek ve olası iş kazaları büyük ölçüde önlenebilecektir.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

Figure **Page**

Figure **Page**

LIST OF TABLES

CHAPTER 1

INTRODUCTION

With the rapid development of the logistics sector, digitalization processes have accelerated. With this acceleration, it is aimed to completely digitalize many processes such as collecting, stacking, preparing for distribution, customer-oriented pricing and distribution of products in warehouses. One can observe the impacts of Industry 4.0, which is considered as the industrial revolution, behind these targets. The terms "Smart Factory" and "Factory of the Future", which came into our lives with the impact of this revolutionary step, are the reflection of digitalization trends in logistics processes (Kayıkçı, 2018).

The logistics sector has annual growth rates of 15% in North America, 7-10% in Europe and 20% in Asia and Turkey (Öztürk, 2011). With the digitalization process in the logistics sector and the increase in customer demands in the supply chain, changes are observed in storage areas and product stacking methods. As the business volume grows, so does the workforce in warehouses, resulting in a corresponding increase in the utilization of equipment.

The increasing demand for supply chain management increases the number of equipment and people, as well as the number of interactions between people and equipment in warehouse areas. In addition, many reasons such as employees not being conscious about occupational health and environmental safety, non-compliance with the rules, and the use of equipment with narrow visibility increase the interaction between people and equipment in warehouses, causing near misses and work accidents.

Reducing accidents in warehouse environments depends on the early detection of undesirable interactions that may occur between people and equipment. Therefore, in this thesis, we worked on developing a video surveillance system with a deep algorithm structure that includes complex object detection, object tracking and action detection features to detect the interaction between human and equipment. In the study, YOLOv7, one of the current models, is used for object recognition, and the Deep SORT algorithm that follows this system provides object tracking in a real-time system. Then, using the

information acquired from these areas, features such as the object's speed, motion vector, and distance between objects are extracted and human-equipment interactions are classified through event identification. Depending on the classification result, different warning systems can be arranged within the warehouse.

1.1. Motivation

As a result of an analyze, summarized in Figure 1.1, examining the work accidents in 2018 of a private company in the logistics sector, which is one of the business areas where physical power, information, technology and automation work together, 21 out of 130 work accidents occurred in the form of impact/collision and 18 occurred in the form of pinching/crushing. It is also determined that 60 of 130 work accidents took place in the warehouse area as shown in Figure 1.2 (Baş and Köseoğlu 2019). When the density of accidents is examined, their distribution according to their causes and location shows that most of the accidents in the logistics sector occur in warehouse areas and the most common causes are falling, impact/collision, and jamming/crushing.

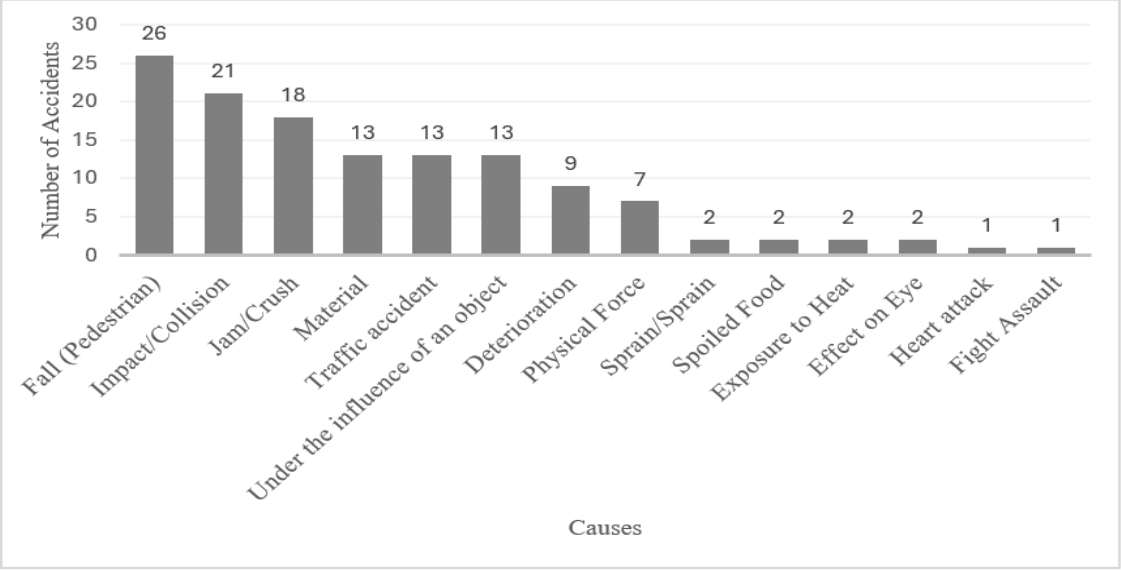

Figure 1.1. Distribution of the number of work accidents according to causes (Baş and Köseoğlu, 2019).

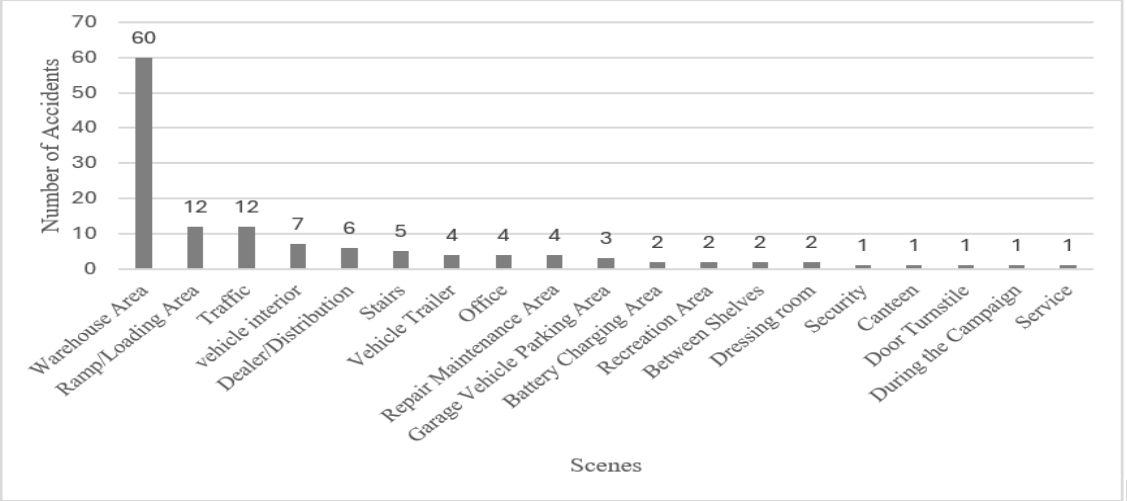

Figure 1.2. Distribution of work accidents by scene (Baş and Köseoğlu, 2019).

This study and its results show that it is a necessity to minimize or even eliminate the interaction between people and equipment in warehouses, and video surveillance systems that monitor person-equipment interaction to be used for this purpose are among the important systems needed in warehouses.

1.2. Thesis Goals and Contributions

The logistics sector is directly related to factors such as economic growth, globalization, and e-commerce. As trade volume increases, storage and transportation volumes increase in parallel. Therefore, warehouses have a very critical role in the logistics industry. Warehouses are closed or open areas within the supply chain where materials are stacked, stored, and designed according to material type in different sizes and features in order to protect and stock them for various purposes and to be used in different periods (Toktaş-Palut and Okçuoğlu 2019). Warehouses are temporary stock points used during the distribution of products. They contribute significantly to the effective execution of logistics activities as points that connect parts of the supply chain. Warehouses can be found as specially constructed stand-alone buildings, or they can be

located next to or within production facilities. Products and materials that can be considered raw materials for storage operations are stored on shelves or in stacks in warehouses. These materials enter the warehouse through warehouse ramps. The loading and unloading materials into the warehouse and other related tasks are facilitated by the employment of vehicles known as forklifts. In addition, there are particular equipment, such as forklift-like reach trucks, pallet trucks, and wheelbarrows, that enable the relocation of the products stacked in warehouses.

The warehousing industry is faced with a significant number of injuries in line with its increasing trade volume and importance. A statistical study conducted in the US in 2019 reported more than 38,000 incidents and 5.3 injuries per 100 workers ("Must-Know Warehouse Injury Statistics" 2024). Highlighting the significant contribution of accidental injuries creates a greater sense of urgency in solving this problem. Impact injuries, which are very common after death in warehouse accidents, reveal the importance of warehouse safety and the priority of precautions to be taken in this regard. The urgency of this issue has a significant impact on the capacity of individuals to ensure occupational safety in warehouses and is a driving force behind the development of all potential applications in this context. Upon reviewing Table 1.1, it is evident that 29 of the 1626 fatal occupational accidents in Turkey in 2014 occurred in the storage and supporting activities of transportation sector.

Table 1.1. The distribution of individuals who died or have been incapacitated due to occupational accidents (T.C. Ministry of Labor and Social Security, SSI Statistical Yearbook 2011/2012/2013/2014 and Kuyucu, 2016)

	The number of insured individuals who died as a result of work-related accidents	**Unable to Work Permanently as a Result of Accident**
Warehouse in Logistics	29	20
Total	1626	1421

In Figure 1.3, percentages of the distribution of lethal accidents in the field of storage and land transportation in the UK are given according to their causes. Here again, it is seen that the majority of accidents are caused by human-equipment interaction in warehouses, and it is revealed that precautions should be taken against these accidents by utilizing today's technologies.

When the accidents and accident possibilities in warehouses are evaluated both globally and in Turkey, it is appraised that observing pedestrian and equipment interactions within the warehouse will be an important step in preventing accidents. Preventing potential accidents can be achieved by minimizing the occurrences of human and equipment contacts, which have the capacity to result in minor injuries, severe handicap, or even fatality.

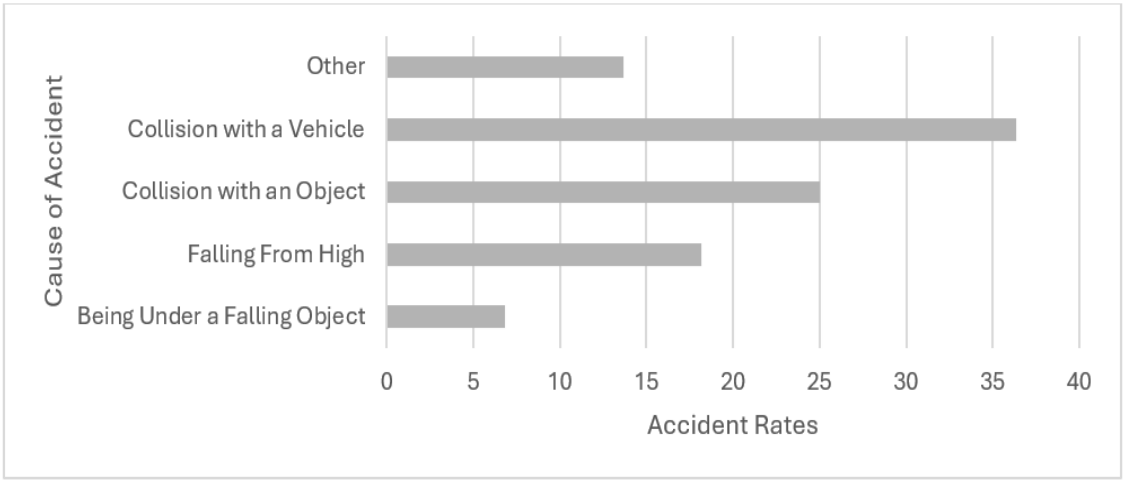

Figure 1.3. Percentage of fatal accidents by type in the road transport warehousing sector - UK example (Kuyucu, 2016).

Computer vision and artificial intelligence concepts, which have been integrated into our daily lives as a result of technological advancements, are solutions to numerous issues that have arisen from the past and continue to persist in the present. In the sphere of occupational health and safety, there has been a discussion of the development of new precautionary approaches and technological advancements in recent years. One of these approaches is the use of artificial intelligence. Artificial intelligence-supported

occupational health and safety technologies are important in identifying and eliminating risk factors. Since it is difficult for occupational health and environmental safety professionals to access many areas at the same time in different places, image processing and analysis technologies offer an effective solution to close this gap. With the digital transformation in the field of occupational health and safety, artificial intelligence applications have been developed in many areas, such as detecting workers in hazardous areas and hazardous situations through video analysis. Hence, it can be concluded that artificial intelligence-based technologies will produce effective results in the field of occupational health and safety and can offer a proactive approach in creating a safe working environment for warehouse workers.

Therefore, this artificial intelligence-based study aims to detect, classify and track objects, to detect human and equipment interactions, to evaluate possible accident probabilities and to generate warning alarms at certain levels as a result of this evaluation in logistics warehouses. With this system, possible accidents will be prevented, and occupational health and environmental safety inspections in warehouses will increase.

The system also contributes to the literature in terms of the methodology used, in addition to the contributions it provides as a safety system in warehouses in real life. When the system is evaluated in general, it uses an original combination of different algorithms in the object detection, object tracking, and action/event detection sections, and it has a structure that works properly in a certain flow situation. The primary framework of the system is to offer input to the action/event detection model and to prevent any interaction between humans and equipment in cases of anomalies. When evaluated from this perspective, object detection and object tracking components of the system provide input to the action/event detection section. Consequently, it may be asserted that the system has a complex structure.

Since the number of visual data representing the moment of the accident or before is quite limited, studies were carried out on completely normal data. While anomaly detections performed in the literature generally work visually, in this system, object detection, object tracking, speed and movement information, and the combination of all people and equipment in each frame are given as input. This enables the system to interpret binary situations rather than visual interpretation. Using collective input from all individuals and equipment in each frame requires more complex and comprehensive

analysis. This allows the evaluation of more complex data structures, unlike existing methods in the literature. In addition, after the action is detected, it becomes known in which frame the data containing the anomaly is located, so that it can be clearly predicted which person and equipment is most likely to experience an accident. Moreover, the classification of the anomaly after detection is among the contributions of the study. Situations exceeding the specified threshold values are classified under defined conditions. New approaches have been developed to determine the threshold value of the alarm by using experimental studies and mathematical modeling in alarm classification. This significantly improves the existing literature. In this case, the alarm level to be generated is determined as encounter, near-miss, or emergency.

In summary, the purpose of the study is to predict human and equipment interactions in warehouses with stacking areas in logistics by using computer vision and deep learning algorithms, and thus to prevent accidents that may occur in warehouses, to protect human health, and to create work environments with high safety measures. Furthermore, the development of the system not only serves practical purposes but also contributes to the literature due to the methodologies developed in the study.

1.3. Outline of the Thesis

The outline of the thesis organized as follows:

Chapter 1 - Introduction: The motivational content of the thesis, the purpose and contribution of the thesis, and finally the draft structure of the thesis are mentioned.

Chapter 2 - Related Works: A literature review regarding the solutions to the problems discussed in the thesis or similar problems is mentioned.

Chapter 3 - Research Background: It provides a comprehensive analysis of the methodologies that can be employed to address the problem, including extensive examination results, example studies, and their applications. The validation methods are also described.

Chapter 4 - Methodology: The core of this dissertation. It provides a detailed description of the methodology proposed. The methods used in the object detection,

object tracking, and action detection sections of the thesis and the datasets used for these methods are presented.

Chapter 5 - Experimental Results: The results obtained with the proposed methodologies are evaluated using both qualitative and quantitative methods.

Chapter 6 - Conclusion and Future Works: It provides conclusion about the proposed study and plans for the future work.

CHAPTER 2

RELATED WORKS

There are many studies in the literature on object recognition, classification, tracking, and analysis and classification of events and actions. However, since a similar approach to the problem in the field of logistics is rare and has not been studied sufficiently before, the approach on the literature review here was created by evolving similar problems in different nearby areas. Therefore, in this section, similar problems and studies addressed in the thesis in the literature are discussed. These studies also include analogous works conducted in different fields other than the logistics sector.

Since the general aim of the study is to predict the interaction between humans and equipment, the research here is focused on the action detection and classification, which is the main topic. The most recent survey by Vahdani and Tian (2022) states that an understanding of human behavior and activity analysis is necessary for real-world applications. However, the majority of real-world films are often observed to be long and choppy, which reduces the effectiveness of action detection algorithms. Therefore, this work addresses the task of temporal activity detection, which aims to identify and categorize the temporal boundaries of actions. The paper examines the performance of deep learning-based algorithms at various levels of supervision, such as fully supervised, weakly supervised, unsupervised, and semi-supervised. Additionally, it concentrates on spatiotemporal action detection, which determines activities in two dimensions: time and place. The study also evaluates the performance of the most recent techniques in comparison to the action detection benchmark datasets and assessment standards. Lastly, practical uses and potential future prospects for this kind of temporal action detection research are covered (Vahdani and Tian, 2022). In line with this review, it was concluded what the general approach to the study should be and what methods are used in solving a similar problem.

Some studies that are likely to be used in the field of logistics have been examined. In the literature, there are many methods used to detect actions that occur with human

behavior. The most common approaches used among these methods are those that use sensor technologies together with deep learning methods. Dawar and Kehtarnavaz (2018) claim that their study provides a deep learning-based detection and fusion technique for recognizing and determining actions of interest in continuous motion streams. It involves randomness and ongoing action flows where interesting actions blend together with uninteresting actions at random. The sensors used in the fusion system are the depth sensor in the camera used in the system and the accelerometer (wearable) sensor. Convolutional neural networks (CNNs) are being utilized to process the images from the depth sensor, and an integration of CNNs and long short-term memory networks is used to handle the inertial data from the accelerometer sensor. Each detection mode first segments all actions, then looks for activities that are relevant to a certain application. Combining the two sensory modalities at the decision level enables the detection of detected actions of interest. Two uses of the developed fusion system have been studied: hand signals for smart TVs and gestures toggling for home health monitoring. The results demonstrate how successfully the developed fusion system manages realistic, continuous action flows, claim Dawar and Kehtarnavaz (2018).

There are many approaches used to perform action detection in videos. According to a study conducted by Hou, Chen, and Shah (2017), it is stated that deep learning achieves excellent results in the fields of object detection and image classification. Nevertheless, the influence of deep learning on video analysis has been restricted as a result of the intricate nature of video data and the absence of labeling. Previous convolutional neural network (CNN)-based video action detection approaches generally consist of two main steps: frame-level action suggestion generation and fusion of inter-frame suggestions. In addition, the majority of these techniques handle temporal and spatial variables independently using a two-stream CNN framework. This paper proposes the Tube Convolutional Neural Network (T-CNN), an end-to-end deep neural network for action identification in videos. Based on 3D convolution characteristics, the suggested architecture is a unified deep network that can identify and localize activity. Initially, a video is split into clips of equivalent duration, and then a set of tube proposals in each clip is generated based on 3D Convolutional Network (ConvNet) features. Finally, tube suggestions from different clips are connected together using network flow and spatio-temporal action detection. T-CNN surpasses the current state-of-the-art in the classification and localization of actions in both trimmed and unedited videos, as

evidenced by extensive experiments on multiple video datasets (Hou et al. 2017). The examination of this study helped to conclude that temporal and spatial analysis was required in this thesis.

Traffic surveillance systems also offer solutions to similar problems. The movement of a vehicle is observed and certain movements are detected. In the study of Yadav, Renu, Ankita, and Anjum (2020), which provides information about the serious effects and causes of road traffic accidents on human life, it is stated that approximately 1.35 million people are affected by accidents every year, and the injuries caused by these accidents affect 20 to 50 million people. Incompatibility between organizations and inadequate enforcement of traffic laws are two important factors that contribute to the occurrence of these incidents, which in turn increases accident rates. Risk factors such as excessive speed, drunk driving, distraction, inadequate infrastructure, unsuitable vehicles, and violation of traffic rules can also cause accidents. In order to respond quickly to such accidents, a detection system using different technologies is required. Various technologies, such as the Global Positioning System (GPS), the Global System for Mobile Communications (GSM), and mobile applications, form the basis of such systems. Additionally, the use of convolutional neural networks (CNN) used in object identification and tracking technology, together with long short-term memory (LSTM), is also mentioned. This study provides an overview of technologies related to automatic road (traffic) accident detection system and explores solutions to reduce the number of accidents (Yadav et al., 2020). In this study, accident detection was performed with the approach created using deep learning methods. In other words, it is aimed to detect an accident when it occurs.

The approaches examined are generally designed to detect the incident while it is occurring or after it has occurred. Since there is no method that can be used to prevent these accidents, it is aimed to examine similar approaches. In the study by Bakheet and Al-Hamadi (2022), a new image-based framework is presented for real-time vehicle accident prediction and detection based on motion-time patterns and fuzzy time slicing. The presented framework proceeds step by step. It starts by automatically detecting moving objects (i.e., vehicles on the road or pedestrians on the sidewalk), then dynamically tracking the detected moving objects based on timed patterns, clustering, and supervised learning. Subsequently, a comprehensive collection of local features is extracted from the timed patterns of moving objects. Finally, an effective deep neural

network (DNN) model has been trained on the extracted features, thereby detecting abnormal vehicle behavior patterns just before an accident. Experiments on real-world vehicle crash videos show that the framework can achieve highly positive results compared to existing approaches; a 98.5% hit rate with a 4.2% false alarm rate was achieved, and it can ensure predictable delays for the purpose of monitoring and surveilling real-time traffic (Bakheet and Al-Hamadi, 2022). The system structure created has similar purposes to our study, in that it has a real-time working structure and takes precautions before an accident occurs.

Studies conducted using images in logistics warehouses have been examined. In the research (Ren et al., 2022), an approach is presented to resolve the issues encountered in the pallet tying accuracy and recognition rate of existing techniques. This system is intended to function as an intelligent forklift merchandise precision transfer system. The system is employed to automatically identify containers that require transportation. Once directed to the appropriate pallet area, the smart forklift uses its camera to take photos of the pallets and uses a deep learning-based recognition algorithm to determine the exact location of any pallet. Finally, the forklift positions the pallet accurately using a high-precision control algorithm. This system significantly increases the recognition rate by introducing the concept of small target detection into the track target recognition system. The Yolov5 algorithm is used to calculate pallet positions, increasing the coverage area and recognition accuracy of the algorithm. This system requires fewer sensors and indicators than prior approaches; it just gives a greater recognition rate and accuracy. Real-world data experiments demonstrate that the pallet recognition rate exceeds 99.5% and the pallet tie accuracy is 100% (Ren et al., 2022). Methodologically, similar approaches are observed with the deep learning method used in our study to detect objects.

Since similar studies that were presented as solutions to similar problems in the field of logistics could not be found, solutions produced for different problems were evaluated. The solution addressed by this thesis is to provide a proactive approach in the field of logistics for the protection of human health. An approach with high accuracy rates is provided with the deep learning methods used in the system. The proactive approach of the system will prevent possible accidents. With this approach, a possible interaction is detected in advance rather than an accident being detected. At the same time, it will be easier to track the violations of the rules of the equipment and people in the warehouse.

CHAPTER 3

RESEARCH BACKGROUND

In this chapter, research on the methodologies that can be utilized in the study, which was designed to minimize or even eliminate accidents caused by person-equipment interaction in logistics warehouses, is addressed. Different research has been conducted for the three main parts that make up the system. For this reason, this chapter consists of three main headings. These are object detection research, object tracking research, and action detection research.

3.1. Object Detection Research

Similar to the perception system of the human eye, acquiring and using qualitative information about objects in images is one of the primary objectives of image processing. Many image processing methods have been developed to meet needs such as detection, identification, classification, and tracking of objects. In particular, finding the target object in images and not losing this object over time is frequently employed in applications in several fields. However, the fact that the object to be tracked is in a variable environment is a fundamental problem that makes object tracking and analysis difficult. Various methods have been developed to overcome these difficulties and ensure successful tracking of the object (Hanbay and Üzen, 2017). There are different methods and application areas for object detection.

Moving-object detection is the first step in video analysis. This process is performed on each frame or the first time the object appears in the video (Joshi and Thakore, 2012). Object detection involves detecting objects belonging to a specific class within an image. This aims to find all instances of objects from known classes, such as individuals, automobiles, or faces within an image. Often, only a few objects are present

in the images, but it is important to acknowledge that these objects may be in various possible locations and scales and therefore need to be investigated (Amit et al., 2021).

Zhao et al. (2019) provide that object detection is related to video analysis, image understanding, and image processing. This relationship can be proven by the fact that object detection algorithms can accurately identify objects in images and videos, improving analysis and processing processes in these areas. Object detection methods started with traditional methods and continue to develop with deep learning methods today. While there are many problems, such as architectural structure, in traditional methods, it is said that these problems are solved with deep learning methods (Zhao et al., 2019).

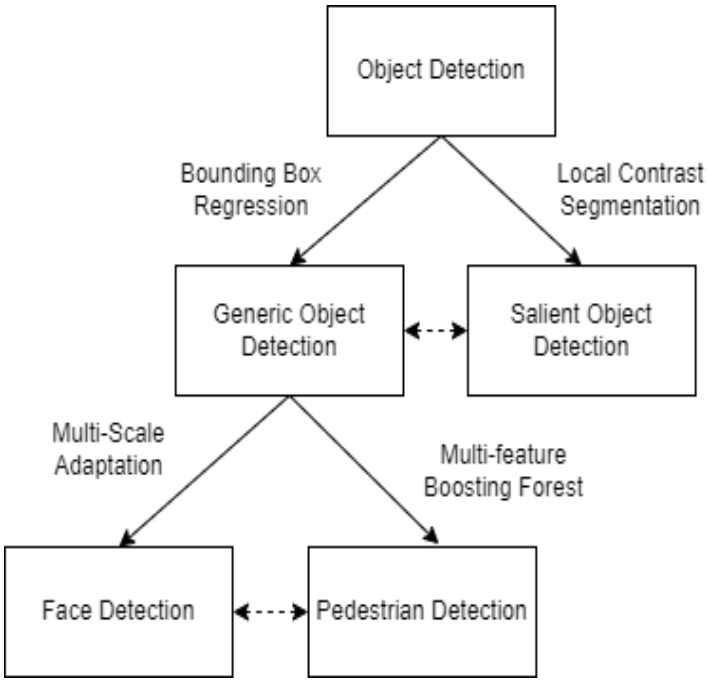

Figure 3.1. Object detection and its application domains (Zhao et al., 2019)

The models represented in numerous application domains and their distinct characteristics are summarized in the schematic overview illustrated in Figure 3.1 (Zhao et al., 2019). Boundary box regression is employed to detect general objects, while local contrast enhancement and pixel-level segmentation are employed to identify salient

objects. Techniques such as multi-scale adaptation and multi-feature fusion/boosting forest are employed to implement pedestrian and face detection, which is closely related to generic object detection. The relationships between object detection techniques in various application domains are illustrated in this figure. It also emphasizes the distinctions between images with conventional structures, such as pedestrian and face images, and images with more intricate structures, such as general objects and scene images (Zhao et al., 2019).

3.1.1. Generic Object Detection

Generic object detection has to deal with different degrees of variation in various classes of objects. According to Wang, Yang, Zhu, and Lin (2013), this requires descriptive and flexible object representations that are effective for evaluating many locations, while the computations are manageable. Generic object detection based on CNN architectures is achieved by bounding box regression (Zhao et al., 2019). This form of detection usually attempts to detect a wide range of natural categories, unlike specific object category detection, where a specific predefined category (e.g., faces, pedestrians, or vehicles) potentially exists. Despite the existence of thousands of objects in the visual world, the research community is currently more focused on the localization of configured objects (e.g., automobiles, human faces, bicycles, and planes) and articulated objects (e.g., people, cows, and horses) (Liu et al., 2019). The spatial position and width of the object can be loosely defined in ways such as an axis-aligned rectangle (a rectangle that tightly surrounds the object), a precise pixel-based segmentation mask, or a closed boundary. As seen in the examples in Figure 3.2, it is known that bounding boxes are the most frequently used method in the literature today for assessing general object detection algorithms. (Everingham et al., 2010; Zhang et al., 2013; Lin et al., 2014).

