T.C.
ISTANBUL ATLAS UNIVERSITY
GRADUATE SCHOOL OF EDUCATION

MASTER'S THESIS

MULTI-TARGET DETECTION& TRACKING USING MACHINE LEARNING METHODOLOGIES

Muhammad JUNAID

DANIŞMAN
Prof. Dr. Naim AJLOUNI

Bilgisayar Mühendisliği Anabilim Dalı

Program Adı
Bilgisayar Mühendisliği(İngilizce) Programı

ISTANBUL, 2024

T.C.
ISTANBUL ATLAS UNIVERSITY
GRADUATE SCHOOL OF EDUCATION

MASTER'S THESIS

MULTI-TARGET DETECTION& TRACKING USING MACHINE LEARNING METHODOLOGIES

Muhammad JUNAID

DANIŞMAN
Prof. Dr. Naim AJLOUNI

Anabilim Dalı Adı

Program Adı

T.C.
ISTANBUL ATLAS UNIVERSITY
GRADUATE INSTITUTE
THESIS APPROVAL PAGE

STUDENT NAME-SURNAME	Muhammad JUNAID
STUDENT NUMBER	222116003
PROGRAM NAME	Computer Engineering (English) Program

The thesis titled "Multi-Target Detection and Tracking Using Machine Learning Methodologies," prepared by Mohammad Junaid in the Department of Computer Engineering at Istanbul Atlas University, has been accepted by the jury as a Master's thesis.

Thesis Defense Date : 12/09/2024

Title, First Name, Last Name of Jury Member	**Institution Affiliated**	**Signature**
Prof. Dr. Naim AJLOUNI	İstanbul Atlas Üniversitesi	
Doç. Dr. Ahmet GÜRHANLI	Topkapı Üniversitesi	
Dr. Öğretim Üyesi Adem ÖZYAVAŞ	İstanbul Atlas Üniversitesi	

This thesis has been approved by the jury in accordance with the relevant articles of the Graduate Education and Examination Regulations of Istanbul Atlas University and has been accepted by the decision of the Institute Administrative Board.

Prof. Dr. Hafize UZUN

Director of the Graduate Education Institute

DECLARATION

I hereby declare that this thesis is my own original work, and that I have adhered to scientific ethical principles and standards at all stages of preparation, data collection, analysis, and presentation of findings. I confirm that I have properly cited all data and information not obtained within the scope of this study and have included these sources in the bibliography. Furthermore, I declare that this work has been screened by the scientific plagiarism detection program used at Istanbul Atlas University and meets the prescribed standards. I acknowledge that if, at any time, it is determined that I have acted contrary to this declaration, I accept all moral and legal consequences that may arise.

Muhammad JUNAID

(İmza)

DEDICATION

I dedicate this to my spouse and child.

BUDGET SUPPORT

MULTI-TARGET DETECTION& TRACKING USING MACHINE LEARNING METHODOLOGIES

No budget support has been received from any institution for this thesis study.

ACKNOWLEDGMENT

I would like to thank and express my sincere gratitude and appreciation to my supervisor Prof. Naim Mahmood Musleh Ajlouni, for believing in me, guiding me, motivating me when it was tough, for giving advice and encouragement, and his time, patience and constant support towards the successful completion of this project.

I would like to also thank my family for their prayers and support while I faced challenges throughout the cycle of this research. I thank friends and colleagues who were there for me, supported me, and encouraged me.

I thank God for his blessings, protection and grace.

Muhammad JUNAID

Table of Contents

LIST OF ABBREVIATIONS

CNNs	Convolutional Neural Networks
Yolo	You Only Look Once
SDD	Single Shot Multibox Detector
SORT	Simple Online and Real Time Tracking

LIST OF FIGURES

Page no

ÖZET

Muhammad, Junaid. (2024). Makine Öğrenme Yöntemleri ile Çok Hedefli Tespit ve Takip. Yüksek Lisans Tezi, İstanbul Atlas Üniversitesi, Lisansüstü Enstitüsü, Bilgisayar Mühendisliği Bölümü, Bilgisayar Mühendisliği Yüksek Lisans Programı, İstanbul.