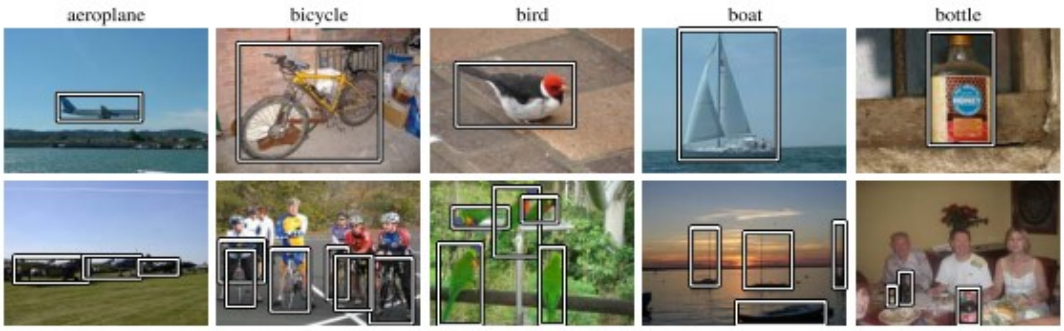

Figure 3.2. Example of the generic object detection (Everingham et al., 2010)

3.1.1.1. Bounding Box Regression

Bounding box regression is a widely used technique to improve or estimate localization boxes in modern object detection methods. Traditionally, regressors are trained to adjust region proposals or predefined anchor boxes to closely match the bounding boxes of target object classes (Lee et al., 2019).

Object detection is a crucial task in computer vision that involves identifying and locating objects within images or videos. Unlike image classification, which focuses solely on recognizing which objects are present, object detection also requires determining their precise location in the image. This is often represented by bounding boxes surrounding objects. Traditional object detection methods are based on sliding window techniques and pattern matching. Candidate regions are extracted using sliding windows at different scales, and their features are compared with ground truth features to determine the object class and location. However, these methods have limitations. The large number of candidate regions that are generated frequently results in a high level of computational complexity, and can suffer from poor performance due to differences in object appearance and geometric deformations (Sun et al., 2019).

Figure 3.3 Object detection example: find and locate objects in images

According to Sun et al. (2019), in recent years, numerous object detection algorithms that are based on deep learning such as R-CNN (Region-based convolutional Networks), faster R-CNN (Faster Region based Convolutional Networks), SSD (Single Shot Multibox Detector), FPN (Feature Pyramid Network), YOLO (You Only Look Once), and mask R-CNN (Mask Region-based Convolutional Network) have been developed, and significant gains in performance have been made. Typically, these algorithms have modules for suggesting potential regions and refining the bounding boxes. The candidate region suggestion module detects prospective locations that may contain items, while the BBR module enhances the bounding boxes of these regions to achieve more precise placement. Despite the design differences, most object detection algorithms include both modules. BBR, which was first introduced by Felzenszwalb et al., has demonstrated efficacy in enhancing the localization accuracy and mean average precision (mAP) of object detection algorithms., especially in industrial applications such as robotics and autonomous vehicles (Sun et al., 2019).

Figure 3.4 shows a schematic diagram of bounding box regression. The green frame represents the actual bounding box of the manually marked object, the blue frame represents the suggestion bounding box, and the red frame represents the estimated bounding box resulting from the BBR module (Sun et al., 2019).

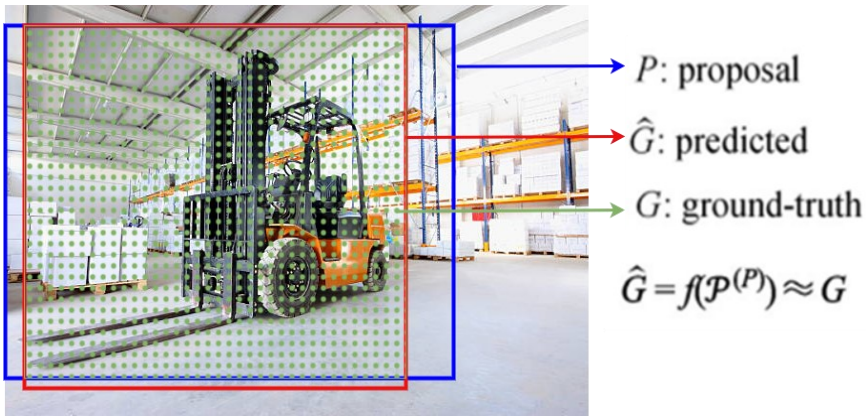

Figure 3.4. Illustration of bounding box regression (Sun et al., 2019)

$$f : \mathrm{P}^{(P)} \rightarrow \hat{G} \tag{3.1}$$

The regression model consists of four distinct regression functions according to $\hat{G}$.

$$\begin{cases} G_x = f_x(\mathrm{P}), \\ G_y = f_y(\mathrm{P}), \\ G_w = f_w(\mathrm{P}), \\ G_h = f_h(\mathrm{P}). \end{cases} \tag{3.2}$$

According to Sun et al. (2019), absolute coordinates are greatly reliant on the size of the target image and can be difficult to train and adapt to. Therefore, the relative positions of the proposed bounding boxes and the actual bounding boxes (more accurately, the proposed bounding boxes and the actual bounding boxes) are used. Relative positions are often represented by bounding box transformation coefficients (BTC), which are a set of parameters that determine the process of transforming suggested bounding boxes into actual bounding boxes. Applying a logarithmic transformation to the scale parameters can allow bounding box scales to span a wider range of values and can be useful in ensuring convergence during model training. As a result, the bounding box regression formula is:

$$\begin{cases} t_x = \frac{G_x - P_x}{P_w} = f_x(\mathrm{P}), \\ t_y = \frac{G_y - P_y}{P_h} = f_y(\mathrm{P}), \\ t_w = \log\frac{G_w}{P_w} = f_x(\mathrm{P}), \\ t_h = \log\frac{G_h}{P_h} = f_x(\mathrm{P}). \end{cases} \tag{3.3}$$

3.1.1.2. Multi-Scale Adaptation

Multi-scale adaptation is a technique used in object detection and recognition systems. It is an object recognition technique that addresses the variability of objects by considering that things can appear in many sizes and scales. This technique is based on the idea that objects can have multiple structures (varying sizes and scales). Object detection algorithms should be able to predict that objects can be of different scales and sizes. Multi-scale object detection is essential for analyzing remote sensing images. While traditional feature pyramid networks use multi-level feature extraction methods to accommodate different sizes of objects, recent deep learning-based object detection models have avoided these pyramid representations (Liu et al., 2024).

Multi-scale adaptation processes different resolutions of the same image because it uses the pyramid structure by processing at different resolutions. As a result, this approach can be used to recognize object images of various sizes. For example, if an object is large at the top left corner of an image and small in the lower right corner, this method should be used. Thanks to this feature, multi-scale adaptation can be considered a more flexible and general method. This method adapts better even though the objects are of different scales and sizes.

(a) (b)

Figure 3.5. Examples and challenges of object detection include (a) large differences in object scales; and (b) similarity between objects of similar scales (J. Liu et al., 2022)

As seen in Figure 3.5, one of the difficulties experienced in object detection is large differences in object scales, while another difficulty is similarity between objects of similar scales. It is easier to work on these problems with multi-scale adaptation.

3.1.2. Salient Object Detection

Salient object detection is a method to recognize significant and noticeable objects in a picture. With a structure similar to the human eye, it examines the distinctions between items in a picture in order to find objects that may draw human attention. This object detection method uses certain features in the image to evaluate them. These are the properties of the image's pixels such as color, brightness, border, contrast. The combination of these features creates a score. According to this score result, remarkable objects are determined. This approach is implemented in numerous domains, including image processing, autonomous driving, video analysis, advertising, and robotics. Salient object detection aims to accurately identify and distinguish the most prominent objects in images. This method constitutes an important first step in various systems such as video-object segmentation, light field image segmentation, image-sentence matching, person re-identification, and pattern segmentation (Li et al., 2021).

Salient object detection aims to segment all objects in images and videos and find all significant object structures without any prior knowledge. The priorities it uses when

estimating salience generally make use of low-level priorities such as contrast priority, object priority, background priority, intensity priority, and center. As illustrated in Figure 3.6. (Xi et al., 2019), one of the techniques employed in this context, LPS (label propagation saliency), combines front boundary and object priors into an intra- and inter-label propagation scheme for saliency extraction (Li et al., 2015). The other method shown in the figure, wCtr (Weighted Contrast), performs object detection by inferring saliency using the boundary connection (Zhu et al., 2014).

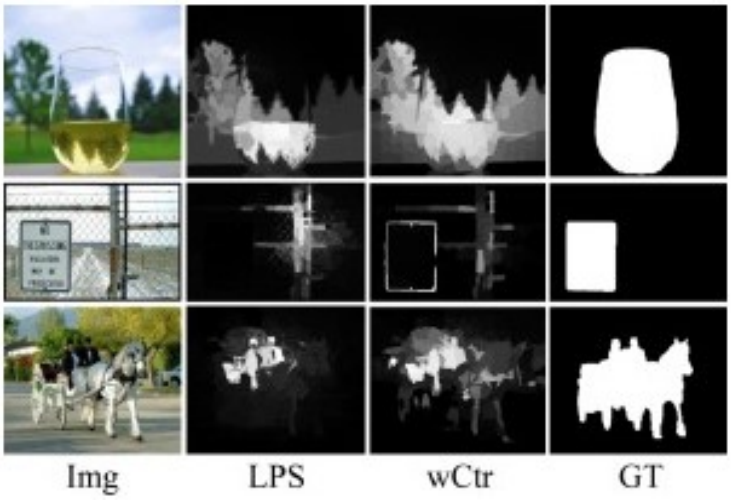

Figure 3.6. Salient object detection results (with LPS, wCtr and ground truth (GT)) (Xi et al., 2019)

Figure 3.6 clearly demonstrates that while these methods work in simple scenes, they fail in complex scenes (Xi et al., 2019).

As a result, salient object detection detects the objects that stand out first. Secondly, it divides all objects into sections. The saliency map is the model's output, in which the intensity of each pixel denotes the likelihood that it is associated with salient objects. This shows that the salient object detection model is a solution to an issue with figure/ground segmentation, and its purpose is to separate the striking object in the foreground from the background. With this feature, the salient object detection model differs from the traditional image segmentation problem (Borji et al., 2015).

3.1.3. Deep Learning Models in Object Detection

According to Pathak et al. (2018), the general definition of object detection is determining which class the object belongs to and estimating the location of the determined object using the bounding box. Object detection has different names depending on the number of classes. These are single-class object detection and multi-class object detection. As shown in Figure 3.7., object detection is identified as an important step in visual recognition. During object detection, there are steps such as verification of the object, detection and localization of the object, classification, naming, and finally description.

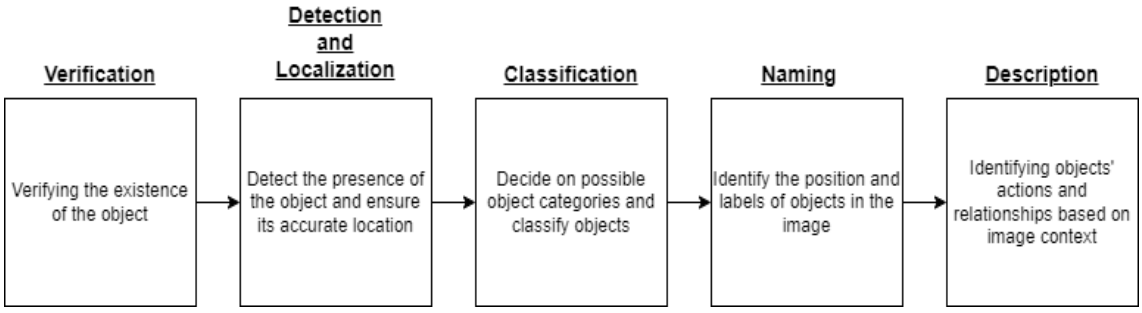

Figure 3.7. Object detection as the most important step in visual recognition activity (Pathak et al. ,2018)

Deep learning models are commonly utilized in the area of object detection due to their high performance regarding accuracy and speed. Convolutional neural networks (CNN) are utilized to automatically extract characteristics from raw image data in these models. As a result, object detection occurs with higher precision. The most commonly used models with deep learning architectures in object detection are convolutional neural networks (CNN), region-based convolutional neural network (R-CNN), fast R-CNN, faster R-CNN, single shot multi-box detector (SSD), mask R-CNN, and You Only Look Once (YOLO).

Recently, the use of artificial neural networks has increased with the use of deep learning in this field. These models are inspired by biological structure and perform very well. Artificial neural networks (ANN) are computational processing systems that have a similar structure to the biological nervous system. ANNs consist of many interconnected neurons that work to learn from input and optimize the final output. The basic structure of an ANN is given in the figure (O'Shea & Nash, 2015). The input, which is usually a multidimensional vector, is kept in the input layer, and this layer distributes the input to the hidden layers. Hidden layers perform an evaluation of information from the previous layer. Evaluations made in this layer are carried out by measuring whether a random variable will affect the outcome positively or negatively. This process is called the learning process. Considering this situation, having multiple hidden layers stacked on top of one another is called deep learning.

Table 3.1. Summary of deep learning models and their features (Olorunshola, Jemitola, and Ademuwagun 2023)

Model	Description	Speed	Accuracy	Use Case
CNN	Basic CNN for image classification and feature extraction	Slow	Moderate	Feature extraction, image classification
R-CNN	Generation region proposals and uses CNN for classification	Slow	High	Object detection with high accuracy
Fast R-CNN	Improved R-CNN with a single-stage training process; It uses ROI pooling.	Faster than R-CNN	High	Object detection with improved speed
Faster R-CNN	Improved R-CNN with Region Proposal Network (RPN) to detect faster	Faster than Fast R-CNN	Very High	Real-time object detection

Cont. on next page

Table 3.1. cont.

SSD	Its network detects ob jects in a single pass.	Fast	High	Real-time object detection
Mask R-CNN	Improved version of faster R-CNN with additional mask prediction branch; Sampling provides segmentation.	Slower due to segmentation	Very High	Instance segmentation
YOLO	A single convolutional network estimates multiple bounding box and class probabilities simultaneously.	Very fast	High	Real-time object detection

As given in the Table 3.1, a summary table has been created with the features of all models that can be used in the object detection stage. As seen in the table, YOLO has features that can be considered successful in real-time projects.

3.1.3.1. Convolutional Neural Network (CNN)

In recent years, deep learning has performed well in solving problems in different areas including visual recognition, speech recognition, and natural language processing. Among the deep neural networks used differently in this field, convolutional neural networks (CNN) are exceedingly preferred (Gu et al., 2018). CNN have a similar structure to ANN. For CNN, which is made up neurons that optimize themselves through learning, each neuron receives a new input and performs an operation (O'Shea & Nash, 2015). Convolutional neural networks (CNN) have a similar structure to artificial neural networks (ANN). From raw image vectors as input to the result for CNN, the entire network represents a single weight. The last layer contains the loss functions associated with classification. The only difference between CNN and ANN is the area of image

pattern recognition, which is a widely used area of CNNs. This enables the features of the image to be encoded. As a result, network image foundations have become possible to solve problems, and a reduced number of parameters are required for the model to work (O'Shea & Nash, 2015).

CNN have a different architecture with hidden layers. Convolutional neural networks have three common hidden layer structures: convolutional, pooling, and fully connected. Among these common hidden layers, the convolutional layer and pooling layer are layers specific to convolutional neural networks (Dai, 2021). Figure 3.8 explains the working principle of a CNN in its simplest form. The image in the figure represents an input image (for example, a forklift). This image is converted into various feature maps by passing through successive convolution and pooling layers. These feature maps are used to extract different elements and details in the image. The last layer usually contains fully connected layers and performs final operations such as classification (O'Shea & Nash, 2015).

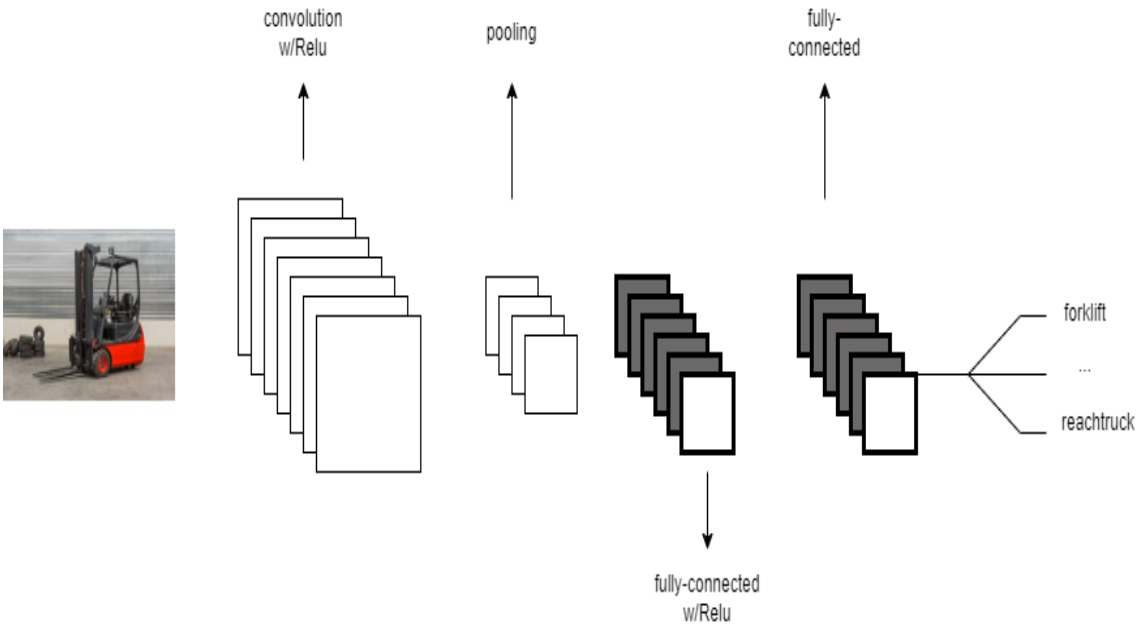

Figure 3.8. Example of basic CNN architecture (O'Shea & Nash, 2015)

The convolutional layer is the initial layer that extracts features from an image. Since the pixels of the image are associated only with adjacent and nearby pixels, this layer allows to maintain the relationship with different regions of the image. In this layer, filtering is done with a smaller pixel filter. In this case, the image size is achieved without losing any relationship (Hossain & Sajib, 2019). The convolutional layer initially analyzes the raw data (e.g., image pixel values) in the neural network by utilizing the convolution process on this data. It can identify a variety of features, including boundaries, textures, and more intricate patterns, by employing a variety of filters. Consequently, the convolutional layer concentrates on the extraction of low-level features from the input data and transmits this information to the subsequent layers. Figure 3.9 shows the convolution operation, which moves the filter over the image and sums the pointwise products between the filter and the overlapping region at each place. This operation generates a new feature map. Each result is the result of the filter's multiplication and addition with the associated image region.

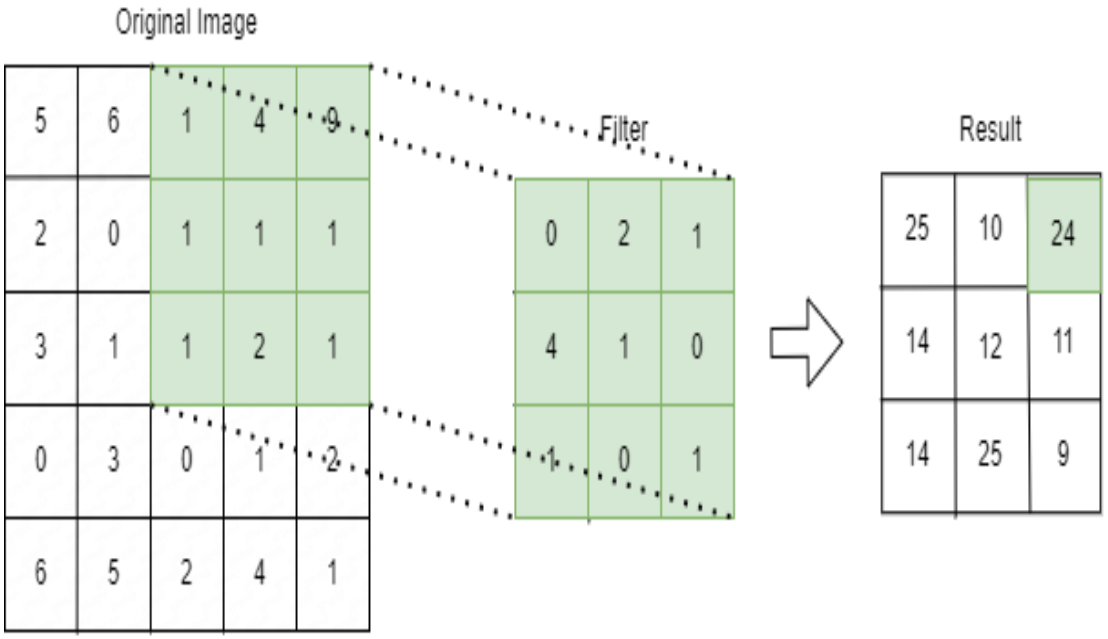

Figure 3.9. Example of convolution

The pooling layer also solves the complexity by decreasing the count of parameters. It is the layer applied to solve the overfitting problem. There are two different methods: max pooling and average pooling (Hossain & Sajib, 2019). Figure 3.10 shows instances of maximum and average pooling procedures on an image. During the max pooling operation, 2x2 sections of the image are picked, and the highest pixel value in each area is chosen. For example, the 2x2 square in the upper left has the values 8, 6, 2, and 0. The maximum value for this area is 8, and it is added to the top-right matrix as a result of max pooling. Average pooling involves selecting 2x2 areas and averaging the pixel values inside them. For example, the average of 8, 6, 2, and 0 in the 2x2 space at the upper left is 4. This value is added to the bottom right matrix as a result of average pooling.

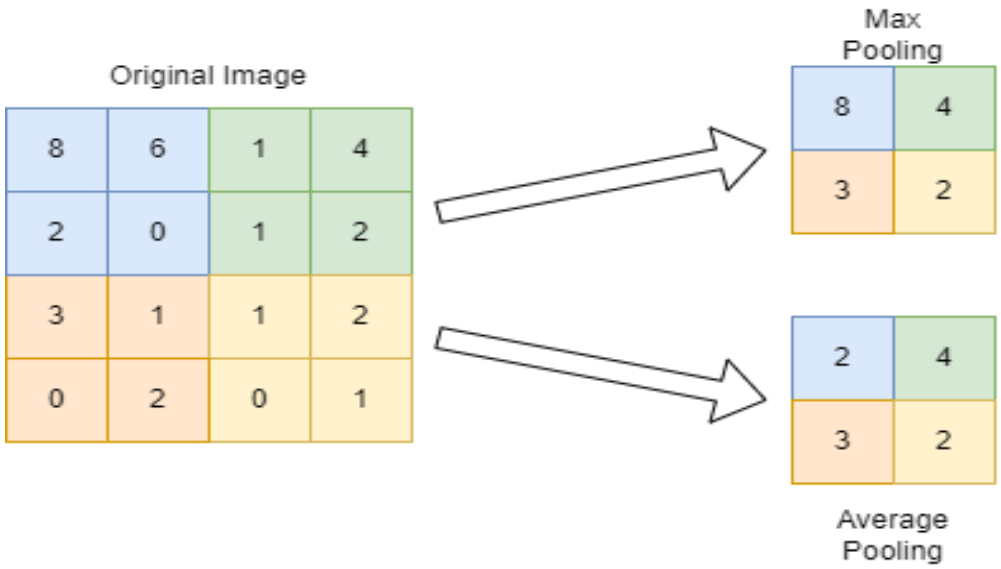

Figure 3.10. Max pooling and average pooling example

The fully connected layer contains neurons from two adjacent layers that are also connected to them, and they do not rely on any one layer (Hossain & Sajib, 2019). Together with the pooling layer, it greatly reduces time-space warping (O'Shea & Nash, 2015). Figure 3.11 depicts the fully linked layer structure in a deep learning model. These layers are typically employed as the final step of CNN, providing basic information for tasks such as categorization. The input layer receives the model's input data. This data is

typically derived from feature maps or preceding layers. The Fully Connected Layer is a structure in which each neuron connects to all neurons in the previous layer. This layer applies weight and bias values to the data from each input. As a result of these processes, the features that the model must learn become apparent. The output layer delivers the model's ultimate output. This output contains the probability of belonging to a specific class (for example, the type of an object) in classification issues (Kalaycı & Asan, 2022).

CNNs have become one of the most potent algorithms in the field of deep learning. They can be used with their constantly evolving structure and it is applicable to certain problems.

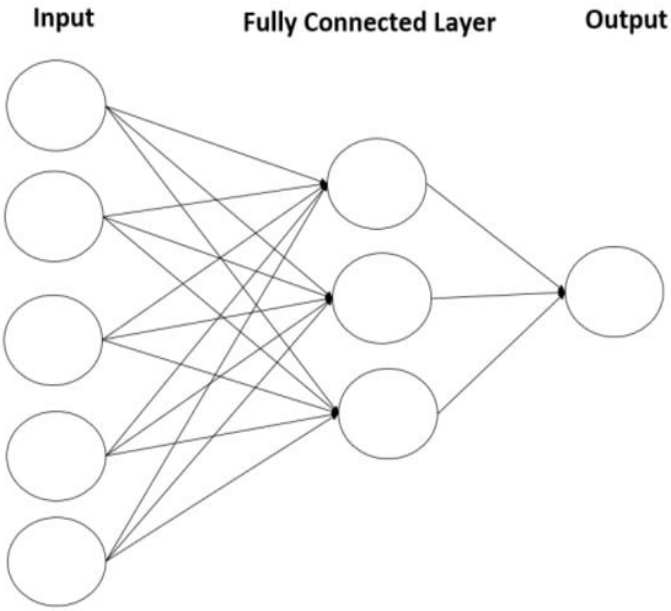

Figure 3.11. Fully connected layers for CNN (Kalaycı & Asan, 2022)

3.1.3.2. You Only Look Once (YOLO)

You Only Look Once (YOLO) is a very popular and widely used algorithm (Sultana, Sufian, & Dutta, 2020). About the history of YOLO, it is possible to assert that the first version of YOLO began to be used in 2015, when Redmon and others introduced it. Different versions of YOLO, which became very popular in the field object detection

with its introduction, have been developed. After this, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLO-LITE, YOLOv7, and YOLOv8 continued to be a solution to the problems with developments (Jiang et al., 2022).

YOLO models are suitable for real-time data prediction due to their high accuracy and computational efficiency. Due to its architecture, YOLO works on a single convolutional neural network, which facilitates object identification in images with different backgrounds (Jernbäcker, 2022). According to Redmon et al. (2016), in normal life, when people look at an image, they perceive the objects in the image, their location, and their interactions. In different systems, classifiers are reused, and this is how object detection occurs. The approach of YOLO models differs from different systems in this regard. The approach treats object detection as a single regression problem, including coordinates of bounding box and probabilities of class from the pixels of the image. The general structure of YOLO is given in the Figure 3.12. A convolutional neural network estimates multiple boundary boxes and their classification probabilities at once and simultaneously. These models learn from the entire image and optimize detection performance immediately after learning. It provides advantages over traditional object detection models.

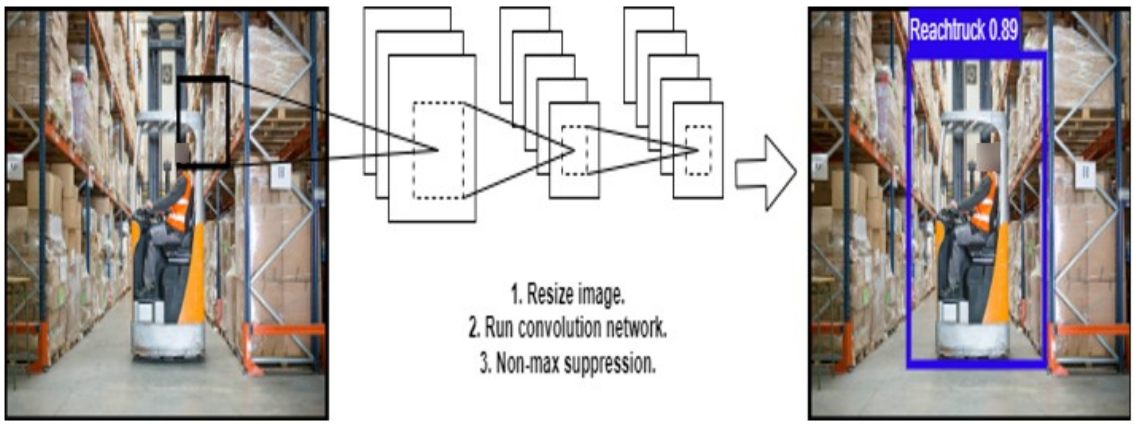

Figure 3.12. General Structure of the YOLO detection system

YOLO provides an advantage by being faster than traditional models. Additionally, since it approaches the solution as a regression problem, it does not contain

a complex pipeline structure. YOLO works very efficiently in object detection problems on real-time images, as it processes very quickly, provided the necessary hardware is available (Redmon et al., 2016). Due to its architecture, YOLO processes the image by dividing it into an *S*x*S* grid. Detection of the object occurs on the grid where the object's center is located. Each grid cell estimates the boundary box B and the confidence score of this boundary box and checks whether an object is in this box and the accuracy of the box it predicts. The confidence score is obtained with the following formula:

$$\text{Confidence Score} = \Pr(Object) * IOU_{pred}^{truth} \tag{3.4}$$

where $\Pr(Object)$ is the probability of the existence of the object, and IOU_{pred}^{truth} (Intersection over Union) is the overlap rate between the ground truth and the estimated predicted box. The accuracy of the model is measured with this formula.

If the cell contains no objects, the confidence score is set to zero. Otherwise, the confidence score should be identical with the intersection over union (IOU), which is the overlap value between the predicted box and the actual box (Redmon et al., 2016).

For each boundary box, five prediction values are created: *x, y, w, h*, and a confidence score. (*x,y*) represents the box's center coordinates, *w* and *h* signify the width and height when the whole image is evaluated, and the confidence score represents the IOU. Conditional probabilities are estimated for each grid cell. These values vary depending on whether or not an object is present in the grid cell. For each grid cell, a set of class probabilities is estimated without being associated with the number of boxes. Class-specific confidence scores for each box are obtained with the following formula:

$$\Pr(Class_i | Object) * \Pr(Object) * IOU_{pred}^{truth} = \Pr(Class_i) * IOU_{pred}^{truth} \tag{3.5}$$

This score also reflects both the likelihood of the class being present in the box and the fit between the predicted box and the object.

Figure 3.13 illustrates the operation of the YOLO (You Only Look Once) object detection algorithm. Initially, the image is partitioned into a grid with SxS dimensions. This illustration illustrates a scenario that includes a dog and a bicycle. Bounding frames for specific objects and confidence scores for their presence are estimated in each grid cell. The accuracy of this estimate and the reality of the object's presence in the compartment are both indicated by the confidence score. Additionally, the probabilities for specific classes (e.g., bicycle, dog) are included in each grid cell. The class probability map is employed to ascertain the class to which the objects belong. Ultimately, the final object detection is accomplished by combining the bounding boxes with high confidence scores and the objects contained within them. A bicycle and a dog are identified in this instance.

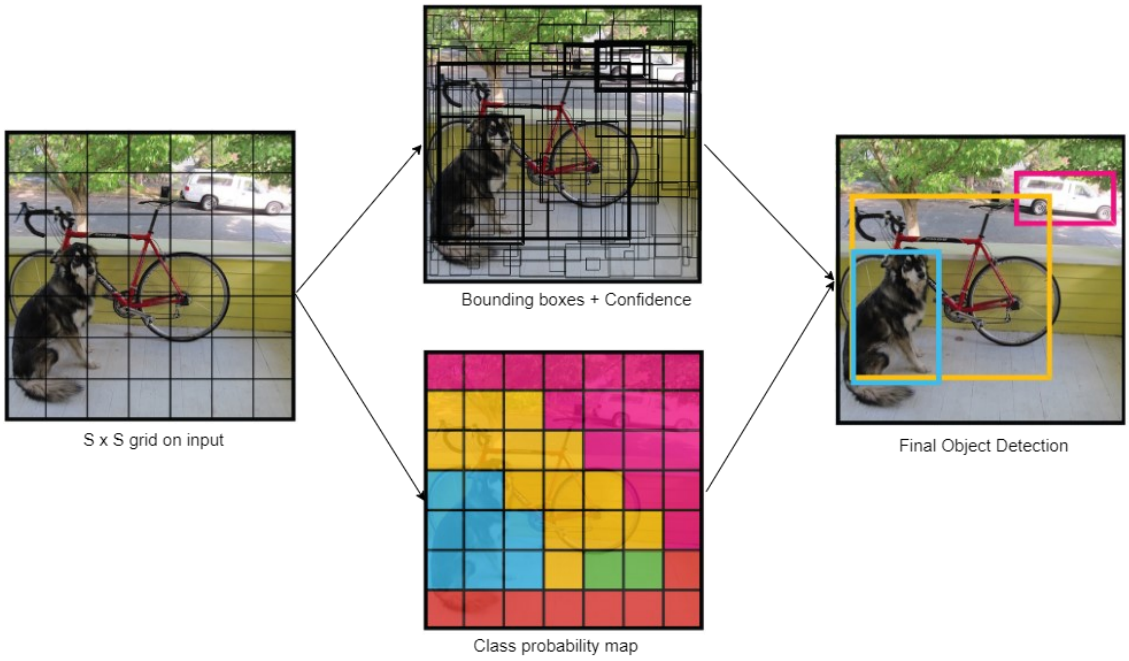

Figure 3.13. Example of object detection with YOLO (Redmon et al., 2016)

When evaluated, YOLO works successfully in real-time object detection. Apart from this, it has its own development story. There are certain differences between the YOLO model, which was introduced in 2016, and the versions developed day by day. Version differences are given in the table. Detailed information about the YOLO model

is given in the Table 3.2. Apart from this, information about the year the model was released, the main features of the model, and the prominent information for evaluating its performances are included. When analyzing the table, it is observed that the YOLOv7 model is more successful than other models (Quach et al., 2023).

Table 3.2. Comparison of YOLO models

Model	Year	Main features	Performance Evaluation
YOLOv1	2016	First version, real-time object detection, utilization a single neural network for detection	Basic real-time object detection abilities
YOLOv2	2017	Improved accuracy, utilization of batch normalization, multi-scale predictions	Higher accuracy with multi-scale predictions
YOLOv3	2018	Using the residual blocks, multi-scale predictions, FPN	Better performance
YOLOv4	2020	CSPDarknet53 backbone, mish activation	Improved speed and accuracy
YOLOv5	2021	Self-training, transfer learning	Improved processing time
YOLOv6	2022	EfficientRep backbone, optimization of hardware, real-time identification	Efficient hardware usage, Real-time application
YOLOv7	2022	Improved small object detection, ELAN block	High processing time, Highest accuracy, highest recall rates

When traditional object detection models and other YOLO versions are evaluated, YOLOv7 is a preferable model for real-time object detection solutions since it has higher accuracy and faster speed. YOLOv7 provides several advantages over other methods. These are easier and better integration, a more robust loss function, accurate object detection, model training efficiency, and improved label assignment. Providing these advantages is a more robust and higher-speed network architecture (Yadav et al., 2022).

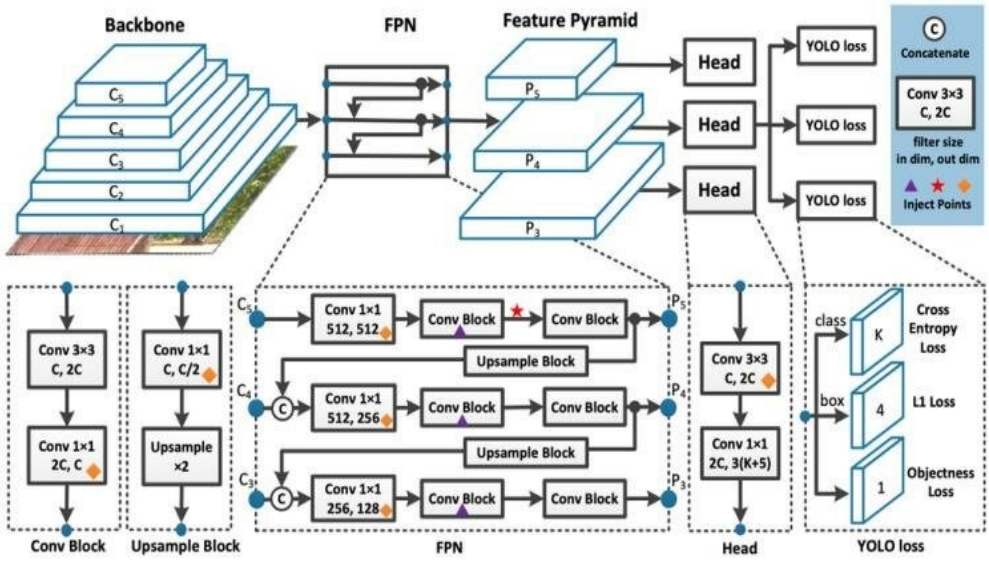

Figure 3.14. The architecture of YOLOv7 algorithm (Mostafa et al., 2023)

According to Figure 3.14, the YOLOv7 architecture uses the Extended Efficient Layer Aggregation Network (E-ELAN). According to Mostafa et al. (2023), E-ELAN has a structure that focuses on computational intensity and parameters. One of the significant advantages of "E-ELAN is that it enables deeper networks to learn and converge more effectively by controlling the gradient path" (Mostafa et al., 2023).

YOLOv7 is an algorithm with extremely important developments used for object detection. This model has the necessary architecture to resolve complex and real-time issues. Given the model's success, it is regarded as an important development for computer vision.

3.2. Object Tracking Research

In certain problems, tracking the detected object is as important as detecting the object. In this case, it is necessary to know in which direction the movement of the detected object occurs. There are many algorithms used for object tracking. One of the solutions to the problems tried to be solved with computer vision is moving object tracking. There are many methods or algorithms used for moving object tracking. With the development of deep learning, there is a growing interest in moving object tracking.

The first proposal on moving object tracking was in 2016, Bewley et al. proposed the SORT algorithm based on the Kalman filter. However, since this algorithm did not provide regular follow-up, its speed caused the algorithm to be avoided as a solution. As a solution to this situation, the Deep-SORT algorithm was proposed by Wojke et al. The developed tracking algorithm can provide long-term tracking. Furthermore, it has been noted that the identity transformation of objects decreases during the viewing period. However, since it requires deep feature extraction and causes difficulties in some problems, its preference is open to evaluation. Although there are many methods, another popular tracking algorithm among the methods used today is the Byte-Track algorithm, introduced by Zhang et al. in 2021 (Yu et al., 2023).

In general, the diagram below is applied to object detection and tracking solutions.

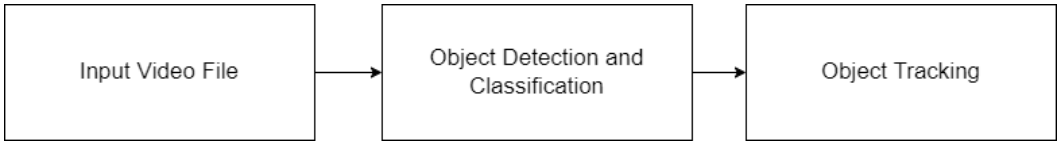

Figure 3.15. The approach of object detection and tracking solutions

3.2.1. SORT (Simple Online and Realtime Tracking) Algorithm

The Simple Online and Real-time Tracking (SORT) algorithm is the simplest and fastest tracking solution, based on the Kalman filter and making associations between bounding boxes of objects. This algorithm produces simple and rapid results by taking full advantage of the basic movements and properties of objects. In this way, it effectively carries out the tracking process of the object (Pereira et al., 2022).

The SORT algorithm makes calculations about the movement of objects iteratively through the Kalman filter (KF). In connection with this, it performs the connection between the detection and the tracking process with the Hungarian algorithm (Pereira et al., 2022).

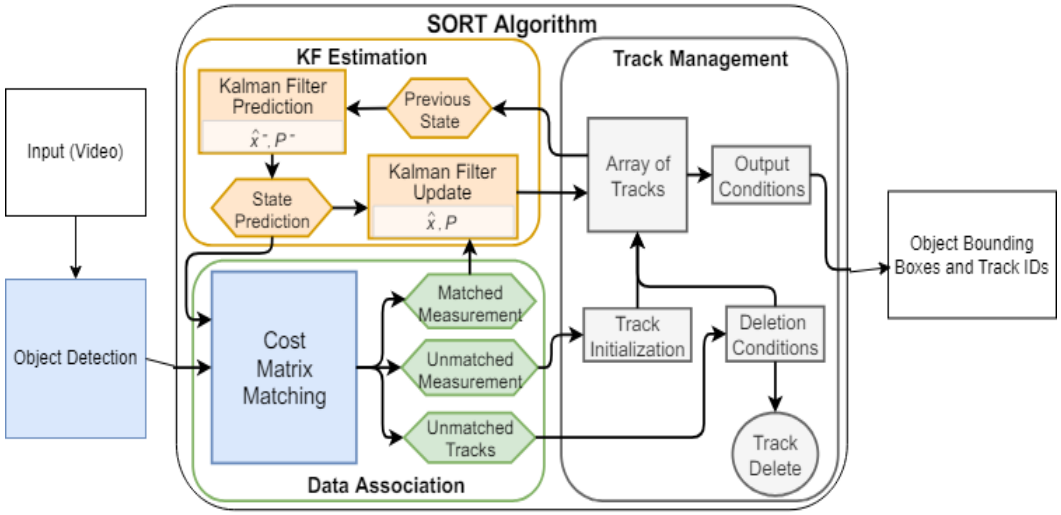

Figure 3.16. General structure of SORT algorithm (Pereira et al., 2022)

Bewley et al. (2016) describe the displacement of objects between frames using a linear constant velocity model, independent of the movements of other objects and camera movements. The situation of each target is modeled in the formula below.

$$x = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^T \quad (3.6)$$

The formula variables *u* and *v* represent the horizontal and vertical pixel location respectively, which are in the target's center. The symbols *s* and *r* denote the scale and ratio, respectively, which indicate the dimensions and ratio of the bounding area of the target. By employing the Kalman filter, the velocity components are efficiently solved, resulting in the optimal update of the target's final state. In the absence of any detection, only the linear velocity model is used and predicted without any corrections (Bewley et al., 2016).

In the data association module in the SORT algorithm, the bounding boxes in the image provided by the object detection module are matched with the bounding boxes predicted by the Kalman filter (Pereira et al., 2022). The Kalman filter module predicts the motion module for each object in the image using the linear constant velocity model. If the object cannot be associated with any tracking, the tracking state is estimated. The track management module consists of structures that enable traces to be created or deleted on the image (Pereira et al., 2022).

3.2.2. Deep-SORT Algorithm

The Deep-SORT algorithm is an advanced object tracking algorithm that was born from the SORT algorithm and emerged with different improvements. Unlike the SORT algorithm, the appearance information of objects is integrated, which thus strengthens the relationships in the image.

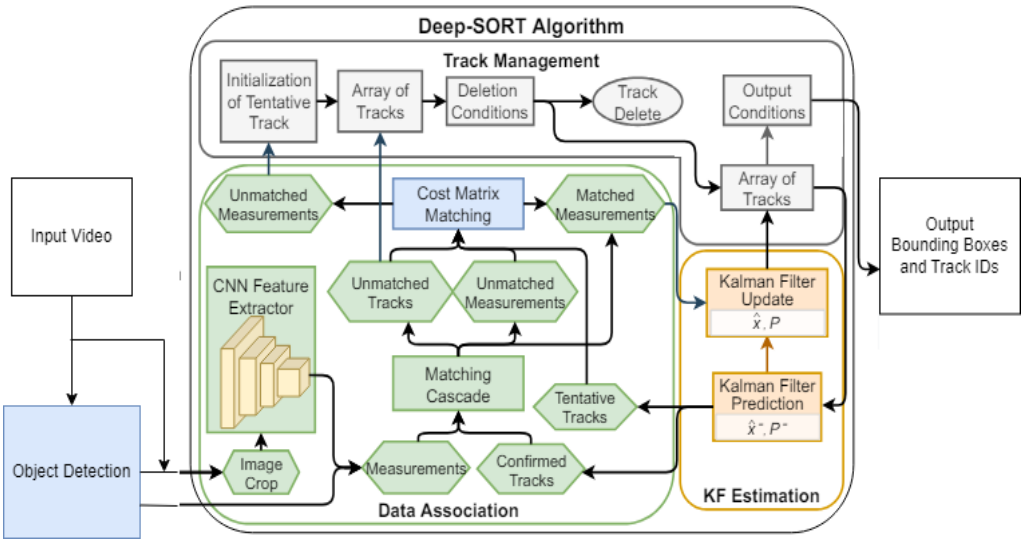

Figure 3.17. General structure of Deep-SORT algorithm (Pereira et al., 2022)

According to Pereira et al., the data association module provides view metrics based on integrated pre-trained CNNs. In this case, traces are re-identified after long-term obstruction. The Kernel Filter Prediction and Monitoring Management module has a similar structure to SORT and performs the same task as it does in the SORT algorithm. Deep-SORT has a structure similar to the working logic of the SORT algorithm. The assignment of detected boundary boxes to tracks is solved using a two-stage matching cascade. In the first phase, motion and appearance metrics are used to correlate valid tracks. In the second stage, newly created detections are associated with mismatched traces. Motion information integration is achieved by associating the predicted situations and detections using the Mahalanobis distance, and the result gives the motion information. In addition to the metrics obtained from using the Mahalanobis distance, a cosine distance measurement occurs between each trace and each measurement, which measures the distance. This creates the second metric. The features of the view are calculated with the pre-trained CNN model (Pereira et al., 2022).

According to X. Hou et al. (2019), the Mahalanobis distance, which represents spatial information and is the first distance, can be formulated as follows:

$$d^{(1)}(i,j) = \left(\mathbf{d}_j - \mathbf{y}_i\right)^T \mathbf{S}_i^{-1} (\mathbf{d}_j - \mathbf{y}_i) \tag{3.7}$$

The second distance showing the appearance information is illustrated by the formula below. While *r* in this formula represents the appearance descriptor, R_i represents the views of the last 100 objects associated with the i^{th} trace (Hou et al., 2019).

$$d^{(2)}(i,j) = \min\left(1 - \mathbf{r}_j^T \mathbf{r}_k^{(i)} \middle| \mathbf{r}_k^{(i)} \in R_i\right) \tag{3.8}$$

The integrated cost matrix is represented by the formula below (X. Hou et al., 2019).

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^2(i,j) \tag{3.9}$$

A gate matrix $b_{i,j}$ becomes 1 when both spatial and view gate functions are 1. If zero, it indicates whether (i,j) is a valid match for spatial and view. New detections are associated using this cost and gate matrix. The gate matrix is demonstrated by the formula below (Hou et al., 2019).

$$b_{i,j} = \prod_{m=1}^{2} b_{i,j}^{(m)} \tag{3.10}$$

The Deep-SORT algorithm is an algorithm based on the SORT algorithm and developed upon it, among object tracking algorithms. The Deep-SORT algorithm, which provides more reliable tracking than the SORT algorithm, can perform object tracking with better performance after the necessary hardware features are provided. The Deep-SORT algorithm, which is likely to run slower than the SORT algorithm due to its advanced structure, should be chosen based on the usage scenario and determined performance criteria.

3.3. Action/Event Detection Research

In the current problem, the interaction between pedestrians and equipment within the warehouse must be predicted. When the problem is evaluated from this perspective, certain studies have been carried out in these areas since the information determined through the images in the warehouse must be obtained. The study includes steps from general evaluation to special evaluation.

The field of computer vision is developing rapidly, influenced by the developments of deep learning techniques. Studies in this field are generally used in different areas such as object detection, autonomous vehicles, and security systems. Detecting an event or estimating an event beforehand is a very popular solution. However, since each problem has different dynamics, there are different approaches in the field of action detection. There are many methods used in this field to detect or predict a possible event. According to the conclusion drawn from the literature review, these methods are deep learning models such as Convolutional Neural Network (CNN), Region-based CNN, Fast R-CNN, Faster R-CNN, You Only Look Once (YOLO). In addition, machine learning algorithms and autoencoders can also be used in this problem by extracting the necessary features.

3.3.1. Deep Learning Methods in Action/Event Detection

Several deep learning models are commonly employed in the field of action detection. These are examples of deep learning models, specifically convolutional neural networks (CNN), Region-based convolutional neural networks (R-CNN), Fast R-CNN. For example, in their study, Ghosh, Sunny, and Roney (2019) use convolutional neural networks to detect accidents that may occur in traffic. One of the purposes of using CNN is to increase accuracy and efficiency.

- Convolutional Neural Network (CNN)

Convolutional neural networks (CNN) are an end-to-end learning model because of their architecture. This is due to the fact that it allows the parameters to be trained with the gradient descent method. The CNN model has the ability to fully learn all features of the image (Zhiqiang and Jun, 2017).

In the study of Rajesh et al. (2020), they aim to detect the moment of the accident by using convolutional neural networks and produce an alarm system at the time of the incident. According to the study of Xu et al. (2015), it is recommended that event detection of a large-scale dataset (video) be performed using convolutional neural networks using limited hardware resources. Depending on the variability of the problem, it is very common to use CNN. Generally speaking, the CNN model, consisting of convolution, pooling, and fully connected layers, is widely used in event detection solutions as well as image classification solutions.

- Region-based Convolutional Neural Network (R-CNN)

Region-based Convolutional Neural Network (R-CNN) is a deep learning method used for object detection. Its basic logic is that it processes the input image using region suggestion to detect objects in the image.

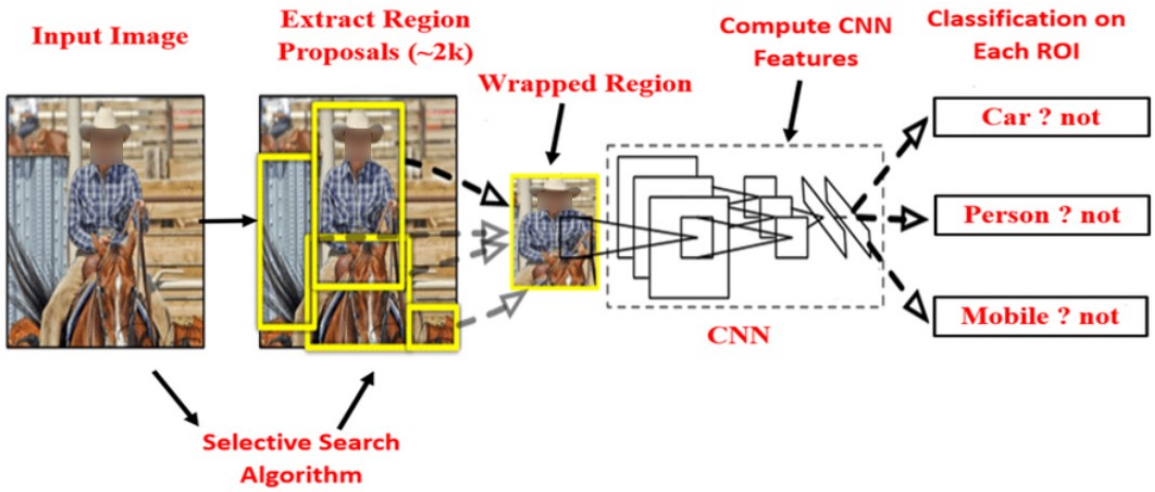

Figure 3.18. The architecture of R-CNN (Murthy et al., 2020)

The general working logic of the R-CNN model is to rescale the region proposals for each object. The regions, which are considered as a new image in themselves, are applied to the previously trained CNN model. This process is called feature extraction. After this stage, the Support Vector Machine (SVM) classifier predicts the object's existence for each region. However, it recognizes object classes. There are many disadvantages of the R-CNN model. The main disadvantage of these is that it is unsuitable for real-time applications. At the same time, the training phase of the model is also quite time-consuming (Murthy et al., 2020).

- Fast Region-based Convolutional Neural Network (Fast R-CNN)

Fast R-CNN is an object detection algorithm proposed by Girshick in 2015, as an improved version of the R-CNN model, to eliminate the disadvantages of R-CNN. The Spatial Pyramid Pooling Networks (SPPnet) method was used in fast R-CNN, which introduced a new architectural approach to eliminate the slowness in R-CNN. The task of SPPnet is to calculate the convolutional feature map for the input image. Then, by using the extracted feature map, it uses a feature vector extracted for each object and thus performs object classification on the image (Girshick, 2015).

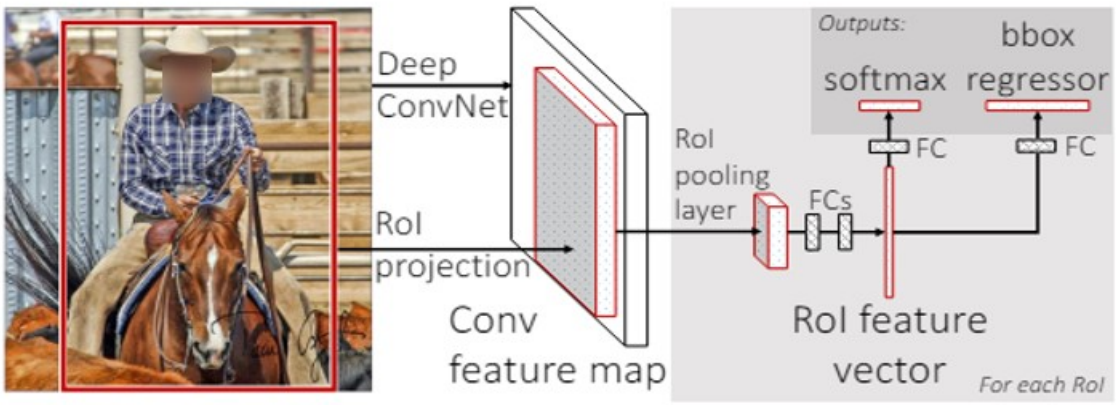

Figure 3.19. The architecture of fast R-CNN (Girshick, 2015)

With SPPnet, fast R-CNN is 10 to 100 times faster than R-CNN, and the training time is also shortened. To summarize the Fast R-CNN working architecture, first the input

image comes to the system. Then, the input image comes to ConvNet, where regions of interest (RoI) are generated. In the next stage, the RoI pooling layer is applied, and the ConvNet output is reshaped. In there, RoIs are adapted to a fixed size. Each RoI is fed by a fully connected network. This section prepares the pooled features for classification and positioning. The SoftMax layer is used to classify objects into regions. The linear regression layer is used to refine the coordinates of the bounding boxes (Hmidani & Ismaili Alaoui, 2022).

3.3.2. Action/Event Detection Methods with Normal Data

As with deep learning, the utilization of machine learning techniques in computer vision has grown. In this case, it is also possible to utilize machine learning in the scenario of action detection. It provides an approach to the problem as a classification and definition problem. Machine learning forms the basic structure of artificial intelligence. Machine learning aims to solve data problems using an algorithm that is suitable for the specific structure of the problem. What is important in machine learning is choosing the right model based on the type of algorithm, number of variables, and type of problem (Mahesh, 2020).

- Supervised Learning

Supervised learning is a technique in machine learning that learns the input-output relationship by using pre-existing data and past data. Labeled data is needed to use supervised learning algorithms (Mahesh, 2020).

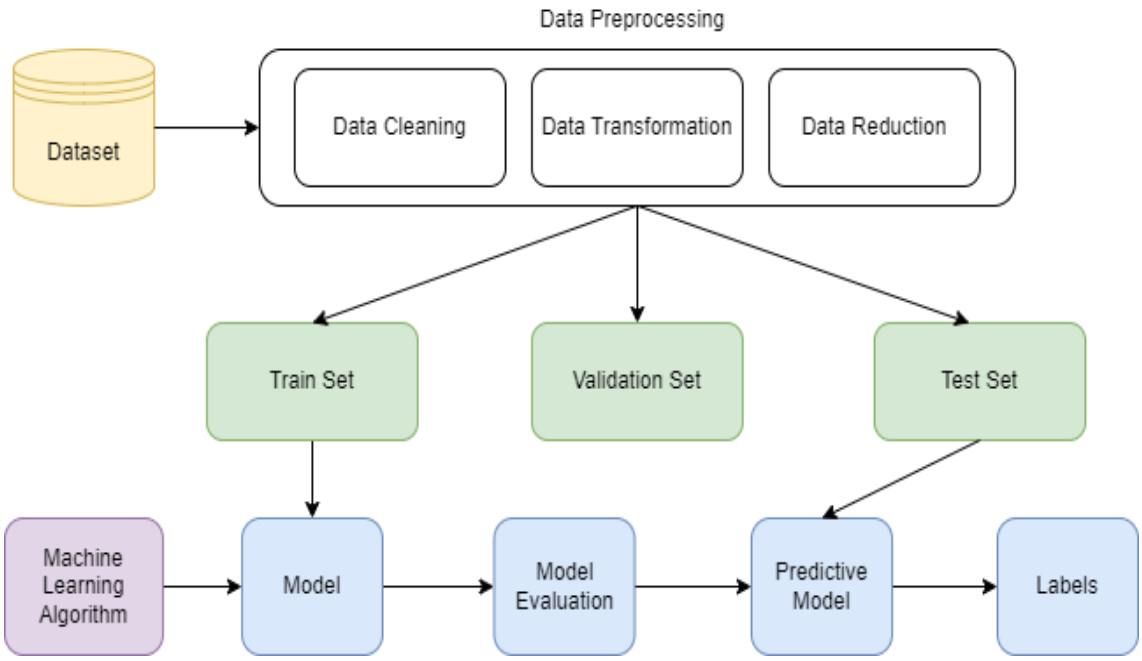

Figure 3.20. Supervised learning workflow

Before utilizing supervised learning methods, the data undergoes preliminary preparations. During the model training process, the dataset is partitioned into three distinct sections: the training set, the validation set, and the test set. Model training is performed using the training set. Testing operations are executed on the learned model using test data. As the data used in this context is already labeled, the resulting output is also tagged. Support Vector Machine (SVM), decision trees, and K-Nearest Neighbors (KNN) are all instances of supervised learning models.

- Unsupervised Learning

Unsupervised learning is a learning method that does not contain any labeling, that is, operates on a data set that does not contain correct answers, and does not involve any teaching process. In unsupervised learning, the algorithms are not given any guidance and they learn the structure in the data on their own. Due to its architecture, it recognizes the data class based on the attributes of the data. These algorithms are used as unsupervised learning models by taking part in clustering and feature reduction methods (Mahesh, 2020).

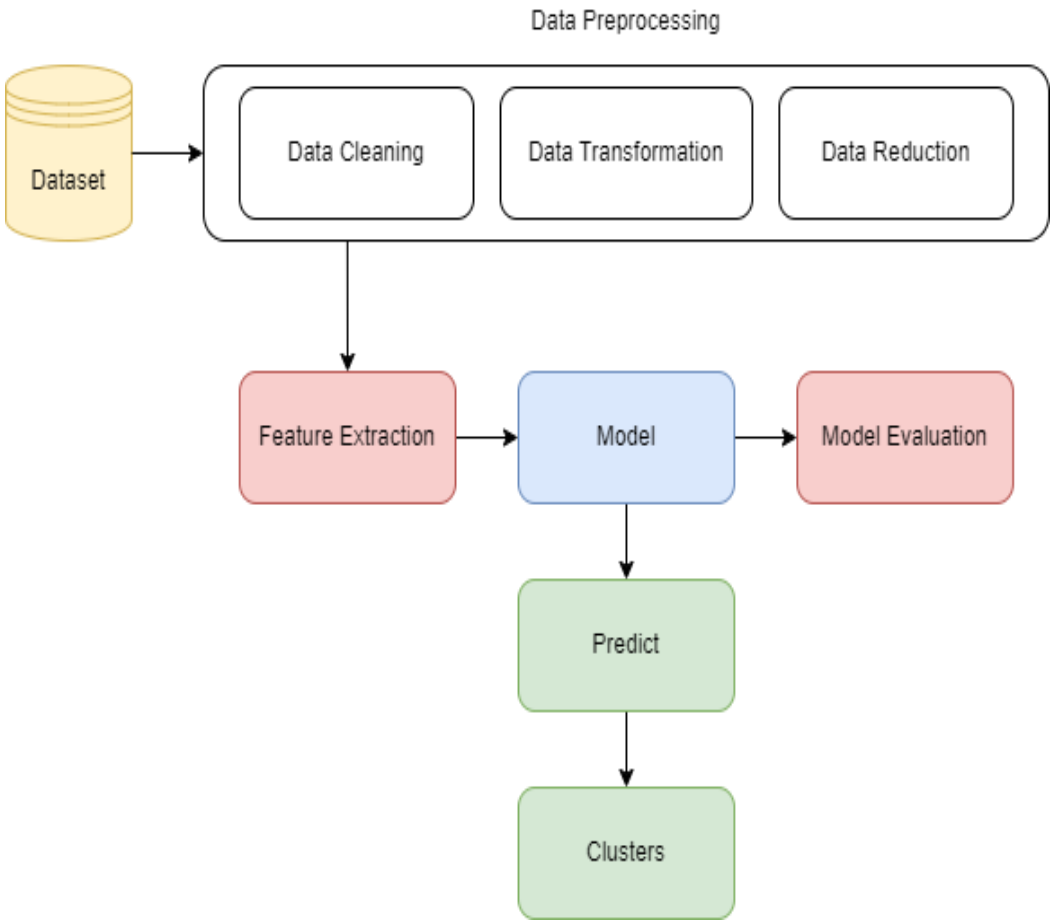

Figure 3.21. Unsupervised learning workflow

In the unsupervised learning architecture, since there is no labeling, data is used to extract features. Based on the extracted features, the model clusters the data and produces an output. Various methods are used to assess the performance of the running model. Examples of unsupervised learning algorithms include DBScan and Isolation Forest.

3.3.2.1. Autoencoders

An autoencoder is a kind of machine learning methods that is employed for unsupervised learning. Its purpose is to identify and learn the important features within a

given dataset. An autoencoder is a neural network that re-encodes data to closely match the input data. Following this procedure, it operates using the reconstruction logic.

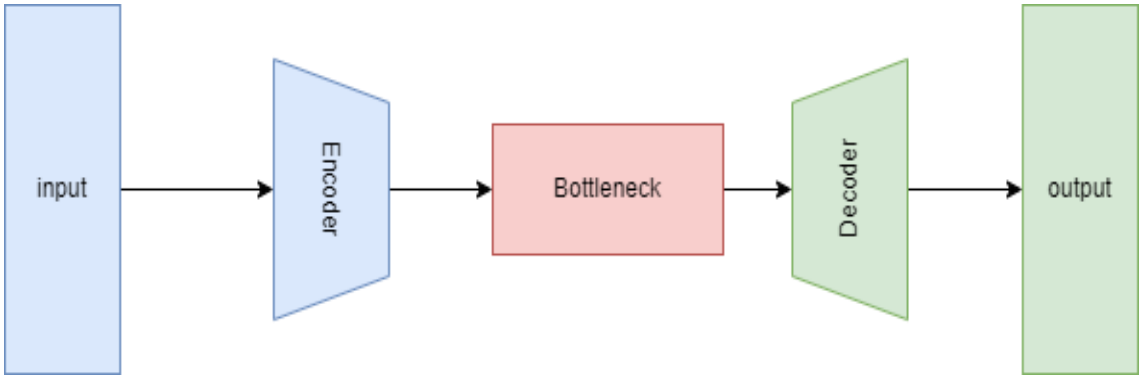

Figure 3.22. General structure of autoencoder

According to the working architecture of an autoencoder, the input is reconstructed in two stages (Mac et al., 2018). In the working logic of the encoder, **x** takes the original input. The hidden layer maps this to *H* with the function **h** = *f*(**x**). The main purpose of the decoder is to produce the equation **x'** = *g*(**h**) that represents the reconstruction. To summarize the situation:

$$\mathbf{h} = f(\mathbf{x}) \qquad (Encoder) \tag{3.11}$$

$$\mathbf{x}' = g(\mathbf{h}) \qquad (Decoder) \tag{3.12}$$

Autoencoders are algorithms that capture data with its basic features and compress the data in the most effective way (Mac et al., 2018). To explain these formulas simply, autoencoder can be summarized as follows.

$$\mathbf{h} = \sigma(W_1\mathbf{x} + b_1) \tag{3.13}$$

$$\mathbf{x}' = \sigma(W_2\mathbf{h} + b_2) \tag{3.14}$$

In the formula, b_i represents a bias vector, while σ represents an activation function. $W_i \in \Re^{I\mathrm{di}\mathrm{x}O\mathrm{di}}$ demonstrates the parameter matrix of the i^{th} layer. Because of the construction of the autoencoder, it tries to minimize the configuration error while creating the **x** value, which denotes the input value, and the **x′** value, which is the output value. The configuration error on which it is based while trying to minimize can be calculated with two different formulas. These options are calculation with L2 norm (3.15) or calculation with Cross entropy (3.16). Below are the formulas for these calculations. (Mac et al., 2018).

$$\min \mathcal{L} = \min E(\mathbf{x}, \mathbf{x}') = \min \parallel \mathbf{x} - \mathbf{x}' \parallel \tag{3.15}$$

$$\mathcal{L}(\mathbf{x} - \mathbf{x}') = -\sum_{c=1}^{M} x'_c \log(x_c) \tag{3.16}$$

There are different types of autoencoders. These are Vanilla (VA), Deep (DA), and Regularized Autoencoder (RA) (LeCun et al., 2015).

- Vanilla Autoencoder (VA):

They are also known as the simplest autoencoder. The single hidden layer H is a neural network and is less dimensional than the input layer. This autoencoder uses Adam optimization and the mean square loss function and learns to reconstruct the input (Mac et al., 2018).

- Deep Autoencoder (DA):

It is an extended version of Vanilla, similar to the original version, but with three hidden levels that are fully integrated. Deep autoencoders are trained with limited training examples. These autoencoders can improve classification performance by capturing more abstract representations from data (Mac et al., 2018).

- Regularized Autoencoder (RA):

Regularized autoencoders have the ability to reconfigure the input. In addition, these autoencoders also have a noise-canceling feature. Additionally, spare autoencoders have a sparsity penalty. This penalty is expressed as Ω(H). The denoising autoencoder

replaces these cases to generalize them if the cost function's reconstruction error term is not found in the training dataset (Mac et al., 2018).

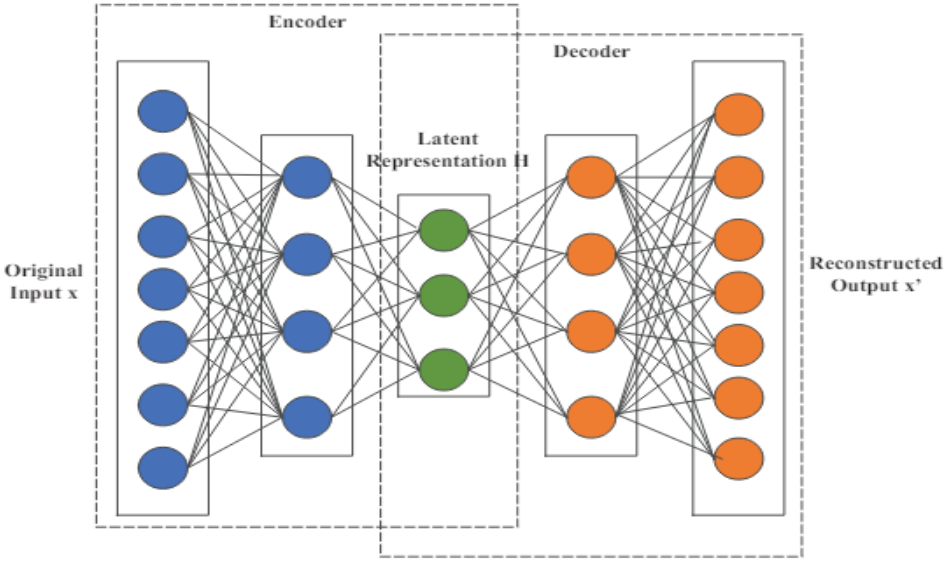

Figure 3.23. Autoencoder architecture (Mac et al., 2018)

Autoencoders are a powerful and flexible neural network model. Their main purpose is to learn data representation and minimize reconstruction errors. Autoencoders can be easily used in different problem areas with different data sets. Its application is widespread in the anomaly detection field.

There are many numerous parameters in autoencoders. The selection of these parameters during training is of great importance. The Grid Search method, which is frequently used in studies in this field, ensures that the model works with the most successful parameters.

- Grid Search Hyperparameter Tuning:

Grid Search is among the traditional methods used for hyperparameter optimization. It performs a complete search within a specific subset of the hyperparameter space of the trained model. It is evaluated by trying all possible combinations within the specified hyperparameter range. The goal is to identify the best hyperparameters for

optimizing the performance of the model. Since there is a detailed parameter search, the probability of finding the most optimal value is quite high. Grid Search has certain advantages because it is easy to implement and understand. In addition, from the perspective of computational cost, it is known as a disadvantageous method due to the large number of hyperparameters and the search for parameters within wide ranges. Additionally, when evaluated on complex datasets, the duration of the study may be long. This is known as a disadvantageous method in terms of being time-consuming (Liashchynskyi and Liashchynskyi 2019).

3.4. Model Evaluation Metrics

There are certain metrics to evaluate the success of a model given to training. While these are expressed with certain nomenclature, some of these expressions can also be evaluated by visualization. First, there are the specific outputs of a model: true positive, false positive, true negative, and false negative. Correctly classified examples include true positives (TP) and true negatives (TN). Misclassified examples include false positives (FP) and false negatives (FN) (Vujovic, 2021).

Methods frequently used to evaluate model errors include mean absolute error (MAE) and root mean squared error (RMSE). The mean absolute error, E_{MAE}, is described as the average of the absolute values of all samples in the test set. It can be stated using the formula (3.17) below (Vujovic, 2021).

$$E_{MAE} = \frac{1}{n}\sum_{j=1}^{n}\left|P_{ij} - \sum_{j=1}^{n}\left|P_{ij} - T_j\right|\right| \tag{3.17}$$

Commonly used error evaluations include root mean squared error (RMSE). In calculating this error value, the square root of the relative square error is taken, and the error is reduced to the same dimensions as the estimated dimensions. It can be expressed with the formula (3.18) below (Vujovic, 2021).

$$E_{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(P_{ij} - T_j\right)^2} \tag{3.18}$$

Confusion matrix is another evaluation metric. As an example, if the binary classifier is given, real values are marked as 1 if true, 0 if false, while positives are marked as 1 and negatives are marked as 0. There, evaluations are performed using TP, TN, FP, and FN values. The situation where the probability value is the same as the real value is expressed as TP, the situation when a positive outcome is predicted but actually a negative situation occurs is expressed as FP value, the situation when a negative situation is predicted but actually a positive situation is expressed as FN, and the situation when a negative situation is predicted and the result is actually negative is expressed as TN (Vujovic, 2021).

The model's success is also evaluated using its accuracy value. The ratio of TP and TN values to all values is the way in which it is expressed (Vujovic, 2021).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.19}$$

Evaluation metrics include precision and recall values. Precision (3.20) and recall (3.21) values are formulated as follows (Vujovic, 2021).

$$Precision = \frac{TP}{TP+FP} \tag{3.20}$$

$$Recall = \frac{TP}{TP+FN} \tag{3.21}$$

F-score is defined as the evaluation metric that expresses the model's achievement in the test dataset. It is calculated with precision and recall values (Vujovic, 2021).

$$F\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.22}$$

The most frequently used metrics to assess a model that has been trained are confusion matrix, accuracy score, precision, recall and F-score. The structure of the model utilized to address the issue determines the variability of these metrics.

- Evaluation Metrics of Object Tracking Models:

To evaluate object tracking models, Multi Object Tracking Accuracy (MOTA) and Multi Object Tracking Precision (MOTP) values are examined. MOTA is a metric that measures how accurately a tracking system works and is calculated with the following formula (3.23):

$$MOTA = 1 - \frac{\sum(FP+FN+IDS)}{\sum GT} \tag{3.23}$$

where FP (false positives) represents the number of falsely detected objects, FN (false negatives) represents the number of undetected objects, IDS (identity switches) represent the number of identity changes of all tracking targets, GT (ground truth) represents the total number of tracked targets (H. Liu et al. 2022). The MOTA value decreases as detection errors (FP and FN) and IDS increase. The closer this value is to 1, the more successful the system is.

MOTP is a metric that measures the object positioning accuracy of the tracking system and is calculated by the following formula (3.24):

$$MOTP = \frac{\sum t,idt,i}{\sum tct} \tag{3.24}$$

where $d_{t,i}$ is the distance (distance between the true bounding box and the predicted bounding box) of the match number i at time t, and c_t is the number of matches at time t (H. Liu et al. 2022). The positioning accuracy of detected objects is measured by MOTP,

and a lower value indicates greater performance because objects are detected closer to their actual locations. MOTP assesses the positioning accuracy of objects, whereas MOTA evaluates the overall detection performance.

CHAPTER 4

METHODOLOGY

We have developed and implemented a video surveillance system and a suitable deep learning model for object detection and tracking along with an action/event detection process and alarm classification. To summarize the general flow of the methodology:

- Object detection: The 7th version of the "You Only Look Once" model was used.
- Object Tracking: This section of the system utilizes the Deep SORT algorithm. With this algorithm, the tracking of objects has become more stable, and the operating accuracy of the system has increased.
- Action/Event Detection: After the object detection and object tracking stages on the images taken from the real-time image cameras in the warehouses, the speed, distance, motion vector, etc. determined on the images are detected. The information is predicted in the machine learning model as input. The shaded area created in the direction of movement for each equipment and the intersection rates of the security circle created around the person are evaluated with the predicted result, and the level of the alarm generated as an output is determined.
- Alarm Classification: The alarm level of the output resulting from the model is determined. Depending on the alarm level, an alarm is generated in the warehouse to attract the attention of people and equipment drivers. The alarm mechanism or type should be selected according to the physical conditions in the warehouse.

When evaluated in general, the system is expected to minimize occupational accidents caused by person-equipment interaction within the warehouse. The true negative, false negative, true positive and false positive rates in the alarms produced by the system are very important. High accuracy of the system will reduce person-equipment interactions, thus preventing work accidents.

4.1. General Structure of System

This study's approach to the problem is to detect person-equipment interaction with real-time videos taken from security cameras in the warehouse without the need for any hardware. As shown in Figure 4.1., the general flow of the system includes detecting people and equipment, tracking these objects, and predicting whether there are any abnormalities between the detected and tracked people and equipment.

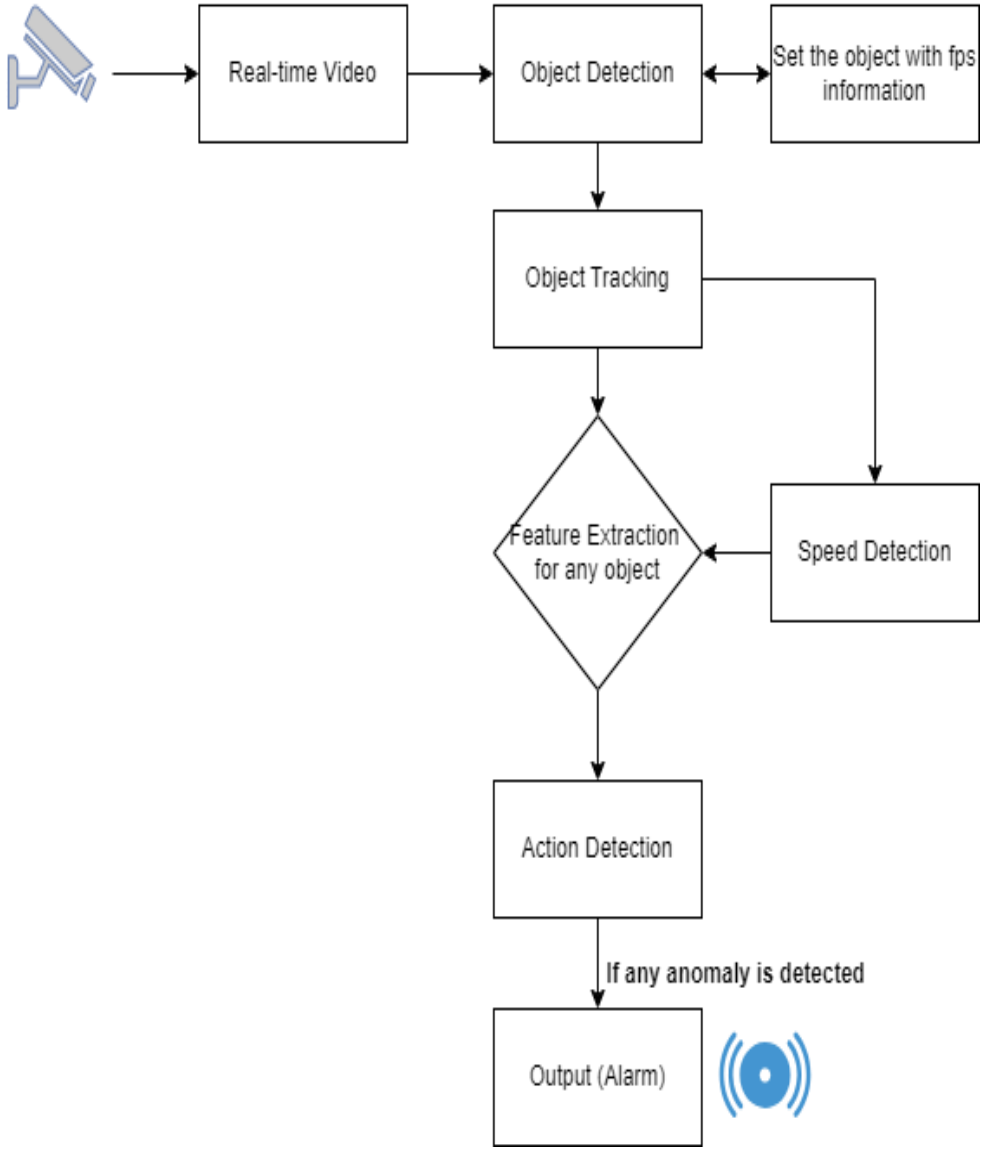

Figure 4.1. Workflow of the system

The flow chart summarizes the system as follows. The object detection part determines whether there are two different objects, person and equipment, on real-time video images taken from security cameras. Then, if the same object is detected repeatedly compared to the frame per second (fps) value of the video, the object detection process is stopped here. Following this control, the detected person or equipment is tracked. Simultaneously, the speed values of the followed person or equipment are measured. After the object detection and tracking phase, feature extraction is performed for the previously defined features. The obtained features, including the speed value, are sent to the action detection model. If any anomaly is detected in there, the output is communicated to the relevant system that an alarm should be given.

4.2. Object Detection Methodology

In the first stage of the system, it is necessary to detect objects in real-time videos. After reviewing the literature studies on object detection, the method to be used and appropriate inputs for this method must be provided. It is extremely important to detect whether objects are person or equipment in the warehouse. For this stage, it is necessary to choose the most suitable model. In this study, the YOLOv7 model is preferred for object detection. YOLOv7 is one of the latest versions of the YOLO series. Due to its architecture, the YOLOv7 model significantly increases accuracy and speed compared to other models in the YOLO series (Yang et al., 2015).

4.2.1. Dataset and Dataset Preprocessing

To train the YOLOv7 model, person-equipment in the warehouse must be viewed from every perspective. The dataset was created by combining free stock photos and images in the warehouse. First, different data preprocessing processes are performed on the obtained images. Then, labeling is performed for the resulting dataset. To prevent the over-fitting issue in a deep learning model like YOLOv7, learning must be conducted

with a substantial quantity of data (Zheng et al., 2022). Data augmentation has been made for images created from real-time video and stock photos.

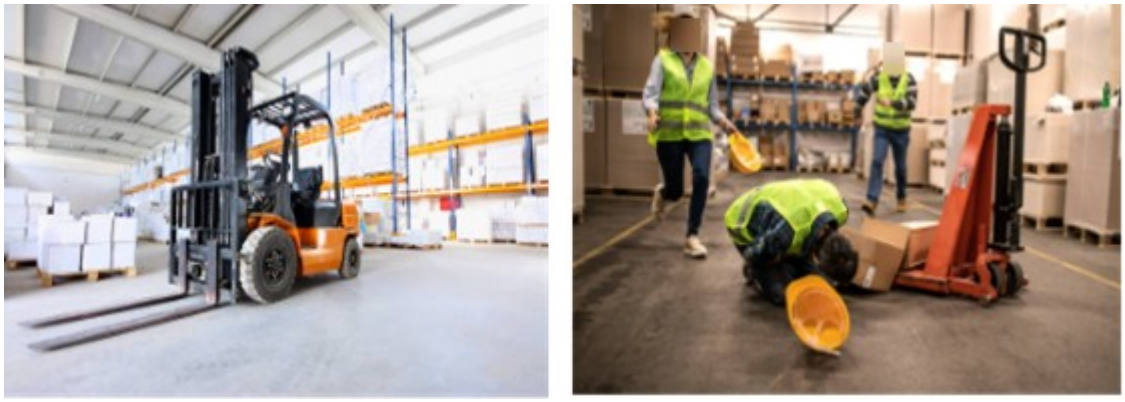

Figure 4.2. Example of data in warehouses

Operations such as gray scale and angle rotation are performed on the created dataset. Thus, overfitting is avoided. Because the number of data has increased. In addition, another problem observed in the warehouse is that the images lack a light environment in certain regions. For this reason, images taken from dark areas of the warehouse are also added to the dataset. The created dataset is divided into two categories: training and validation data.

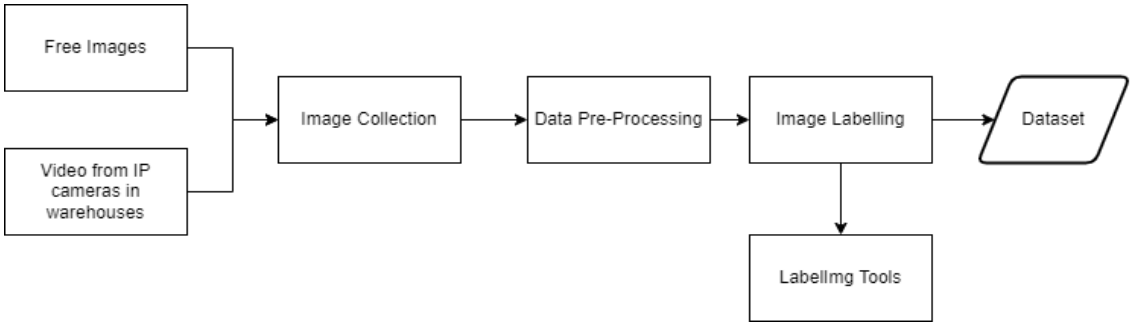

Figure 4.3. Structure of creating dataset

LabelImg, an open-source software, was used to label the created dataset. Figure 4.4 illustrates that the images have two separate names: person and equipment. After the person and equipment on the image were affected, the classifications and their coordinates in the image were recorded in the txt format file of each image (HumanSignal, n.d.).

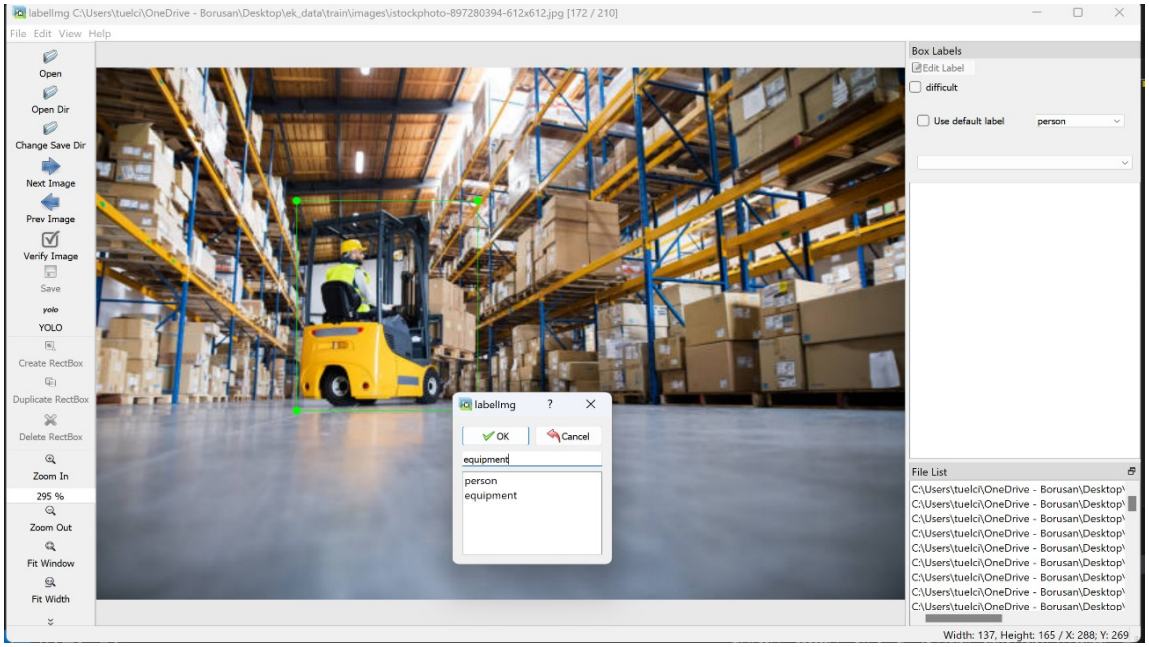

Figure 4.4. Using the LabelImg tool (HumanSignal, n.d.)

4.2.2. Model Training

The process of training the model was conducted using YOLOv7 with the dataset created with person and equipment labels. The architecture of the YOLOv7 model used is given in Figure 4.5. The YOLOv7 model has been extended with E-ELAN. According to Wang et al. (2022), E-ELAN uses expansion, merging and mixing cardinality to increase the network's capacity for learning without disrupting the original gradient path. E-ELAN used in the YOLOv7 model does not allow any changes to the architecture at

the transition layer. Nevertheless, it has the ability to modify the structure within the calculation block.

The first layer of the YOLOv7 model is a convolutional layer. In this layer, the input image undergoes processing, and low-level features are extracted. This layer is succeeded by a down sampling layer. The objective of this layer is to decrease the spatial resolution of the feature map while simultaneously increasing the quantity of feature maps. After this layer, it contains an up sampling layer. The purpose of this layer is to decrease the number of feature maps and enhance the spatial resolution of the feature map. The layers are implemented using transposed convolution layers, which up sample the input feature map by inserting zeros between its parts and then convolve the resulting tensor with a set of filters (Ye et al., 2023).

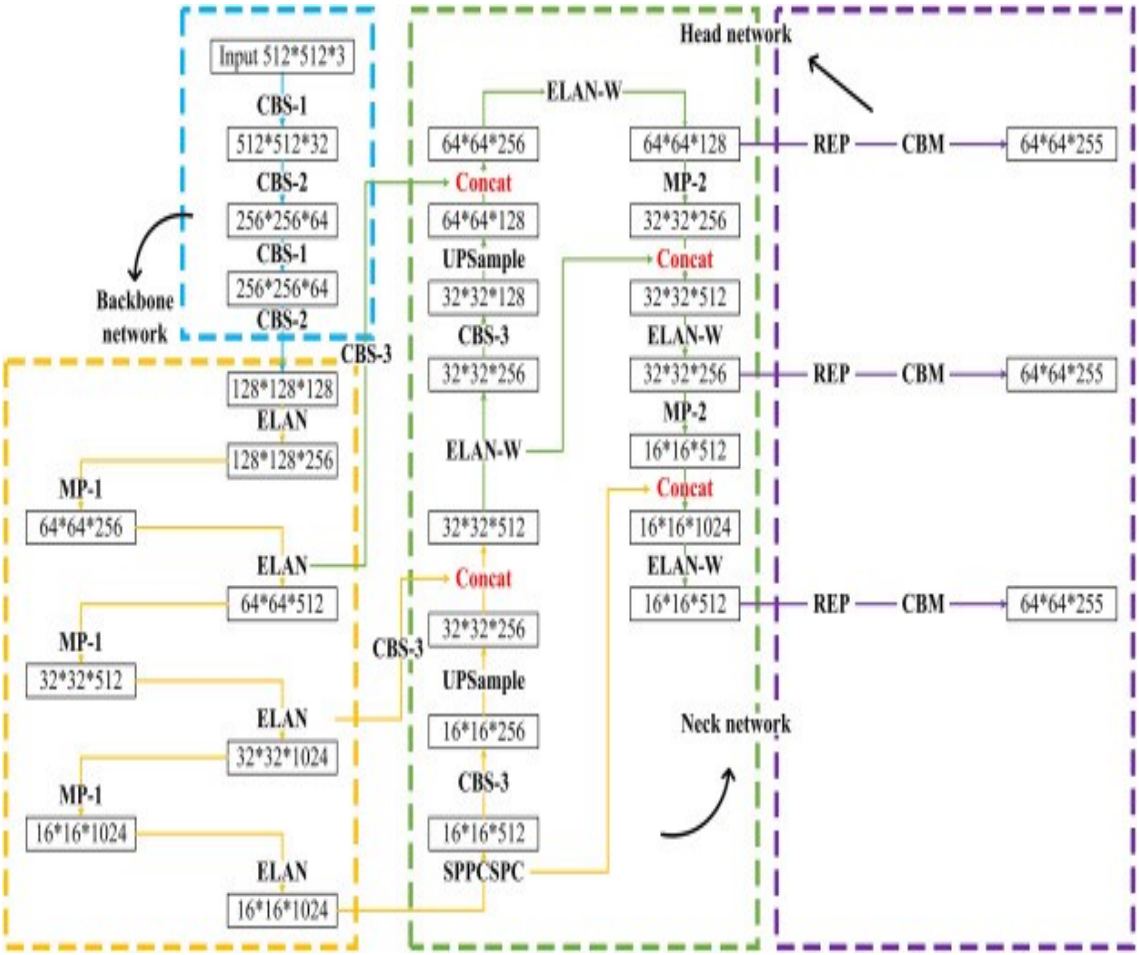

Figure 4.5. YOLOv7 model architecture (Ye et al., 2023)

It includes jump connections in the YOLOv7 architecture. These jump connections combine features from different layers of the network and increase the accuracy of object detection. The output of one layer is combined with the input of another layer to form a jump connection. Generally, this flow occurs from the down sampling layer to the up sampling layer (Ye et al., 2023).

It includes anchor boxes, which are predefined boundary boxes in the YOLOv7 architecture and are created to detect objects in the input image. The anchor box is utilized to approximate the position and size of objects in the input image and is learned by the network during training. There is also a classification and regression layer in the YOLOv7 layer. This layer is used to estimate category labels and bounding box coordinates for each anchor box (Ye et al., 2023).

The batch size, which reflects the number of samples utilized in each training iteration, is set to eight. The number of epochs gives the number of data set rounds to be passed throughout the training process. In the study, the epochs' number was determined to be 100. The input image size is determined to be 640x640. At the same time, some additional parameters are used in model training. Critically important learning parameters are learning rate, momentum, and weight decay parameters. The learning rate determines how quickly the model's weights are updated. If this parameter is high, the model will be unstable, while if it is low, the training time will be longer. In the study, the learning rate was determined to be 0.01. Momentum is the value that provides acceleration in updating the learning rate and is determined as 0.937. The weight decay parameter is used to prevent overfitting problems. In this research, it was calculated to be 0.0005. There are numerous parameters like this. Before training the model, these parameters should be optimized and updated to address the problem.

The YOLOv7 model offers numerous advantages. The model combines quick item detection and great accuracy. YOLOv7's architecture is optimized to save weight and computational costs. The model employs a combined scaling approach. There is an E-ELAN design that can boost the network's continuous learning capacity. YOLOv7 can now reparametrize thanks to RepConv (Reparametrized Convolution) layers.

4.3. Object Tracking Methodology

One of the most crucial aspects of the study after the object detection phase is to monitor the movement of the detected object. It is critical to monitor the movement of the detected person and equipment and to know the possible interaction with the result. Deep SORT is defined as a tracking technique that enables objects to be tracked in computer vision systems and gives a unique identity for each object while tracking.

Deep SORT is derived from the SORT algorithm (Simple Online Real-Time Tracking) algorithm. In the algorithm using a simple Kalman Filter, the calculation of data correlation for each frame is measured by the Hungarian method. The identity problem in the SORT algorithm is solved in the Deep-SORT algorithm. Based on the logic of the SORT algorithm, targets belonging to objects that do not match between frames are deleted. This raises the identity problem. The Deep-SORT algorithm includes a deep learning method. This transformed the algorithm into an advanced version of the SORT algorithm. Additionally, with the integration of deep learning, the identity key is eliminated, which improves tracking (Yadav et al., 2022). It is involved as a result of improving the missing components in the SORT algorithm.

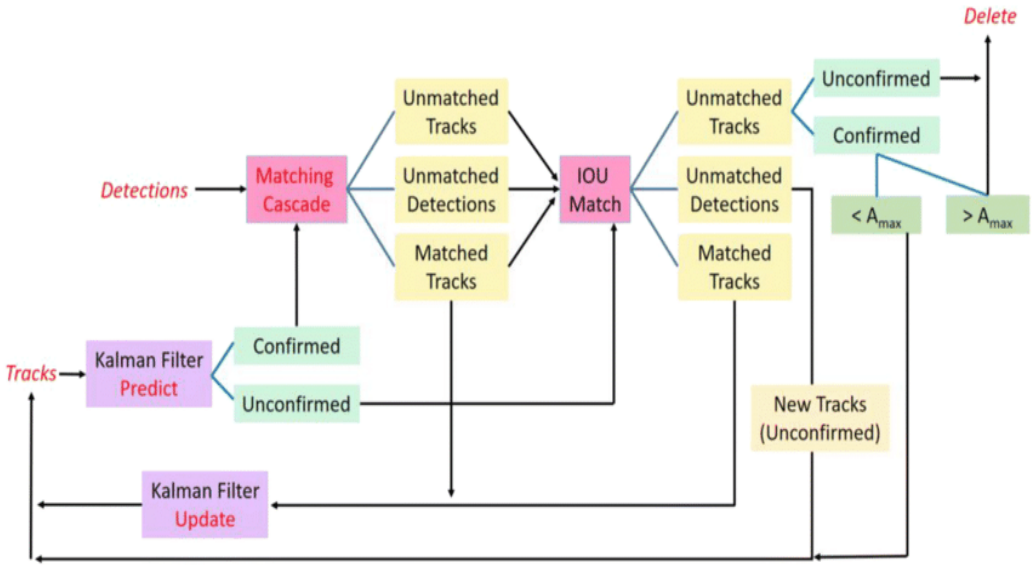

Figure 4.6. Process diagram of Deep-SORT algorithm (Chang et al., 2022)

Figure 4.6 also shows the tracking process of DeepSORT algorithm. First, detections enter into the "Matching Cascade" process and unmatched and matched tracking data are separated. Matches are made between tracking and detection data using IOU (Intersection over Union) metric. Kalman Filter updates and validates tracking data, thus increasing the accuracy in tracking. Unmatched tracking data is marked as "Unconfirmed" and those falling below a certain threshold (A_{max}) are deleted.

Kalman Filter is used in Deep-SORT (Deep Simple Online and Real-time Tracking) architecture. Kalman Filter expresses the state of each object using eight variables. It represents the bounding box centers (u,v). The aspect ratio is represented by a, while the image height is represented by h. The variables u', v', a', h' express the individual speeds of the bounding box centers, aspect ratio and image height. The purpose of the Kalman filter is to make appropriate predictions for boundary boxes. The Kalman filter allows the noise generated during detection to be considered (Yadav et al., 2022).

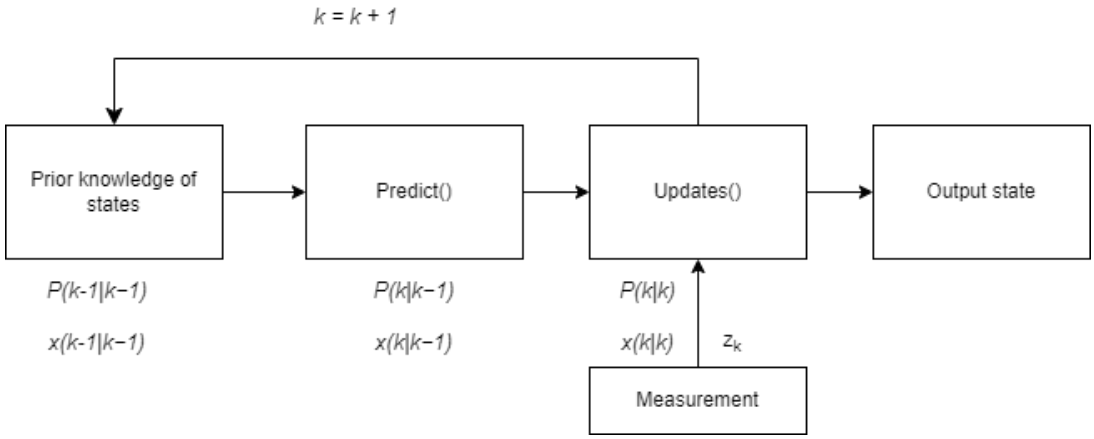

Figure 4.7. Workflow of Kalman Filter (Tithi et al., 2020)

Kalman filter works in two stages: prediction and update stage. The estimations for the current state variables are given as $x(k|k-1)$, and the uncertainties of these estimates are expressed as $P(k|k-1)$. In the update phase, the outcome of the subsequent measurement is observed. These results are expressed with z_k. The estimations are revised by employing a weighted average $(x(k|k), P(k|k))$ (Figure 4.7).

The Hungarian algorithm in the architecture of the Deep SORT algorithm is used together with the Kalman Filter to match objects with traces. This algorithm ensures data relationship between frames by establishing a relationship metric. Calculates the overlaps in the resulting metric boundary boxes. These matches, which work in image space, are supported by the Kalman filter (Yadav et al., 2022).

The area in the architecture of the Deep SORT algorithm that represents the last stage is IoU matching. IoU calculates the proportion of overlap between two bounding boxes and ensures correct matches are made. IoU matching reduces large variations that can be caused by variations or partial overlaps between images (Yadav et al., 2022).

The ReID (Re-Identification) model enables extracting the appearance features of each object in the Deep-SORT architecture. The ReID model, which is used to eliminate the identity switching problem experienced in the SORT algorithm, increases tracking accuracy (Yadav et al., 2022).

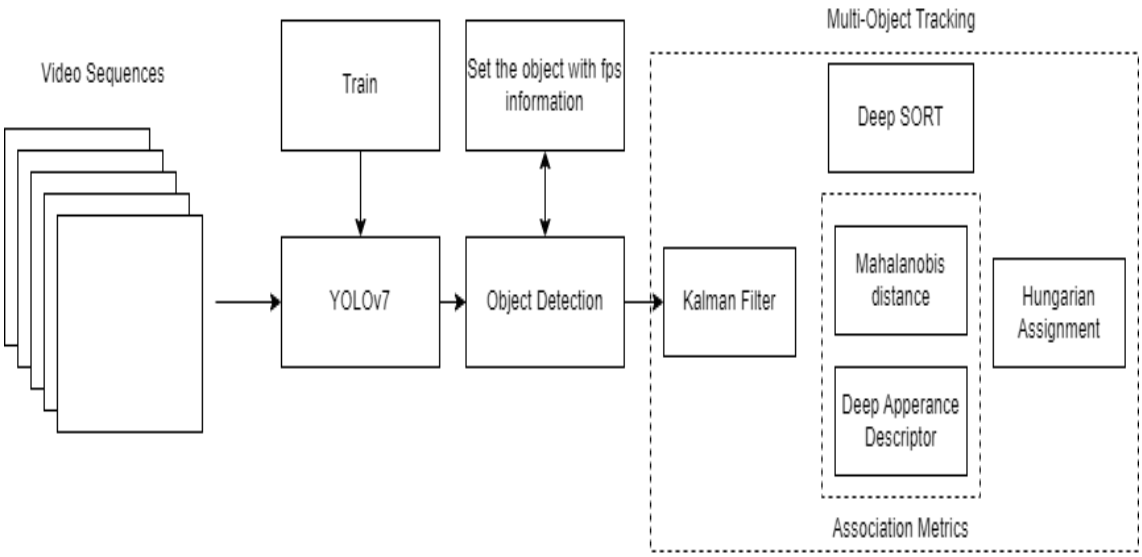

Figure 4.8. Combine of object detection and object tracking system

After the object detection phase, the remaining part of the flow diagram is as shown in the dashed box in Figure 4.8. In the object tracking part of the study, the Deep SORT algorithm was utilized. This is due to the fact that Deep SORT is more reliable and

has higher accuracy than the SORT algorithm. Additionally, its stability is among the reasons why it is preferred. The use of the ReID model and IoU matching in the system minimizes the identity problem and ensures that the system operates accurately and fluently in real-time.

4.4. Action/Event Detection Methodology

In the action detection part of the study, the model trained with information about person-equipment is expected to detect the event in case of a possible anomaly. It is planned to generate an alarm within the warehouse for the perceived potential accident risk. When the system is activated in real life, accidents caused by person-equipment interaction are expected to decrease. The implementation of this system will take place in an actual warehouse. In the field of action detection, certain features are needed to train the model. For this, there is a feature extraction section in the action detection area. Afterwards, model training is performed with the created dataset in Figure 4.9.

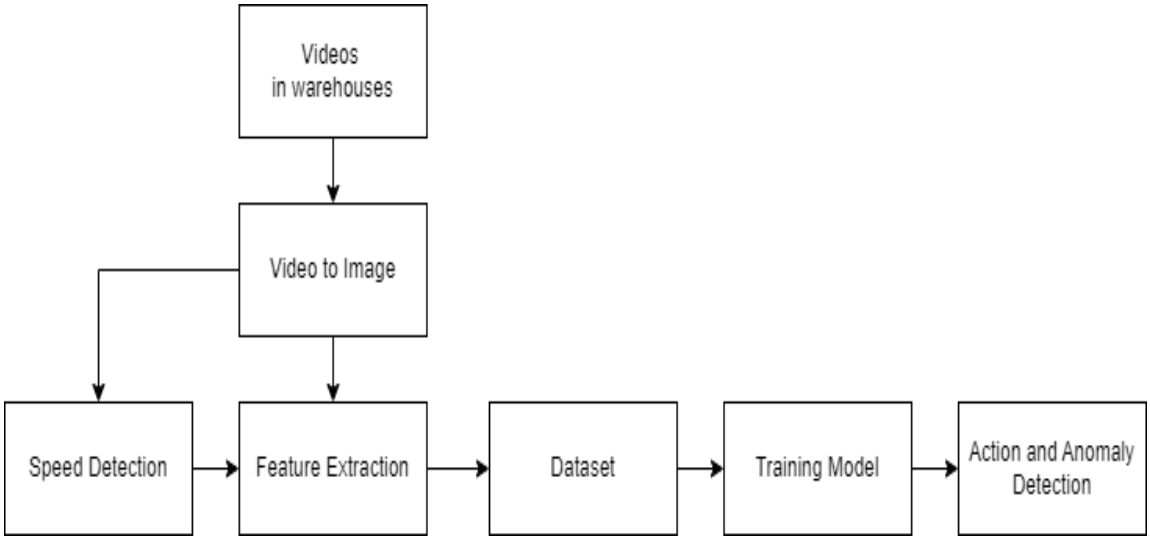

Figure 4.9. General workflow of action detection

4.4.1. Feature Extraction

By performing object detection and tracking on existing warehouse images, information such as information about whether the object is a person or equipment, the location information of the center of the object (x_c, y_c), the coordinates of the object's motion vector (x_m, y_m), the motion angle and the speed of the object are collected. This information utilize pre-trained models that have been previously employed in the domains of object detection and object tracking. After collecting information about the person and equipment in the frame of the video, a new dataset was created. The sample dataset and features are given below.

- Type of Object: Person/Equipment
- Center Point of Object: (x_c=0.32, y_c =0.57)
- Motion Vector: (x_m =3, y_m =4)
- Motion Angle: 53.2 (degree)
- Speed: 5 km/h

Among the inputs to be given to the model, object information is provided by YOLOv7, which is an object detection model. The center coordinates of the object are created using the coordinates of the bounding boxes.

$$x_c = \frac{x_1+x_2}{2}, \quad y_c = \frac{y_1+y_2}{2} \tag{4.1}$$

The motion vector information of the object is found from the difference between the positions of the object in the first-time interval by using the Deep SORT algorithm. The coordinates of the motion vector are used to determine the movement angle of the object. With this coordinate information, the $\frac{y_m}{x_m}$ ratio is calculated. Angle information is provided by taking the inverse tangent of this ratio. In this formula (4.2), *y* gives the movement of the motion vector on the y-axis, and *x* gives the component on the x-axis. The resulting angle is in radians. To convert this angle information to degrees, the value

is multiplied by $\frac{180}{\pi}$ (4.3). For speed information, the system is then fed into the speed detection section.

$$\theta = \tan^{-1}\frac{y_m}{x_m} \tag{4.2}$$

$$Motion\ Angle = \theta \times \frac{180}{\pi} \tag{4.3}$$

The information about people and equipment obtained is converted into combinations for each frame. The interaction of each person in each frame with all the equipment in the frame is calculated. For speed information, the system is fed from the speed detection section.

4.4.1.1. Speed Detection System

In the speed detection system, it is expected to detect the real-world speeds of objects detected from real-time images. There are a few different methods. However, the most significant issue that must be addressed is camera calibration. Since security cameras do not have enough image quality to detect speed due to their angle and image quality, camera calibration is extremely important. It is possible to perform camera calibration using OpenCV libraries. The main reason why straight lines appear curved is radial distortion (OpenCV: Camera Calibration, n.d.). The formulas for radial distortion are given below (4.4 and 4.5).

$$x_{distorted} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{4.4}$$

$$y_{distorted} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{4.5}$$

Additionally, tangential distortions may occur on the image. This is due to the fact that the image-taking lens is not aligned parallel to the image plane. In this case, some

areas in the image appear closer to the real image (OpenCV: Camera Calibration, n.d.). Tangential distortion equations are given below.

$$x_{distorted} = x + \ [2p_1xy + p_2(r^2 + 2x^2)] \tag{4.6}$$

$$y_{distorted} = y + \ [p_1(r^2 + 2y^2) + 2p_2xy] \tag{4.7}$$

Considering the distortion types, five coefficients are needed.

$$Distortion\ Coeffient = (k_1 \ \ k_2 \ \ p_1 \ \ p_2 \ \ k_3) \tag{4.8}$$

While performing this calibration, information about the camera is also needed. Focal length (f_x, f_y) and optical centers (c_x, c_y) contain information about the camera. Using this information, a 3x3 matrix is created.

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.9}$$

After the information about the camera and the resulting distortion is known, the camera calibration is completed by using OpenCV libraries. After camera calibration, studies were carried out to detect speed.

- 3D Transformation Speed Detection:

In this speed detection method, world coordinates (3D) are converted into image coordinates. While coordinates of a point in three-dimensional space are expressed by **X**, the image coordinates in 2D space are expressed by **Y** matrix (Hartley & Zisserman, 2004).

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ 1 \end{bmatrix} \tag{4.10}$$

$\mathbf{R}_x, \mathbf{R}_y, \mathbf{R}_z$ matrices (4.11, 4.12, 4.13) are matrices that represent transformations around the *x, y* and *z* axes. In the **R** formula, the variable r_x denotes the rotation angle around the x-axis, r_y indicates the rotation angle around the y-axis, and r_z signifies the rotation angle around the z-axis (Foley et al. 1993).

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-r_x) & \sin(-r_x) & 0 \\ 0 & -\sin(-r_x) & cos(-r_x) & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.11}$$

$$\mathbf{R}_y = \begin{bmatrix} \cos(-r_y) & 0 & -\sin(-r_y) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(-r_y) & 0 & cos(-r_y) & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.12}$$

$$\mathbf{R}_z = \begin{bmatrix} \cos(-r_z) & \sin(-r_z) & 0 & 0 \\ -\sin(-r_z) & \cos(-r_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

In conclusion, **R** is calculated with the formula below (Foley et al. 1993).

$$\mathbf{R} = \mathbf{R}_x \cdot \mathbf{R}_y \cdot \mathbf{R}_z \tag{4.14}$$

T represents the translation matrix containing the camera position, while *H* represents the height of the camera (Hartley & Zisserman, 2004).

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -H \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

P is the pixel transformation matrix. The pixel transformation matrix is employed to translate world coordinates into pixel coordinates. This transition plays a crucial role in the camera projection system. In this formula, *cx* and *cy* indicate the image's horizontal and vertical center points. These points demonstrate the camera's optical center and are often situated in the center of the image. In the actual world, *pu* represents the width of one pixel. In the real world, *pv* represents the height of one pixel.

$$\mathbf{P} = \begin{bmatrix} \frac{1}{pu} & 0 & cx \\ 0 & \frac{1}{pv} & cy \\ 0 & 0 & 1 \end{bmatrix} \tag{4.16}$$

It is necessary to convert 2D pixel coordinates to 3D world coordinates. The $\mathbf{Y}_z$ formulation (4.17) represents how pixel coordinates are transformed into image plane coordinates (Hartley & Zisserman, 2004).

$$\mathbf{Y}_z = \mathbf{P}^{-1} \cdot \mathbf{Y} \tag{4.17}$$

The **dv** represents (4.18) the direction vector that originates from the camera center and heading to the **Y** pixel point (Hartley & Zisserman, 2004).

$$\mathbf{dv} = \begin{bmatrix} \mathbf{Y}_z[0] \\ \mathbf{Y}_z[1] \\ -f \end{bmatrix} \tag{4.18}$$

Ground plane is the set of coefficients that represent the equation of the ground plane in the camera coordinate system. This plane represents the ground level (ground) plane when viewed from the point where the camera is located. This plane represents *n* normal vector and *d* constant. The given vector *n* and constant *d* are found as follows.

$\mathbf{S}_0$ represents in the ground plane prior to the transformation (4.19).

$$\mathbf{S}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.19}$$

The vector $\mathbf{N}_0$ represents the normal vector of the ground plane prior to the transformation (4.20).

$$\mathbf{N}_0 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \tag{4.20}$$

The new position of the point on the plane is determined by utilizing transformation matrices (4.21) (Hartley & Zisserman, 2004).

$$\mathbf{S}_c = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \mathbf{R} \cdot \mathbf{T} \cdot \mathbf{S}_0 \tag{4.21}$$

The new position of the normal vector is determined by utilizing transformation matrices (4.22) (Hartley & Zisserman, 2004).

$$\mathbf{N} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{R} \cdot \mathbf{T} \cdot \mathbf{N}_0 \tag{4.22}$$

The constant term, denoted as d, in the equation of a plane determines the distance of the plane from the origin (4.23).

$$d = -(as_1 + bs_2 + cs_3) \tag{4.23}$$

The intersection point between the line vector and the ground plane can be denoted as *t*, and it can be expressed as follows (Hartley & Zisserman, 2004):

$$t = \frac{-d}{n \cdot dv} \tag{4.24}$$

In the last step, calculate the world coordinates, it is represented by (4.25) and (4.26). **X** represents the world coordinates (Hartley & Zisserman, 2004).

$$\mathbf{Y}_c = \mathbf{dv} \cdot t \tag{4.25}$$

$$\mathbf{X} = (\mathbf{R} \cdot \mathbf{T})^{-1}\mathbf{Y}_c \tag{4.26}$$

After the coordinate calculation, speed calculations are made. First, the Euclidean distance formula (4.27) is employed to determine how far the object moves in one frame of time (Linguo et al. 2022).

$$distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{4.27}$$

d_{pixel} is the pixel distance the object moves in the video image. d_{meters} is the real-world movement distance of the object. The *ppm* is the conversion relationship between pixel distance and actual distance. To calculate its speed, it is necessary to know

how far the object moves in one second. In units of pixels, the speed of the vehicle in meters per second is calculated according to the conversion relationship. The constant 3.6 in the formula is the multiplier used to convert the speed of meters per second (m/s) into kilometers per hour (km/h) in speed formula (4.29). The calculation is done as follows (Linguo et al. 2022):

$$d_{meters} = \frac{d_{pixel}}{ppm} \tag{4.28}$$

$$Speed = d_{meters} * fps \ * 3.6 \tag{4.29}$$

4.4.2. Action Detection System

Model training was carried out using the features extracted to predict the interaction of person and equipment in the warehouse and to detect anomalies in movements. The model used is the Vanilla Autoencoder.

First, a dataset must be created for model training. At this stage, real video images were collected from the warehouse without anomalies. Videos that were not identified as anomalies were examined by the "Occupational Health and Environmental Safety" teams on all possible person and equipment movements in the warehouse. For each frame obtained from these videos, human position information, movement vector information, movement angle, movement direction, and speed information were combined with the equipment's position information, movement vector, movement angle, movement direction, and speed information. In summary, the combination of every person and all equipment in each frame was provided and a dataset was created. A sample dataset is provided in the Table 4.1.

Table 4.1. Example of dataset for action detection

Object1	Location_x1	Location_y1	Motion Vector1	Motion Angle1	Motion Direction1	Speed1
Person	746.0	140.0	0, 0	0	No Motion	0
Person	731.2	143.5	0, 1	90	Down	2

Object2	Location_x2	Location_y2	Motion Vector2	Motion Angle2	Motion Direction2	Speed2
Equipment	257.6	472.5	0, 0	0	No Motion	0
Equipment	372.5	960.3	0, 0	0	No Motion	0

The created dataset contains normal data. Normal data is included in the training. The dataset is partitioned into two parts: 80% training data and 20% validation data. Model training is carried out with the training data. The reconstruction error value of the model is obtained with the validation data. This error value signifies a difference between the model when it reconstructs the data according to the input data (Hung et al., 2019). The threshold determined according to the reconstruction error value helps determine whether the data is abnormal or normal. It is expressed as the average reconstruction error value, $\bar{E}$ (Mac et al., 2018).

$$\bar{E} = \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \mathbf{x'}_i\| = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{n}(x_{ij} - x'_{ij})^2\right) \tag{4.30}$$

Standard deviation is expressed by s:

$$s = \sqrt{\frac{\sum_{i=1}^{N}(E_i - \bar{E})^2}{N-1}} \tag{4.31}$$

And the threshold value is expressed by θ:

$$\theta = \bar{E} + \alpha \times s \tag{4.32}$$

N in the formulas (4.30, 4.31, and 4.32) is the number of input vectors, n is the size of the input vectors, and α is the parameter chosen depending on the classification problem (Mac et al., 2018).

Figure 4.10 also summarizes the working process of an Autoencoder model and action detection system. In the first stage, the normal dataset and real-time data are collected and passed through feature extraction and data preprocessing stages. This data with extracted features is processed using a Vanilla Autoencoder model. During the training process of the model, the reconstruction error is calculated on the validation dataset and real-time data. If the reconstruction error value does not exceed the threshold value, this data is in the normal class. However, if the reconstruction error value is greater than the threshold value, the data is classified as abnormal data. After this stage, if there is any abnormality in the data, the abnormality is graded by looking at the threshold and reconstruction error values. If the reconstruction error value is between the threshold value and two threshold values, an anomaly called an encounter occurs. If the reconstruction error value is between two threshold values and three threshold values, a near miss anomaly occurs. If the reconstruction error value is greater than three threshold values, it is classified as an emergency anomaly.

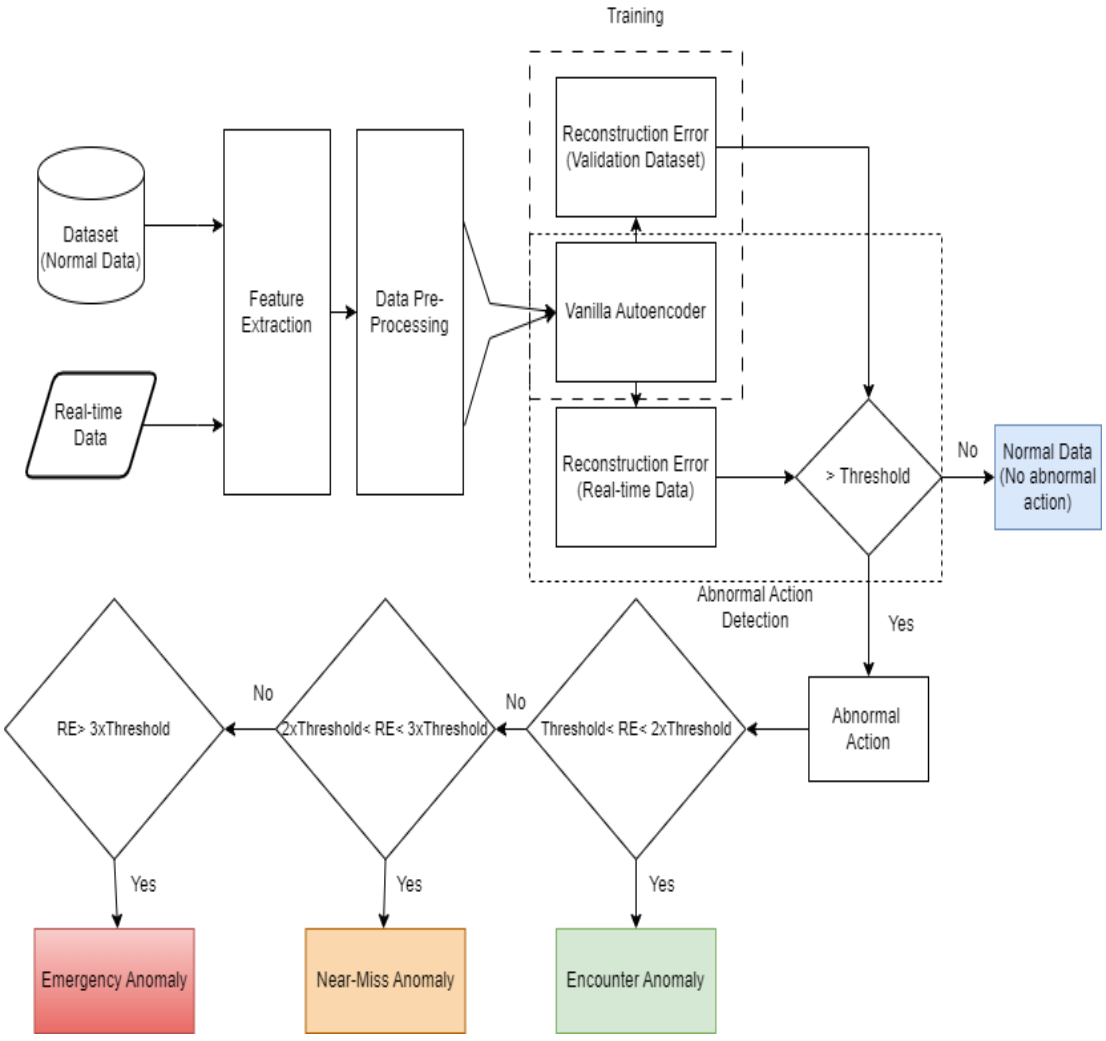

Figure 4.10. Workflow of Autoencoder model and action detection system

Parameter optimization should be done in order to get maximum performance from model training. There are certain parameters for the vanilla autoencoder. These are latent space size, learning rate, batch size, epochs, activation function, optimizer, and dropout rate. The dropout rate is optional for the vanilla autoencoder. However, since the problem is open to overfitting, it was employed in this study. The Grid Search method was utilized to find the most suitable version of certain hyperparameters for the problem and data (Figure 4.11). The best parameters are obtained by evaluating the best four models found with the Grid Search method.

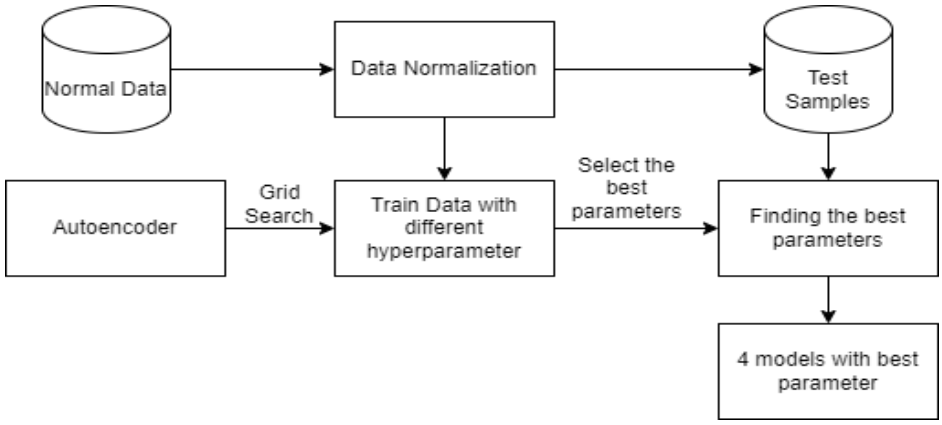

Figure 4.11. Grid Search parameter tuning

The model was created with the most optimal parameters found. After completing the training, we used the developed model to make predictions on various test sets that included both abnormal and normal data. It was determined in which frame the abnormally calculated data was an anomaly. The resulting system was made suitable for real-time operation.

4.4.3. Alarm Classification

Alarm classification is critical for the system. At this stage, giving the alarm in the right area and correctly is essential for the system to work correctly. After the action/event detection section was created in the system, tests were carried out on the previously prepared data for an accident or where an accident was possible. At this stage, the Kolmogorov-Simirnov (KS) test, one of the most widely recognized methodologies for testing normality, is performed. The KS test, which is typically utilized to determine whether a sample comes entirely from a population with a certain continuity, is used to define what type of distribution the hypothesized hypothesis actually is (Drezner, Turel, and Zerom, 2010).

The KS test is applied to the dataset consisting of images taken during or before the accident. The values considered are the reconstruction error values. In there, the

distribution was tested to be a Gaussian (normal) distribution. According to these empirical values, it was observed whether there was a Gaussian distribution. According to the KS test results, the *p*-value was determined to be 0.32. These values were found to be higher than the 0.05 significance level, indicating that the hypothesis was acceptable. In this scenario, it can be said that the reconstruction error values obtained with the test dataset are Gaussian (Normal) distribution.

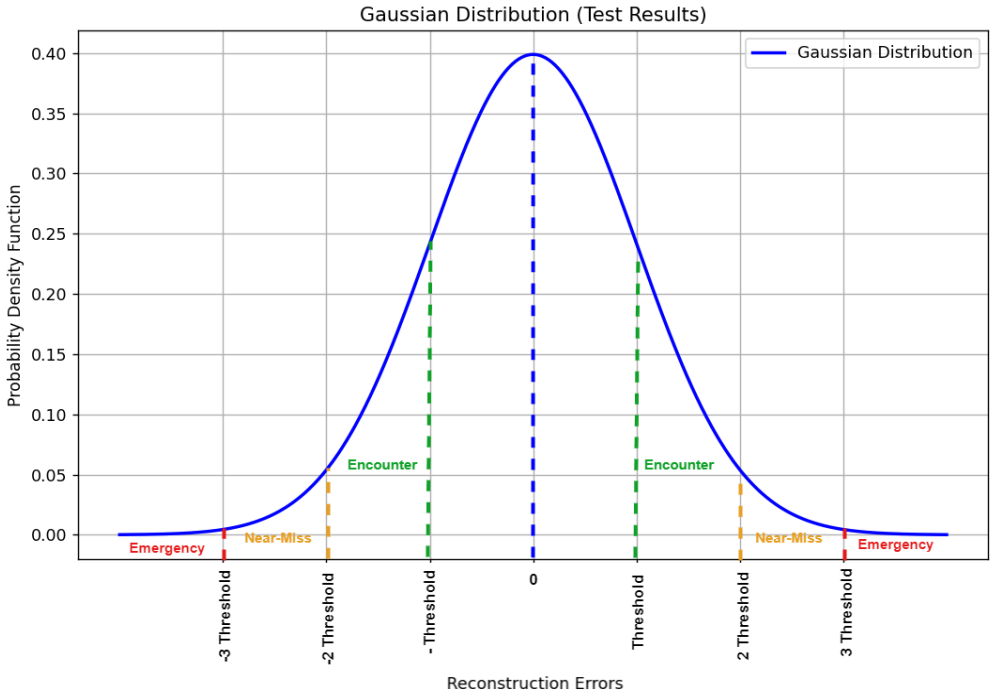

Figure 4.12. The distribution of alarm classification system

According to these results, if the restructuring error value of the input data into the system is between the threshold value and two threshold values, it is classified as an alarm encounter. If it is between two threshold values and three threshold values, it is classified as a near miss; If it is greater than three thresholds, it is classified as an emergency.

CHAPTER 5

EXPERIMENTAL STUDY AND RESULTS

The system, which aims to minimize human and equipment interaction within the warehouse, is aimed at obtaining the best results in the fields of object detection, object tracking, and action detection and ensuring that the system operates with maximum performance. Experimental studies were conducted on the system in the areas of object detection, object tracking, speed detection, the creation of the action detection model, and testing the system. As mentioned in the methodology section, the system includes object detection, object tracking, and action detection parts. Each part was evaluated separately. Separate experimental studies were conducted for each part. As a result, an overall assessment of the system was made in the action detection section, and the system's overall success was monitored.

5.1. Object Detection Studies and Results

In the object detection part of the system, object discrimination must be made in the images taken from the warehouse's security camera. In this case, first of all, the classification of objects has a significant position in the system. In experimental studies, it was thought that the issue should be resolved by determining what kind of labeling structure should be used for the images in the warehouse. The solutions presented are supported by experimental studies.

In this case, the model progressed with different labeling. A separate class is determined for each equipment, and labels include forklift, reach truck, pallet jack, etc. It was realized as follows. In Figure 5.1, the outcomes of the model are given. The model has an 84% accuracy rate in the forklift class. However, when looked at, it misclassifies between a reach truck, a wheelbarrow, and a forklift. In the pallet jack class, there is a 75% accuracy rate. However, as seen in the figure, it also appears to be classified as a

reach truck or forklift. The person classification is 95% correct. Misclassification is almost non-existent. The Reach truck label was classified correctly at a rate of 55%. This shows that the model is not successful at the desired level in the reach truck classification. The wheelbarrow tag also has 87% accuracy. In general, the model's achievement can be interpreted favorably. However, there are problems with classification, and it is predicted that it will reduce the success of the system.

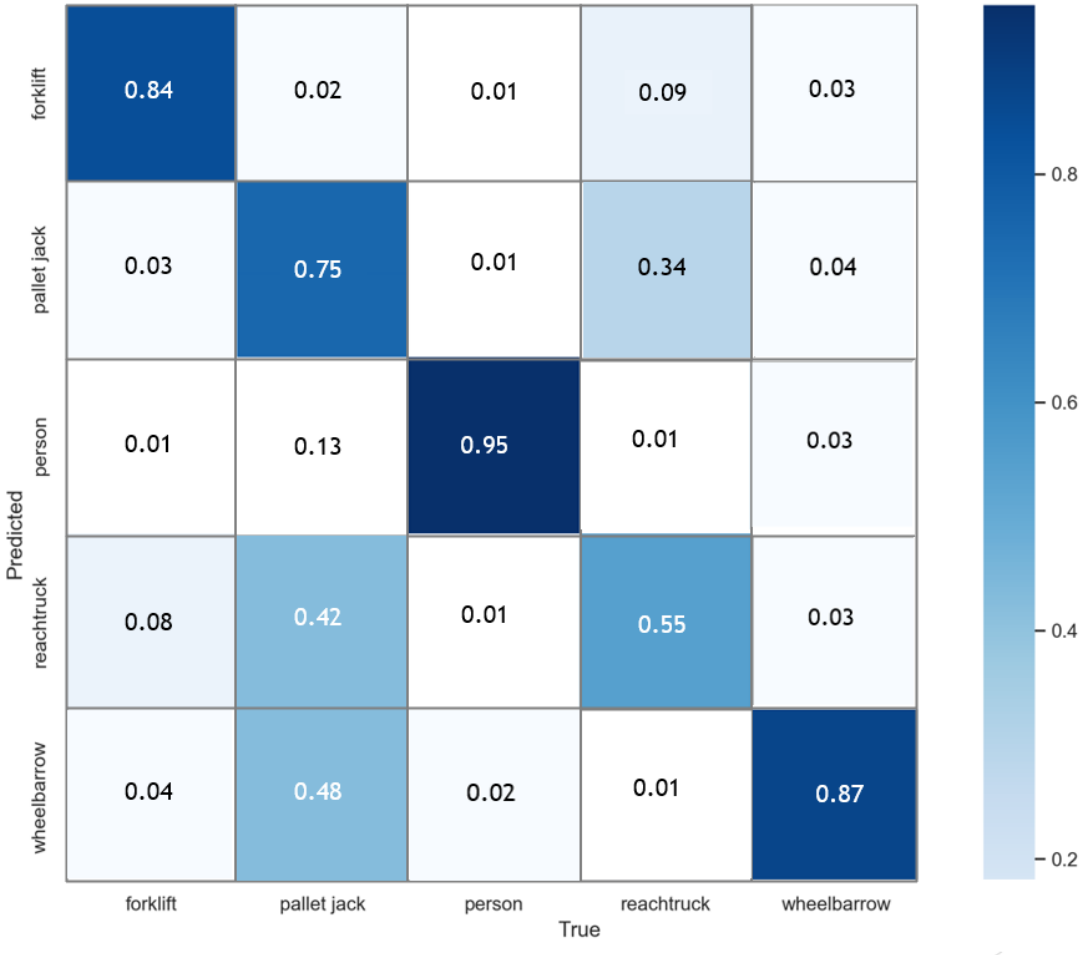

Figure 5.1. Confusion matrix with multiple labels

Consequently, it was thought that the separation of people and equipment would be sufficient in the system. This is because the interaction takes place between people and equipment, and the type of equipment does not matter at the time of the accident. In

addition, the model only needs to distinguish between individuals and equipment. The model trained with 1078 data labeled as person and equipment predicts the person class with 86% accuracy. The equipment class is predicted with 88% accuracy. Overall, the model predicts the separation of people and equipment quite accurately.

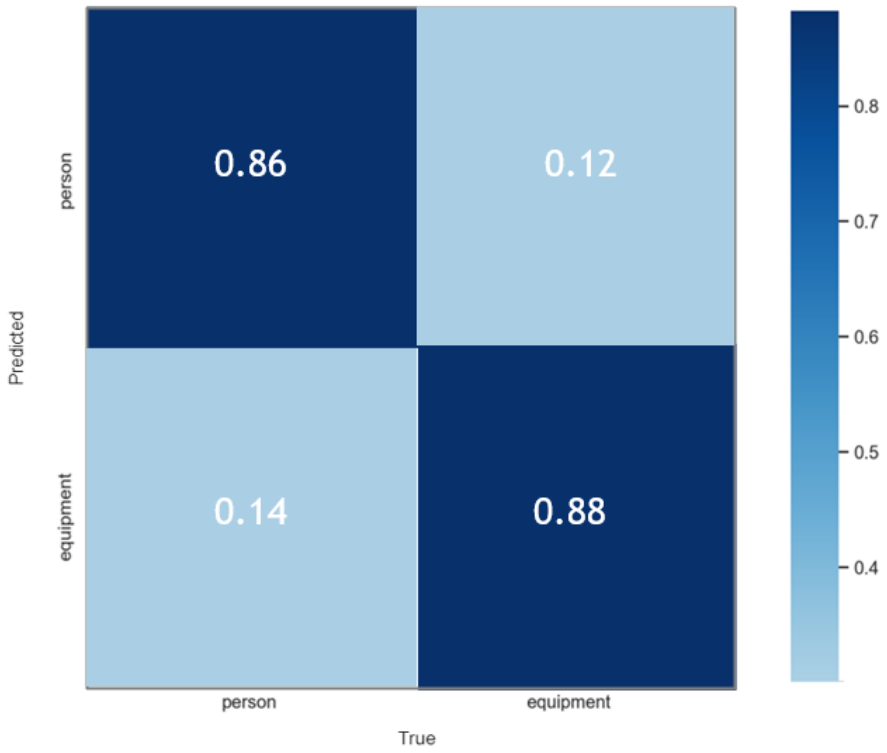

Figure 5.2. Confusion matrix with two labels

F_1 score is a criterion that balances the accuracy and sensitivity of the model. The confidence metric refers to the confidence level or probability value that the model determines for its predictions. If we need to evaluate the F_1 curve in Figure 5.3, this graph shows the F_1 score according to the confidence level of the model. It increases as the confidence level for the person class of the model increases. For confidence levels of 0.9 and above, performance degradation is observed. The average F_1 score in the equipment class of the model, within the confidence interval between 0.3 and 0.9, is around 84%. When evaluated for all classes, it was observed that the F_1 score of the model was fixed at 89% at the 0.3 confidence level. The model is successful in terms of performance when

the overall performance of the model is assessed. It is acknowledged that a balanced accuracy and sensitivity across all classifications has been achieved.

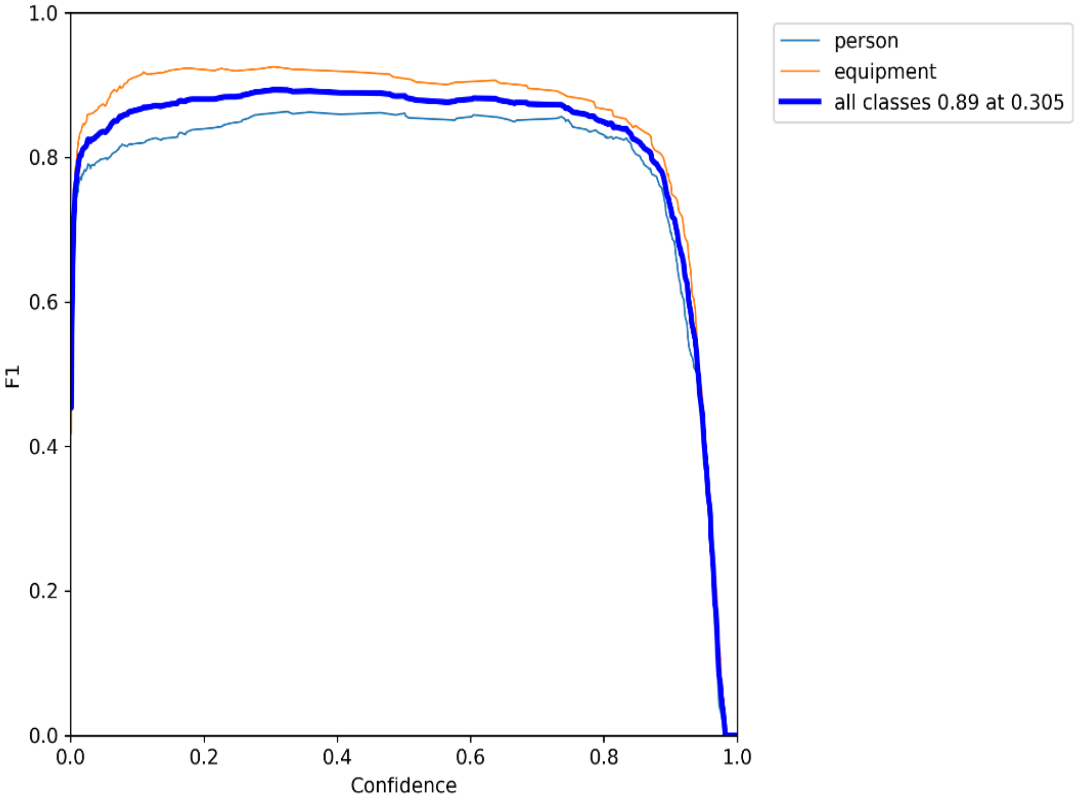

Figure 5.3. F_1 Curve

One of the important graphs used to evaluate the performance of the model is the Receiver Operating Characteristic Curve. The ROC curve is a graphical representation used to evaluate the performance of a classification model. The ROC curve shows the False Positive Rate (FPR) on the x-axis and the True Positive Rate (TPR) on the y-axis. The curve is close to the upper left corner, indicating that the model is operating with a high TPR and a low FPR, indicating good performance.

TPR and FPR values of the model are used to construct the ROC curve. These values indicate that the model has a good performance in distinguishing classes and making correct predictions. In Figure 5.4, all ROC curves show that the model has a good

performance overall and is successful in distinguishing positive classes correctly. The curves have high TPR and low FPR, indicating that the model makes low false positive and high true positive predictions.

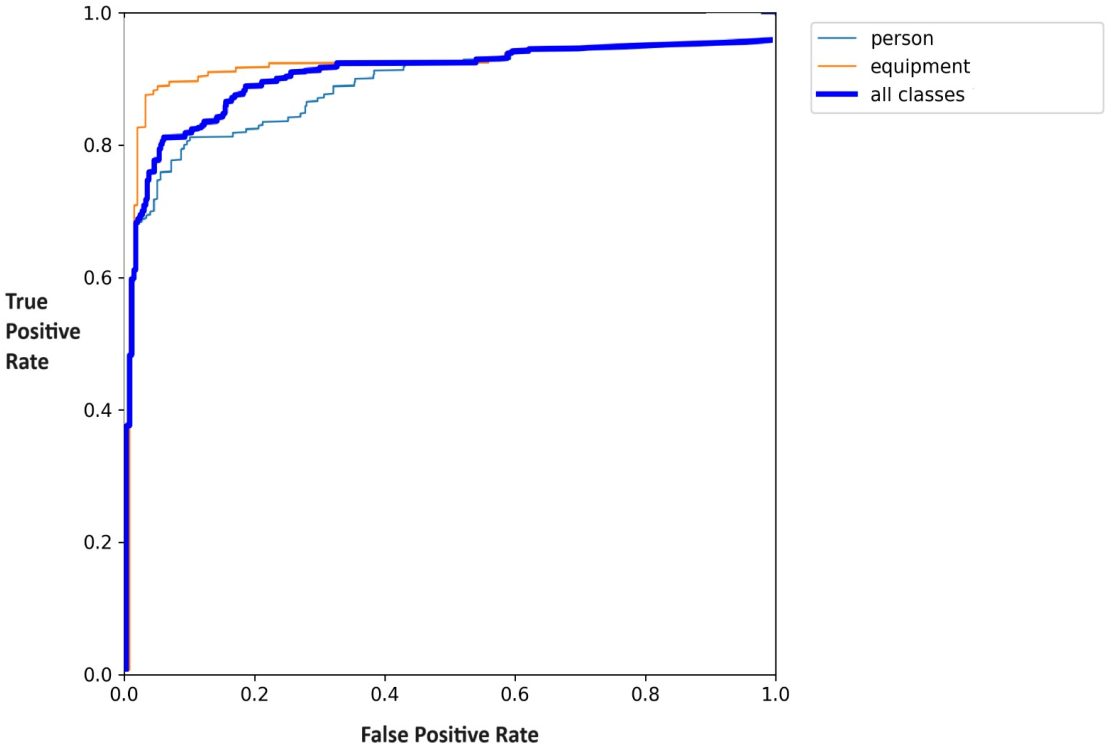

Figure 5.4. ROC Curve

When the YOLOv7 model is evaluated in general, it successfully performs person and equipment classification. The model's success is adequate to identify the interaction between them and it has performance metrics that can provide input to the action detection system.

The model successfully classified person and equipment on the given test data (Figure 5.5). With this model, it is possible to provide input for the action detection system.

Figure 5.5. Test results with our dataset

5.2. Object Tracking Studies and Results

In the object tracking part of the system, it is aimed at monitoring the detected classes and keeping information about their movements after the classified person and equipment are detected. At this stage, SORT and Deep-SORT algorithms were applied to track the person and equipment classes detected by YOLOv7. Appropriate algorithm selection was made in line with the experimental results. To evaluate the success of the algorithms utilized in the object tracking section, the Deep SORT algorithm was used by addressing certain reasons in line with the Multiple Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP) parameters and experimental results.

Follow-up was carried out after the SORT algorithm; person-equipment classes were separated. In line with the observations made and the numerical data obtained, it was seen that there were certain problems in the SORT algorithm.

Figure 5.6. The result of the SORT algorithm

To evaluate the success of the algorithms employed in the object tracking section, the Deep SORT algorithm was used by addressing certain reasons in line with the Multiple Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP) parameters and experimental results. Apart from this, it was examined in experimental evaluations made on real-time images.

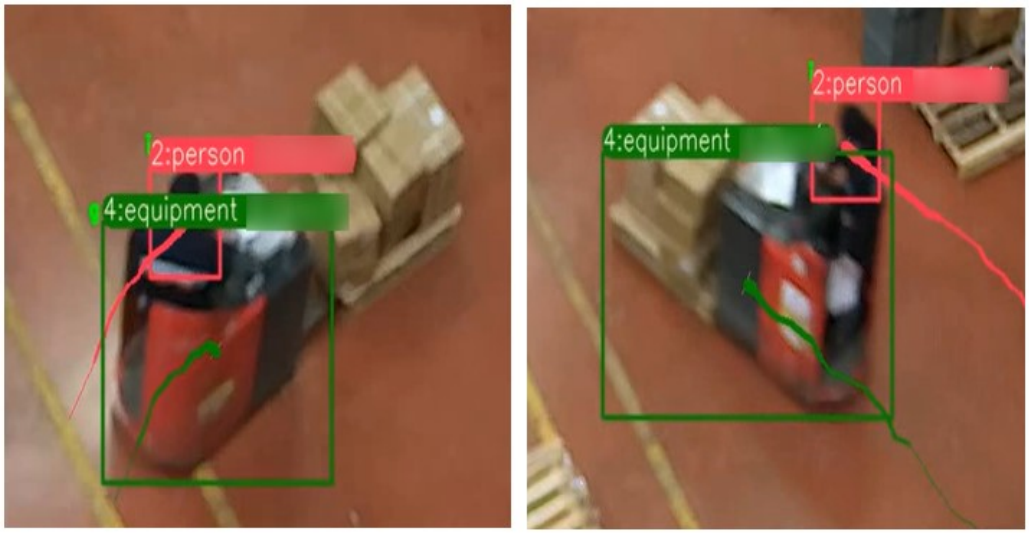

Figure 5.7. Results of the Deep SORT algorithm

In the test dataset, the MOTA value for the Deep SORT algorithm was measured at 85%. The MOTP value is 89.2%. In the study conducted on the SORT algorithm with the same test set, the MOTA value was measured at 78%. The MOTP value was measured at 81%. Considering these values, the Deep SORT algorithm was measured more successfully in the research. The number of ID changes affecting the accuracy values of tracking algorithms also shows that Deep SORT works more stable and more accurately. In addition, the measured fps value is among the important parameters. The algorithm works at 30 FPS. This shows that the algorithm can be used for real-time problems.

5.3. Action/Event Detection Studies and Results

One of the most critical components of the system to prevent possible accidents by predicting the interaction between person-equipment classifications in the warehouse is the action/event detection part. As a result of the work carried out, it is aimed at preventing possible accidents. The Vanilla Autoencoder was utilized in this section of the study. The dataset created for training the model contains completely normal data. Grid Search method was employed to select appropriate parameters for model training with this dataset. In this way, model training was carried out with the best parameters, and the success of the system was increased. The best four models were determined according to the Grid Search method. Model parameters were selected by interpreting the model structures and successes by performing an experimental study on the labeled test data determined by the models. The frame number indicating the x-axis of the graphs provides information about the person and equipment interaction in which frame the data is located.

The model's overall performance is considered successful as the reconstruction error is low and the number of data exceeding the threshold value is low, as seen in Figure 5.8. It has the ability to easily detect anomalies identified in the test data.

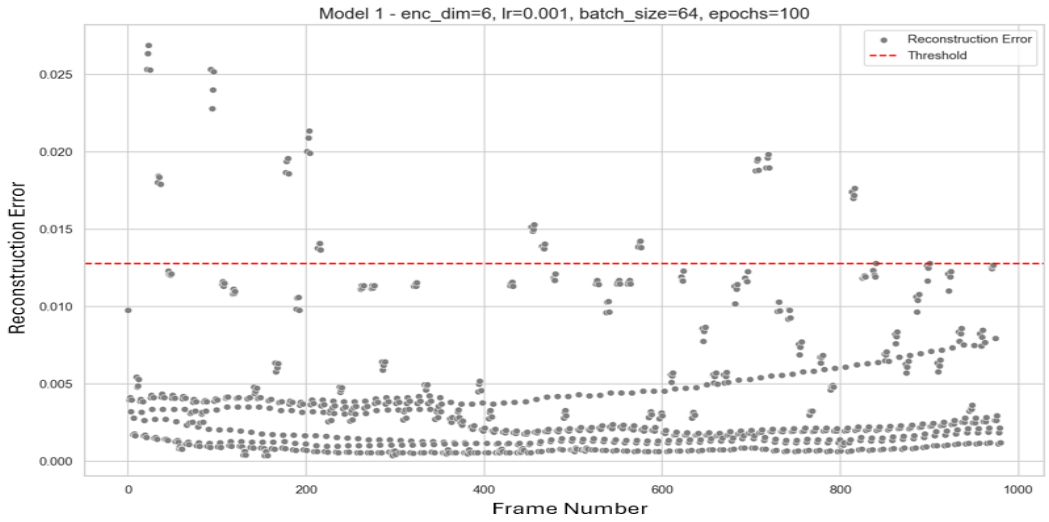

Figure 5.8. The Model 1 results with first parameters combination

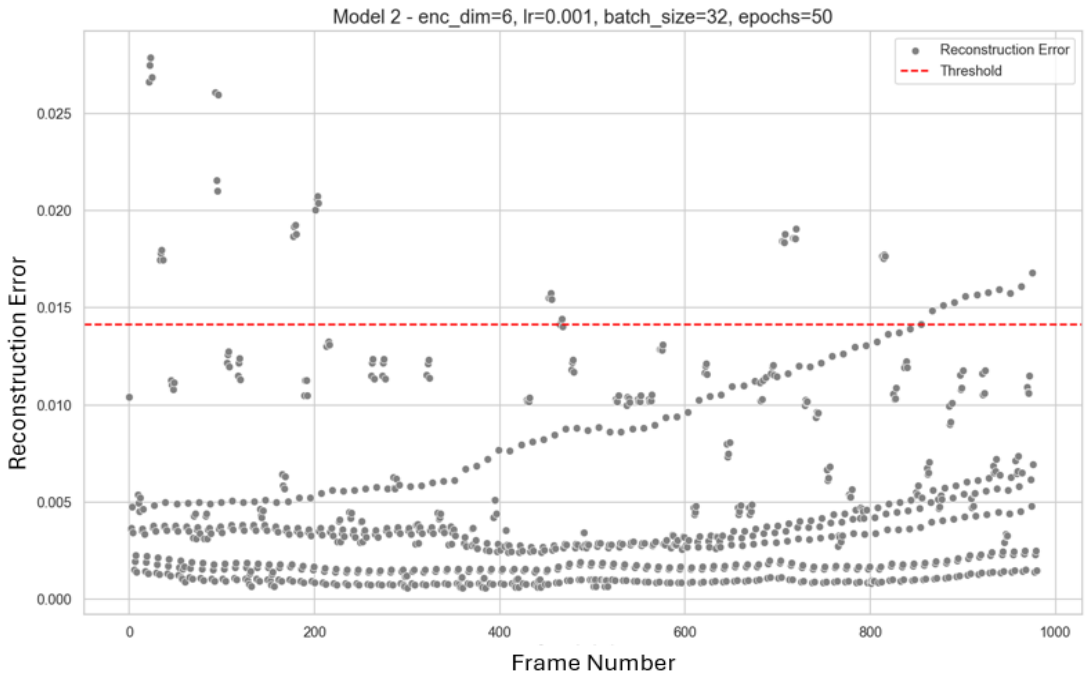

Figure 5.9. The Model 2 results with second parameters combination

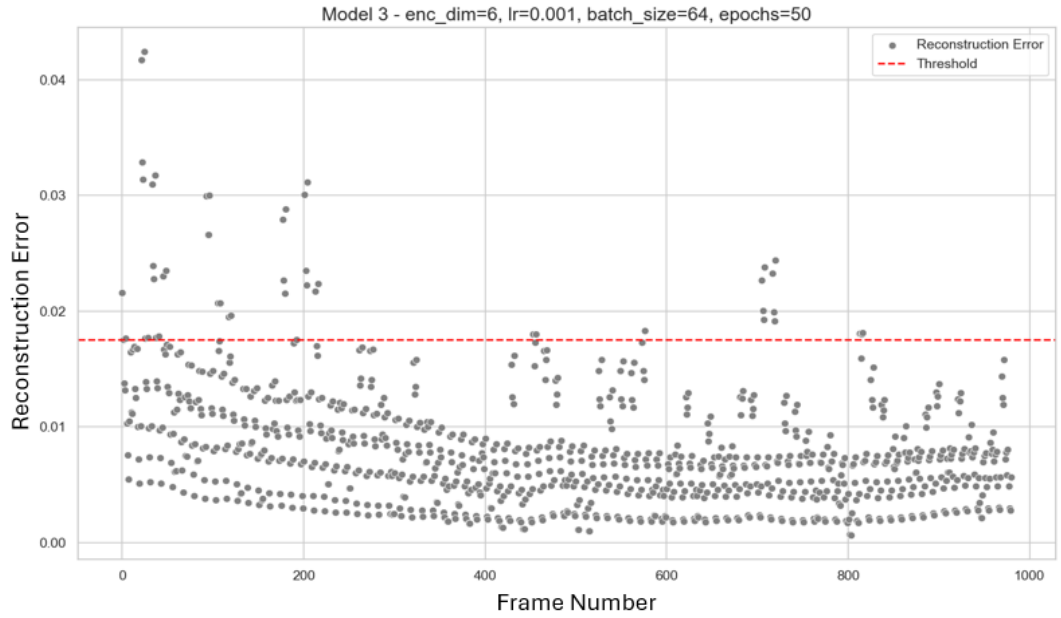

Figure 5.10. The Model 3 results with third parameters combination

When Model 2 is evaluated, it is similar to Model 1 in terms of performance. However, the numbers of false positive, false negative, true positive and true negative may vary depending on Model 1.

Similar to other models, anomaly detection is successful in Model 3 (Figure 5.10). However, the measured threshold value was found to be greater than that of other models. It has been observed that Model 4 works more similar to Model 3 (Figure 5.11). Most data were generated with lower reconstruction error values, and fewer samples were considered anomalies and exceeded the threshold value.

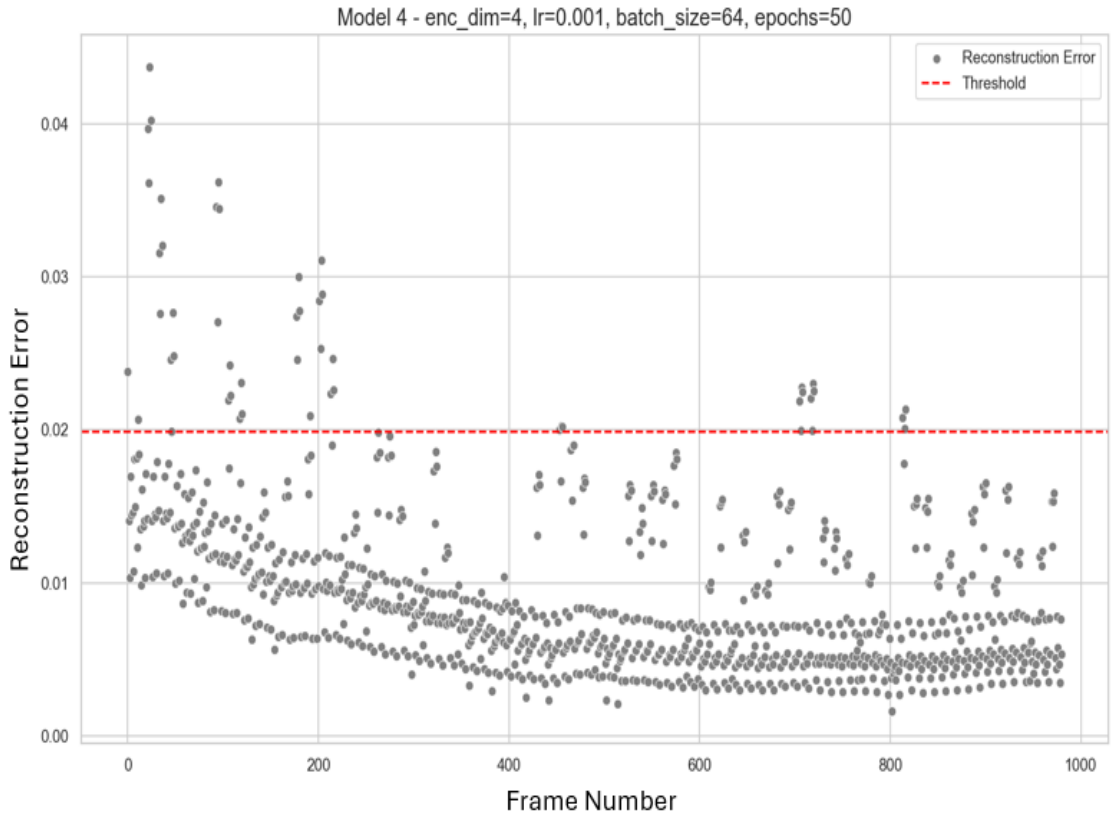

Figure 5.11. The Model 4 results with fourth parameters combination

Table 5.1. Models and parameters with results

Model	**Encoding Dimension**	**Learning Rate**	**Batch Size**	**Epochs**	**Average Reconstruction Error**	**Samples Above Threshold**
Model 1	6	0.001	64	100	0.0085	50
Model 2	6	0.001	32	50	0.0095	47
Model 3	6	0.001	64	50	0.01	45
Model 4	4	0.001	64	50	0.011	60

When all models are evaluated, the four generally selected models are considered successful in terms of performance as they have a successful reconstruction rate. Model 1 can be regarded the most successful model out of the four models as it has the lowest average reconstruction error. Model 2 has higher reconstruction error than Model 1. Based on this outcome, it may be inferred that it would exhibit strong performance. Model 3 and Model 4 have higher reconstruction errors. Considering this result, they are expected to underperform Model 1 and Model 2. When the outcomes of these studies were evaluated, it was decided to perform model training with the parameters of Model 1. After the training was completed with the selected model, testing studies were carried out on the created labeled data.

The examples in Figure 5.12 belong to the video containing the moment of collision between a forklift and a person. In order to interpret the graph produced by the model as output, the system detects the encounter in the first stage. When the reconstruction error value exceeds the first determined threshold value, the "Encounter" alarm status is observed. After this stage, the proximity between the forklift and the person increases seconds later. At this stage, the system produces a "Near Accident" class alarm. If the reconstruction error value exceeds the determined emergency threshold, real-time person and equipment interaction occurs. At this stage, the detected abnormal moments are detected and recorded with the frame and information about the person and equipment. This situation is summarized in the following situation flow (Figure 5.13).

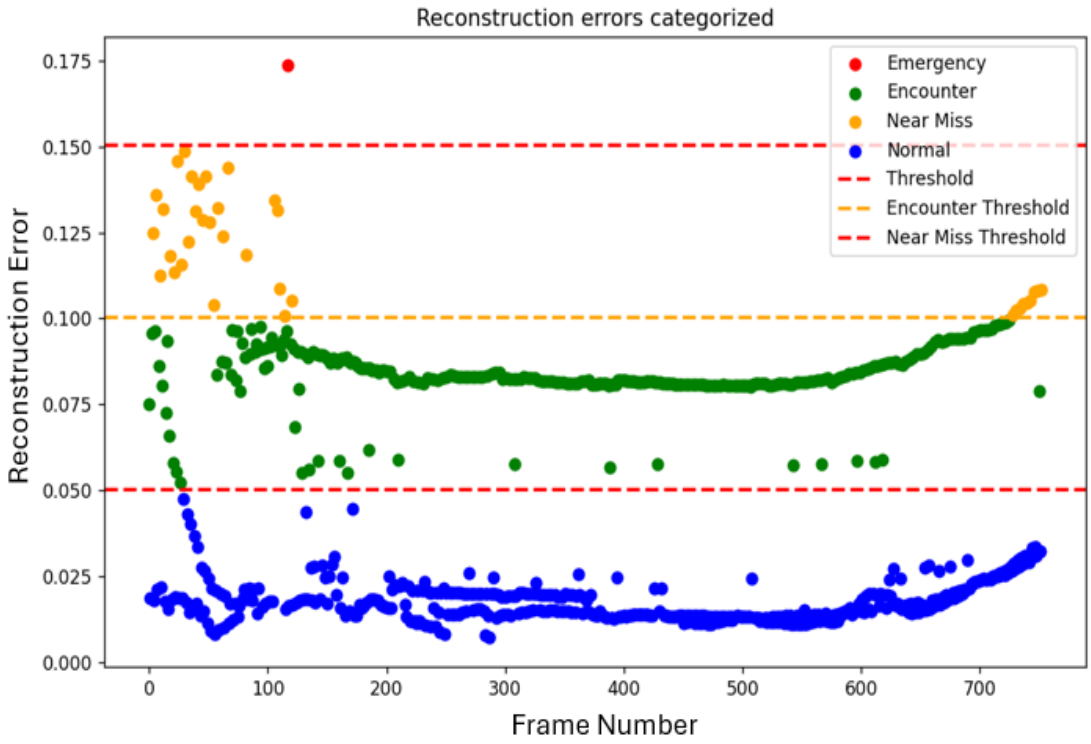

Figure 5.12. Test result in accident situation

For example, the data shown with a red dot among the people and equipment interacting in the 113th frame represents an accident moment. Here, there is interaction information for all people and equipment in a frame. There is interaction information between the same people and equipment before the 113th frame, and according to the information for these binary combinations, it also includes encounter and near-miss information for the same combination. This allows the system to generate an alarm before an accident occurs and to prevent possible accidents before they occur. Similar examples related to the system are given in the Appendix A section of the study.

Figure 5.13. Real-time test results

CHAPTER 6

CONCLUSION AND FUTURE WORKS

In the logistics industry, there is very heavy traffic in the warehouse. In addition to this situation, human behavior, etc. There is a possibility of accidents among employees for many reasons. In line with these reasons, this study aimed to prevent interaction between person-equipment in the warehouse. All studies, including the problem itself, have been tested in the field. Expert opinion was consulted regarding the approach and evaluation of the problem. In addition, real-time studies were carried out on system-related data collection, data processing, etc.

6.1. Conclusion

In the logistics industry, which grows and increases business volume every day, there is a lot of interaction between people and equipment in warehouse areas. Due to the heavy traffic in the warehouse, person-equipment interaction is quite high. "Video Surveillance System" has been developed to reduce person-equipment interaction. The system consists of three main parts. These are object detection, object tracking, and action/event detection. Although the results obtained in the object detection, object tracking, and action detection sections of the system, where the interaction of person and equipment in the warehouse is intended to be predicted, appear to be independent of each other, the success of the system until the output is gradually fed by the success of the mentioned sections.

The You Only Look Once (YOLO) version 7 model was used in the object detection phase for each person and each piece of equipment detected on the image. While the model's success can be quantified as 86% success for person detection, this rate is 88% for equipment detection. Considering other metrics used to monitor the success of the model, the F1 score for all classes was observed to be 89%. The model shows

successful performance in two classes. At this stage, the model performs class discrimination successfully. This provides the necessary input for detecting person-and-equipment interactions.

In the object detection phase, the movement status of the person and equipment classes detected in the object detection section are monitored. Routes of people and equipment, information about motion vectors, etc. Information must be gathered and used in the action/event detection section. There were certain identity problems in the SORT algorithm that was first implemented. In this case, the Deep SORT algorithm, which will not cause any problems in terms of changing the algorithm or identity problems and has a higher accuracy rate, was used. In the Deep SORT algorithm, the MOTA (Multiple Object Tracking Accuracy) value is 85%, while the MOTP (Multiple Object Tracking Precision) value is 89.2%.

In the action/event detection section, autoencoder model training was completed with the inputs obtained from real-time images and the most appropriate parameters. With the model, any abnormal situation, if any, between people and equipment is detected and classified. With the system, it is determined whether the interaction between person and equipment will take place by using information such as object type, object speed, object position, object's motion vector, and movement angle determined from the real-time image. Detected anomalies are classified by threshold values. Following this classification, problematic images were recorded. In addition, according to this anomaly classification, three different alarm types have been created: encounter, near-miss, and emergency.

Thanks to the determined alarm levels, before possible accidents occur, this possibility will be evaluated in the encounter and near-miss area, and sanctions will be taken to prevent the possibility of a collision between people and equipment. In this case, the measures to be taken will be taken proactively and the number of accidents resulting from human-equipment interaction will be minimized thanks to the system. As the success of the system increases in real-time, all accident possibilities will be prevented. This approach can be applied in all logistics warehouses where camera images can be taken with appropriate equipment.

6.2. Future Works

In the event of any interaction between people and equipment to be used in logistics warehouses, the alarm system inside the warehouse will be activated to prevent any possible accidents. When this situation is evaluated, model requirements may change as anomaly conditions change in the warehouse, and at the same time, the model can be improved according to test results. The testing phases of the system will take place in real warehouses, ensuring that the system prevents occupational accidents caused by the interaction of pedestrians and equipment in the live warehouse environment.

It can also be said that one of the problems encountered while creating the system is data. The ready data set consisting of images in the warehouse is quite small and insufficient for the creation of such complex systems. If datasets consisting of images with high data quality are created and the lack of dataset is eliminated, the success of the model can be increased and the potential success of the models to be created in this field can be increased.

In addition, with this system, requirements regarding occupational health and environmental safety will arise in the warehouse. Furthermore, the warehouse environment will be equipped with the capacity to operate with the highest level of safety and the lowest number of work-related accidents. In other stages, a system that can prevent fatal accidents on ramps can be created with security cameras. Additionally, security measures can be taken with real-time video.

REFERENCES

Amit, Yali, Pedro Felzenszwalb, and Ross Girshick. 2021. "Object Detection." *In Springer eBooks*, 875–83. https://doi.org/10.1007/978-3-030-63416-2_660.

Bakheet, Samy, and Ayoub Al-Hamadi. 2022. "A Deep Neural Framework for Real-time Vehicular Accident Detection Based on Motion Temporal Templates." *Heliyon* 8, no. 11: e11397. https://doi.org/10.1016/j.heliyon.2022.e11397.

Bas, Gulsen, and Ahmet Murat Koseoglu. 2019. "Analysis of the Warehouse Work Accidents in Logistics Sector." *Journal of Business, Economics and Finance* 9, no. 1: 262–68. https://doi.org/10.17261/pressacademia.2019.1102.

Bewley, Alex, Zong Yuan Ge, Lionel Ott, Fabio Tozeto Ramos, and Ben Upcroft. 2016. "Simple Online and Realtime Tracking." *2016 IEEE International Conference on Image Processing (ICIP)*, 3464–68. https://doi.org/10.1109/ICIP.2016.7533003

Borji, Ali, Ming-Ming Cheng, Huaizu Jiang, and Jia Li. 2015. "Salient Object Detection: A Benchmark." *IEEE Transactions on Image Processing* 24, no. 12: 5706–22. https://doi.org/10.1109/tip.2015.2487833.

Chang, Chuan-Wang, Chuan-Yu Chang, and You-Ying Lin. 2022. "A Hybrid CNN and LSTM-based Deep Learning Model for Abnormal Behavior Detection." *Multimedia Tools and Applications* 81, no. 9: 11825–43. https://doi.org/10.1007/s11042-021-11887-9.

Dai, Dengyuhan. 2021. "An Introduction of CNN: Models and Training on Neural Network Models." *International Conference on Big Data, Artificial Intelligence and Risk Management*, November. https://doi: 10.1109/ICBAR55169.2021.00037.

Dawar, Neha, and Nasser Kehtarnavaz. 2018. "Action Detection and Recognition in Continuous Action Streams by Deep Learning-Based Sensing Fusion." *IEEE Sensors Journal* 18, no. 23: 9660–68. https://doi.org/10.1109/jsen.2018.2872862.

Drezner, Zvi, Ofir Turel, and Dawit Zerom. 2010. "A Modified Kolmogorov–Smirnov Test for Normality." *Communications in Statistics. Simulation and Computation* 39, no. 4: 693–704. https://doi.org/10.1080/03610911003615816.

Everingham, Mark, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. 2009. “The Pascal Visual Object Classes (VOC) Challenge.” *International Journal of Computer Vision* 88, no. 2: 303–38. https://doi.org/10.1007/s11263-009-0275-4.

Fernandes, Wander, Karin Satie Komati, and Kelly Assis De Souza Gazolli. 2023. “Anomaly Detection in Oil-producing Wells: A Comparative Study of One-class Classifiers in a Multivariate Time Series Dataset.” *Journal of Petroleum Exploration and Production Technology* 14, November. https://doi.org/10.1007/s13202-023-01710-6.

Foley, James D., Richard L. Phillips, John F. Hughes, Andries Van Dam, and Steven K. Feiner. 1993. Introduction to Computer Graphics. http://ci.nii.ac.jp/ncid/BA21516366.

Ghosh, Sreyan, Sherwin Joseph Sunny, and Rohan Roney. 2019. “Accident Detection Using Convolutional Neural Networks.” *2019 International Conference on Data Science and Communication (IconDSC)*. https://doi.org/10.1109/icondsc.2019.8816881.

Girshick, Ross. 2015. “Fast R-CNN.” *2015 IEEE International Conference on Computer Vision (ICCV)*, December. https://doi.org/10.1109/iccv.2015.169.

Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, et al. 2018. “Recent Advances in Convolutional Neural Networks.” *Pattern Recognition* 77: 354–77. https://doi.org/10.1016/j.patcog.2017.10.013.

Hanbay, Kazım, and Üzen Hüseyin. 2017. “Nesne tespit ve takip metotları: Kapsamlı bir derleme.” *TDFD* 6, no. 2: 40–49. http://dergipark.gov.tr/download/article-file/384735

Hartley, Richard, and Zisserman, Andrew. 2004. “Multiple view geometry in computer vision.” https://doi.org/10.1017/cbo9780511811685

Hmidani, Oussama, and E. M. Ismaili Alaoui. 2022. “A comprehensive survey of the R-CNN family for object detection.” *5th International Conference on Advanced Communication Technologies and Networking (CommNet)*. https://doi.org/10.1109/commnet56067.2022.9993862.

Hossain, Md. Anwar, and Md. Shahriar Alam Sajib. 2019. “Classification of Image Using Convolutional Neural Network (CNN).” *Global Journal of Computer Science and Technology*, May, 13–18. https://doi.org/10.34257/gjcstdvol19is2pg13.

Hou, Rui, Chen Chen, and Mubarak Shah. 2017. “Tube Convolutional Neural Network (T-CNN) for Action Detection in Videos.” *2017 IEEE International Conference on Computer Vision (ICCV)*, October. https://doi.org/10.48550/arXiv.1703.10664

Hou, Xinyu, Yi Wang, and Lap-Pui Chau. 2019. “Vehicle Tracking Using Deep SORT with Low Confidence Track Filtering.” *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–6. https://doi.org/10.1109/avss.2019.8909903.

HumanSignal. n.d. “GitHub - HumanSignal/labelImg: LabelImg Is Now Part of the Label Studio Community. The Popular Image Annotation Tool Created by Tzutalin Is No Longer Actively Being Developed, but You Can Check Out Label Studio, the Open Source Data Labeling Tool for Images, Text, Hypertext, Audio, Video and Time-series Data.” GitHub. https://github.com/HumanSignal/labelImg/tree/master.

Hung, Chung-Wen, Wei-Ting Li, Wei-Lung Mao, and Pal-Chun Lee. 2019. “Design of a Chamfering Tool Diagnosis System Using Autoencoder Learning Method.” *Energies* 12, no. 19: 3708. https://doi.org/10.3390/en12193708.

Jernbäcker, Axel. "Kalman Filters as an Enhancement to Object Tracking Using YOLOv7." Master's thesis, KTH Royal Institute of Technology, 2022. https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-325862.

Jiang, Peiyuan, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. 2022. “A Review of Yolo Algorithm Developments.” *Procedia Computer Science* 199 (January): 1066–73. https://doi.org/10.1016/j.procs.2022.01.135.

Joshi, Kinjal A, and Darshak G. Thakore. 2012. “A Survey on Moving Object Detection and Tracking in Video Surveillance System.” *International Journal of Soft Computing and Engineering (USC E)* 2, no. 3. http://www.ijsce.org/attachments/File/v2i3/C0675052312.pdf.

Kalaycı, Tolga Ahmet, and Umut Asan. 2022. “Improving Classification Performance of Fully Connected Layers by Fuzzy Clustering in Transformed Feature Space.” *Symmetry* 14, no. 4: 658. https://doi.org/10.3390/sym14040658.

Kayikci, Yasanur. 2018. “Sustainability Impact of Digitization in Logistics.” *Procedia Manufacturing* 21 (January): 782–89. https://doi.org/10.1016/j.promfg.2018.02.184.

Kuyucu, Çağatay. 2016. “Lojistik Faaliyetlerinde Depolama Süreçlerinin İş Sağlığı ve Güvenliği Risklerinin Değerlendirilmesi.” Master’s Thesis in Occupational Health and Safety, T.C. Çalışma ve Sosyal Güvenlik Bakanlığı İş Sağlığı ve

Güvenliği Genel Müdürlüğü. https://www.csgb.gov.tr/media/1382/cagataykuyucu.pdf.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning." *Nature* 521, no. 7553: 436–44. https://doi.org/10.1038/nature14539.

Lee, Seungkwan, Suha Kwak, and Minsu Cho. 2019. "Universal Bounding Box Regression and Its Applications." In *Lecture Notes in Computer Science*, 373–87. https://doi.org/10.1007/978-3-030-20876-9_24.

Li, Hongyang, Huchuan Lu, Zhe Lin, Xiaohui Shen, and Brian Price. 2015. "Inner and Inter Label Propagation: Salient Object Detection in the Wild." *IEEE Transactions on Image Processing* 24, no. 10: 3176–86. https://doi.org/10.1109/tip.2015.2440174.

Li, Lv Tang Bo, Yijie Zhong, Shouhong Ding, and Mofei Song. 2021. "Disentangled High Quality Salient Object Detection." *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, October. https://doi.org/10.1109/iccv48922.2021.00356.

Liashchynskyi, Petro, and Pavlo Liashchynskyi. 2019. "Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS." *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.1912.06059.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2014. "Microsoft COCO: Common Objects in Context." *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.1405.0312.

Linguo, Chai, Pang Haojie, ShangGuan Wei, and Cai Baigen. 2022. "Method of Multi-lane Vehicles Speed Continuously Perceiving Based on Single Roadside Camera." *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, October. https://doi.org/10.1109/itsc55140.2022.9921759.

Liu, Chun, Sixuan Zhang, Mengjie Hu, and Qing Song. 2024. "Object Detection in Remote Sensing Images Based on Adaptive Multi-Scale Feature Fusion Method." *Remote Sensing* 16, no. 5: 907. https://doi.org/10.3390/rs16050907.

Liu, Haiying, Yuncheng Pei, Qiancheng Bei, and Lixia Deng. 2022. "Improved DeepSORT Algorithm Based on Multi-Feature Fusion." *Applied System Innovation* 5, no. 3: 55. https://doi.org/10.3390/asi5030055.

Liu, Jiahang, Donghao Yang, and Fei Hu. 2022. "Multiscale Object Detection in Remote Sensing Images Combined With Multi-Receptive-Field Features and

Relation-Connected Attention." *Remote Sensing* 14, no. 2: 427. https://doi.org/10.3390/rs14020427.

Liu, Li, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. 2019. "Deep Learning for Generic Object Detection: A Survey." *International Journal of Computer Vision* 128, no. 2: 261–318. https://doi.org/10.1007/s11263-019-01247-4.

Mac, Hieu, Dung Truong, Lam Nguyen, Hoa Nguyen, Hai Anh Tran, and Duc Tran. 2018. "Detecting Attacks on Web Applications using Autoencoder." *SoICT '18: Proceedings of the 9th International Symposium on Information and Communication Technology*, December, 416–21. https://doi.org/10.1145/3287921.3287946.

Mahesh, Batta. 2020. "Machine Learning Algorithms - a Review." *International Journal of Science and Research (IJSR)* 9, no.1. https://doi.org/10.21275/ART20203995.

Mostafa, Mahin, Sami Sadi, Sadiya Afrose Anamika, Md. Shahriar Hussain, and Riasat Khan. 2023. "Automatic Vehicle Classification and Speed Tracking." *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, May, 972–77. https://doi.org/10.1109/icaaic56838.2023.10140935.

Murthy, Chinthakindi Balaram, Mohammad Farukh Hashmi, Neeraj Dhanraj Bokde, and Zong Woo Geem. 2020. "Investigations of Object Detection in Images/Videos Using Various Deep Learning Techniques and Embedded Platforms—A Comprehensive Review." *Applied Sciences* 10, no. 9: 3280. https://doi.org/10.3390/app10093280.

"Must-Know Warehouse Injury Statistics." 2024. GitNux. https://gitnux.org/warehouse-injury-statistics/.

Olorunshola, Oluwaseyi, Paul Jemitola, and Adeniran Ademuwagun. 2023. "Comparative Study of Some Deep Learning Object Detection Algorithms: R-CNN, FAST R-CNN, FASTER R-CNN, SSD, and YOLO." *Nile Journal of Engineering and Applied Sciences* 1, no. 1. https://doi.org/10.5455/njeas.150264.

"OpenCV: Camera Calibration." n.d. https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html.

O'Shea, Keiron, and Ryan Nash. 2015. "An Introduction to Convolutional Neural Networks." *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.1511.08458.

Öztürk, Ayşegül. 2010. “Efficient Warehouse Management and a Study on Efficient Warehouse Strategies of Logistic Warehouses.” İstanbul Ticaret Üniversitesi. https://tez.yok.gov.tr/UlusalTezMerkezi/tezDetay.jsp?id=GprIcvA37IdmIS4H0uZ7UA&no=-crXBIm6LTQC_VO-atHolg

Pathak, Ajeet Ram, Manjusha Pandey, and Siddharth Rautaray. 2018. “Application of Deep Learning for Object Detection.” *Procedia Computer Science* 132 (January): 1706–17. https://doi.org/10.1016/j.procs.2018.05.144.

Pereira, Ricardo, Guilherme Carvalho, Luís Garrote, and Urbano J. Nunes. 2022. “Sort and Deep-SORT Based Multi-Object Tracking for Mobile Robotics: Evaluation With New Data Association Metrics.” *Applied Sciences* 12, no. 3: 1319. https://doi.org/10.3390/app12031319.

Quach, Luyl-Da, Khang Nguyen Quoc, Anh Nguyen Quynh, and Hoang Tran Ngoc. 2023. “Evaluating the Effectiveness of YOLO Models in Different Sized Object Detection and Feature-Based Classification of Small Objects.” *Journal of Advances in Information Technology* 14, no. 5: 907–17. https://doi.org/10.12720/jait.14.5.907-917.

Rajesh, Gokul, Amitha Rossy Benny, A. Harikrishnan, James Jacob Abraham, and Nithin Prince John. 2020. “A Deep Learning based Accident Detection System.” *2020 International Conference on Communication and Signal Processing (ICCSP)*. https://doi.org/10.1109/iccsp48568.2020.9182224.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. “You Only Look Once: Unified, Real-Time Object Detection.” *2016 IEEE Conference on Computer Vision and Pattern Recognition*, June. https://doi.org/10.1109/cvpr.2016.91.

Ren, Jie, Yusu Pan, Pantao Yao, Yicheng Hu, Wang Gao, and Zhenfeng Xue. 2022. “Deep Learning-Based Intelligent Forklift Cargo Accurate Transfer System.” *Sensors* 22, no. 21: 8437. https://doi.org/10.3390/s22218437.

“SGK 2011 / 2012 / 2013 / 2014 İstatistik Yıllığı.” n.d. Ankara, Turkey: T.C. Çalışma ve Sosyal Güvenlik Bakanlığı.

Sultana, Farhana, Abu Sufian, and Prasun Dutta. 2020. “A review of object detection models based on convolutional neural network.” *Intelligent Computing: Image Processing Based Applications*, 1–16. https://doi.org/10.48550/arXiv.1905.01614

Sun, Lipeng, Shihua Zhao, Gang Li, and Binbing Liu. 2019. “High Accuracy Object Detection via Bounding Box Regression Network.” *Frontiers of Optoelectronics* 12, no. 3: 324–31. https://doi.org/10.1007/s12200-019-0853-1.

Tithi, Jesmin Jahan, Sriram Aananthakrishnan, and Fabrizio Petrini. 2020. “Online and Real-time Object Tracking Algorithm With Extremely Small Matrices.” *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.2003.12091.

Toktaş-Palut, Peral, and Fırat Okçuoğlu. 2019. “Depo Tasarımı ve Yerleşimi: Bir Gerçek Hayat Uygulaması.” *Beykent Üniversitesi Fen Ve Mühendislik Bilimleri Dergisi* 12, no. 2: 14–22. https://doi.org/10.20854/bujse.577992.

Vahdani, Elahe, and Yingli Tian. 2022. “Deep Learning-based Action Detection in Untrimmed Videos: A Survey.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, January, 1–20. https://doi.org/10.1109/tpami.2022.3193611.

Vujovic, Željko Đ. 2021. “Classification Model Evaluation Metrics.” *International Journal of Advanced Computer Science and Applications/International Journal of Advanced Computer Science & Applications* 12, no. 6. https://doi.org/10.14569/ijacsa.2021.0120670.

Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. 2022. “YOLOv7: Trainable Bag-of-freebies Sets New State-of-the-art for Real-time Object Detectors.” *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.2207.02696.

Wang, Xiaoyu, Ming Yang, Shenghuo Zhu, and Yuanqing Lin. 2013. “Regionlets for Generic Object Detection.” *2013 IEEE International Conference on Computer Vision*, December, 17–24. https://doi.org/10.1109/iccv.2013.10.

Xi, Xuanyang, Yongkang Luo, Peng Wang, and Hong Qiao. 2019. “Salient Object Detection Based on an Efficient End-to-End Saliency Regression Network.” *Neurocomputing* 323 (January): 265–76. https://doi.org/10.1016/j.neucom.2018.10.002.

Xu, Zhongwen, Yi Yang, and Alexander G. Hauptmann. 2014. “A Discriminative CNN Video Representation for Event Detection.” *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.1411.4006.

Yadav, Durgesh Kumar, None Renu, None Ankita, and Iftisham Anjum. 2020. “Accident Detection Using Deep Learning.” *2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, December. https://doi.org/10.1109/icacccn51052.2020.9362808.

Yadav, Vinod Kumar, Pritaj Yadav, and Shailja Sharma. 2022. “An Efficient Yolov7 and Deep Sort Are Used in a Deep Learning Model for Tracking Vehicle and Detection.” *Journal of Xi’an Shiyou University* 18, no. 11: 759–63. http://xisdxjxsu.asia.

Yang, Feng, Xingle Zhang, and Bo Liu. 2022. “Video Object Tracking Based on YOLOv7 and DeepSORT.” *arXiv (Cornell University)*, January. https://doi.org/10.48550/arxiv.2207.12202.

Ye, Guanting, Jinsheng Qu, Jintai Tao, Wei Dai, Yifei Mao, and Qiang Jin. 2023. “Autonomous Surface Crack Identification of Concrete Structures Based on the YOLOv7 Algorithm.” *Journal of Building Engineering* 73 (August): 106688. https://doi.org/10.1016/j.jobe.2023.106688.

Yu, Chaoran, Zhejun Feng, Zengyan Wu, Runxi Wei, Baoming Song, and Changqing Cao. 2023. “HB-YOLO: An Improved YOLOv7 Algorithm for Dim-Object Tracking in Satellite Remote Sensing Videos.” *Remote Sensing* 15, no. 14: 3551. https://doi.org/10.3390/rs15143551.

Zhang, Xin, Yee-Hong Yang, Zhiguang Han, Hui Wang, and Chao Gao. 2013. “Object Class Detection.” *ACM Computing Surveys* 46, no. 1: 1–53. https://doi.org/10.1145/2522968.2522978.

Zhao, Zhong-Qiu, Peng Zheng, Shou-Tao Xu, and Xindong Wu. 2019. “Object Detection With Deep Learning: A Review.” *IEEE Transactions on Neural Networks and Learning Systems* 30, no. 11: 3212–32. https://doi.org/10.1109/tnnls.2018.2876865.

Zheng, Jianfeng, Hang Wu, Han Zhang, Zhaoqi Wang, and Weiyue Xu. 2022. “Insulator-Defect Detection Algorithm Based on Improved YOLOv7.” Sensors 22, no. 22: 8801. https://doi.org/10.3390/s22228801.

Zhiqiang, Wang, and Liu Jun. 2017. “A review of object detection based on convolutional neural network.” *36th Chinese Control Conference (CCC)*. https://doi.org/10.23919/chicc.2017.8029130.

Zhu, Wangjiang, Shuang Liang, Yichen Wei, and Jian Sun. 2014. “Saliency Optimization from Robust Background Detection.” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2014.360.

APPENDICES

APPENDIX A

A.1. Action/Event Detection Detailed Experimental Results

In the Action/event detection section, the results regarding the test data are given below. Tests were carried out on data containing accident moments or similar anomalies regarding the created model. The results of the tests and their images are listed below.

There are two anomalies, one that occurred in the first test and one that has a high probability of being an accident. The reconstruction error increases before the accident occurs. Then the accident happens. As seen in this test, the probability of the accident occurring was observed before the accident occurred. Then the accident happened.

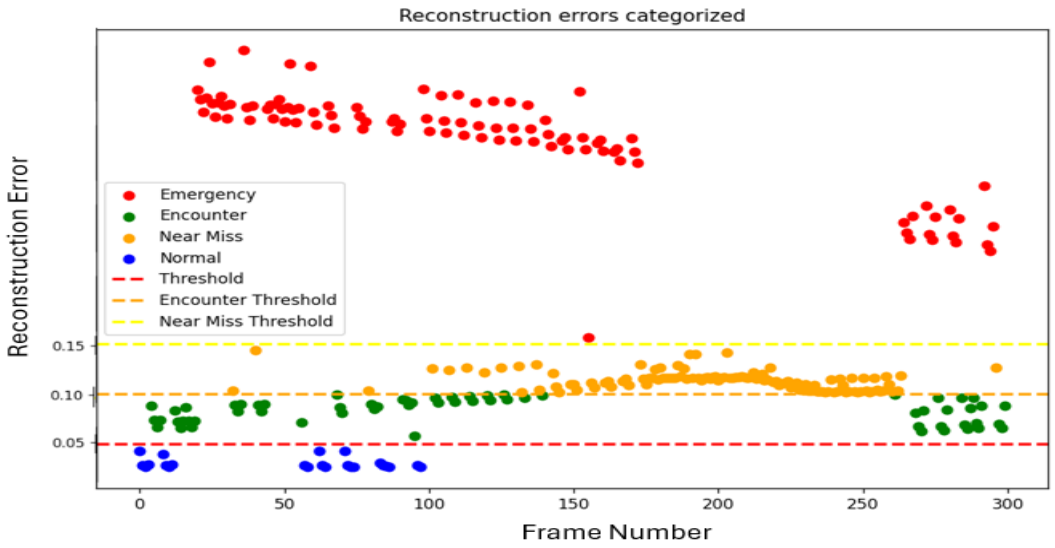

Figure A.1. Graph of Test1 Result

As seen in the figure, this situation is normally addressed when there is no interaction within the warehouse. It was later understood that there would be interaction between the rapidly approaching equipment and the walking person. This situation has been alerted as an encounter. When this alarm was ignored, the accident occurred. The forklift hit the pedestrian at high speed and as a result, the pedestrian was thrown rapidly inside the warehouse. Since there may be interaction between the pedestrian being thrown and the other forklift moving at that moment and the movement of both objects is aggressive, it is marked as "Emergency 2" in this case.

Figure A.2. Test1 Results

In another test data containing anomalies, samples exceeding the reconstruction value were recorded as frames.

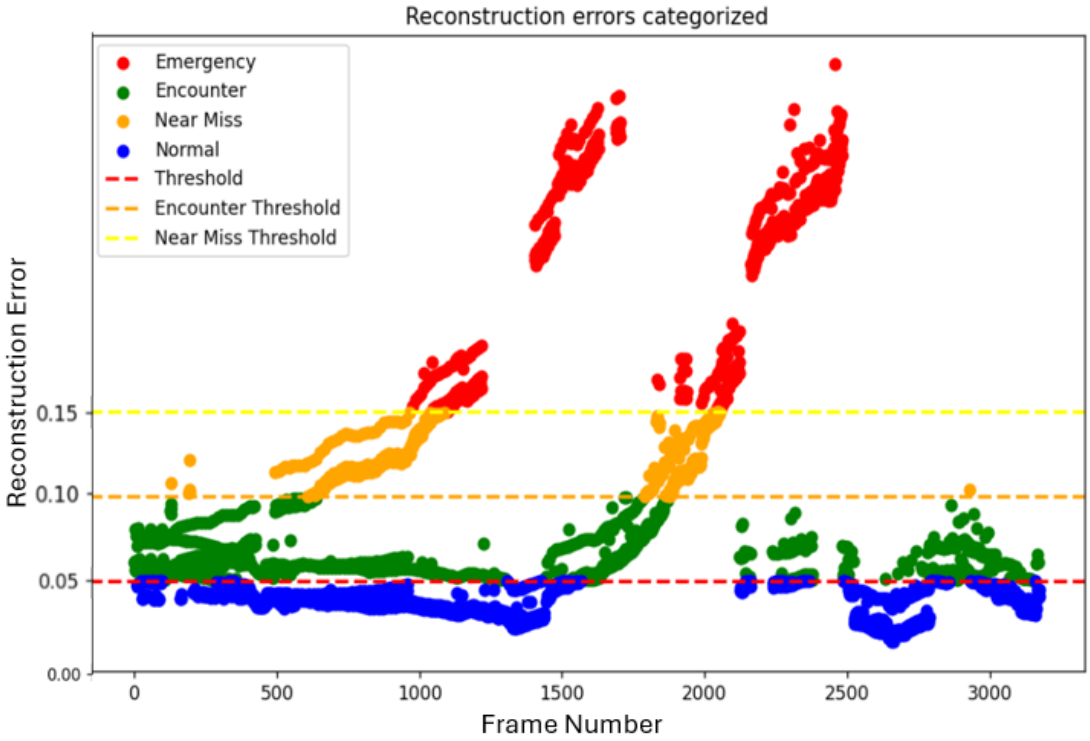

Figure A.3. Graph of Test2 Result

As seen in the Figure A.3, the system gives an alarm when it sees an accident potential, and as the level increases, the alarm level of the system changes to emergency.

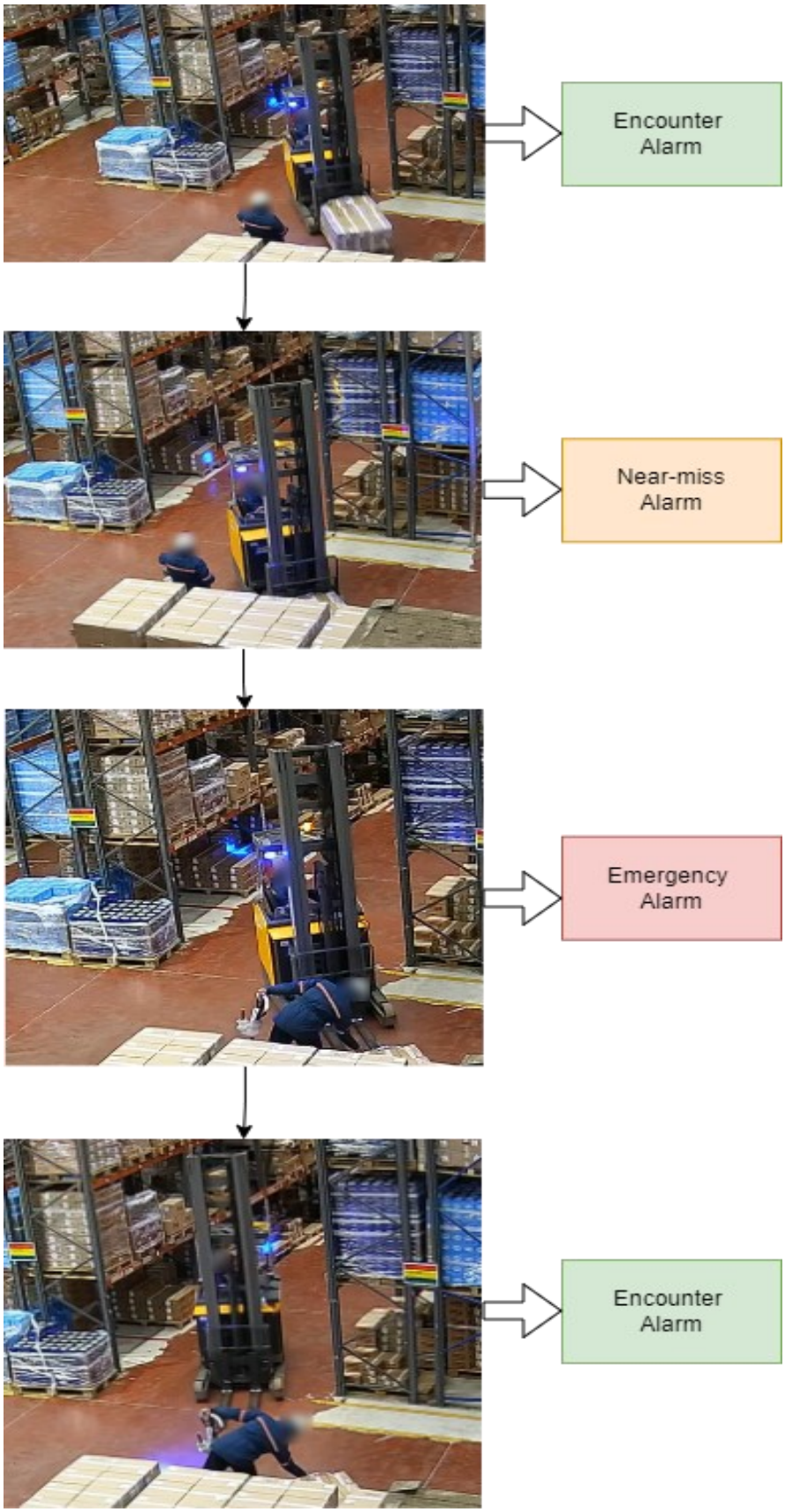

Figure A.4. Test2 Results