Bu tez, çok hedefli insan tespiti ve takibi konusunda makine öğrenme yöntemlerini kullanarak kapsamlı bir çalışma sunmakta olup, birden fazla kamera kaynağından elde edilen verilerin entegrasyonuna odaklanmaktadır. Gelişmiş gözetim sistemleri ve otomatik izleme ihtiyacının artmasıyla, dinamik ortamlarda bireylerin etkili bir şekilde tespit edilmesi ve takibi büyük önem kazanmaktadır. Bu araştırma, çeşitli ortamlarda gerçek zamanlı insan tespiti için YOLO (You Only Look Once) ve Faster R-CNN gibi en son derin öğrenme modellerinin uygulanmasını incelemektedir. Mevcut yöntemler üzerine kurulan bu çalışma, yalnızca ana kişileri tespit edip takip etmekle kalmayıp, aynı zamanda yeni tespit edilen bilinmeyen kişileri "ilgi altındaki kişiler" olarak sınıflandıran yenilikçi bir çerçeve sunmaktadır. Bu hiyerarşik yaklaşım, bireyler arasındaki ilişki yönetimini geliştirirken, özellikle kalabalık veya gizlenmiş senaryolarda kimlik sürekliliğinin doğruluğunu artırmaktadır. Önerilen sistem, çoklu kameradan elde edilen çerçevelerde tespitlerin bağlantısını kesintisiz bir şekilde sağlamak ve kimlik değiştirmeleri etkin bir şekilde yönetmek için Macar algoritması ve Deep SORT gibi ileri düzey veri ilişkilendirme tekniklerini kullanmaktadır. Çeşitli veri setleri üzerinde yapılan değerlendirmeler, önerilen yöntemin gerçek dünya uygulamalarında sağlamlığını ve ölçeklenebilirliğini göstererek, çoklu kamera girişlerinin tespit ve takip performansını önemli ölçüde artırdığını ve durumsal farkındalığı geliştirdiğini ortaya koymaktadır. Bu araştırma, insan tespiti ve takip sistemlerindeki mevcut sınırlamaları ele alarak, bilgisayarlı görü alanında büyüyen bir katkı sağlamakta olup, gelecekteki davranış analizi ve etkileşim tanıma araştırmalarına temel oluşturmakta, karmaşık ortamlarda bireyler arasındaki ilişkileri daha iyi bağlamsallaştırmaktadır.

Anahtar Kelimeler: Çok hedefli takip, Makine öğrenimi, Gözetim sistemi.

ABSTRACT

Muhammad,Junaid.(2024).Multi-Target Detection and Tracking using Machine Learning Methodologies. Master's, Istanbul Atlas University Graduate Institute, Department of Computer Engineering, Master's Thesis program in Computer Engineering,Istanbul.

This thesis presents a comprehensive study on multi-target human detection and tracking using machine learning methodologies, with a focus on integrating data from multiple camera sources. As the demand for advanced surveillance systems and automated monitoring increases, effective detection and tracking of individuals in dynamic environments become paramount. This research explores the implementation of state-of-the-art deep learning models, such as YOLO (You Only Look Once) and Faster R-CNN, for real-time human detection across varied environments. Building on existing methodologies, we introduce an innovative framework that not only detects and tracks primary individuals but also identifies and categorizes newly detected unknown individuals as "sub persons of interest." This hierarchical approach allows for enhanced relationship management between individuals and improves the accuracy of identity retention over time, especially in crowded or occluded scenarios. The proposed system employs advanced data association techniques, such as the Hungarian algorithm and Deep SORT, to seamlessly link detections across frames from multiple cameras while managing identity switches effectively. Evaluations conducted on diverse datasets highlight the effectiveness of the proposed method, demonstrating its robustness and scalability in real-world applications. Our findings indicate that leveraging multi-camera inputs significantly enhances detection and tracking performance, providing improved situational awareness. This research contributes to the growing field of computer vision by addressing current limitations in human detection and tracking systems, paving the way for future advancements in surveillance technologies and intelligent monitoring solutions. This work lays foundational concepts for future research into behavioral analysis and interaction recognition, further contextualizing the relationships between individuals in complex environments.

Keywords: Multi-target tracking, Machine learning, Surveillance system

1. INTRODUCTION

1.1 RESEARCH BACKGROUND

Human detection and tracking in dynamic environments present significant challenges in the realms of computer vision and artificial intelligence. With the rapid proliferation of surveillance systems and automated monitoring technologies, there is a growing demand for robust algorithms capable of accurately identifying, tracking, and managing multiple targets simultaneously.

Recent advancements in deep learning have markedly improved the performance of human detection algorithms. For instance, models like YOLO (You Only Look Once) and Faster R-CNN have showcased exceptional accuracy in real-time object detection tasks, enabling the swift and efficient identification of individuals in complex scenes (1, 2). While detecting individuals is critical, the effective tracking of these individuals across multiple camera sources introduces additional challenges. Real-world scenarios often entail occlusion, varying viewpoints, and intricate interactions among multiple targets, thereby complicating the task of maintaining accurate identities.

In multi-target tracking systems, preserving identity across frames is imperative. Traditional methods such as Kalman filtering and particle filters have been effectively utilized in tracking algorithms (3, 4). However, these approaches can face difficulties in scenarios where individuals are in close proximity or when sudden changes in appearance occur. The integration of advanced data association techniques, including the Hungarian algorithm and modern approaches like Deep SORT, has enhanced the ability to link detections across frames, successfully managing occlusions and identity switches (5, 6).

In contexts where multiple camera sources are employed, the system must address challenges related to data fusion and camera synchronization (7). By implementing multi-source tracking systems, enhanced coverage and robustness can be achieved in human detection and tracking tasks. An innovative approach in this domain involves establishing a hierarchy of interest, wherein newly detected individuals can be categorized as "sub persons of interest"

linked to a main "person of interest" when certain conditions are fulfilled, such as specific behavioral patterns or contextual relevance.

This introduction underscores the emerging complexities in human detection and tracking within multi-camera environments. The subsequent sections will explore the methodologies employed to address these issues, providing insights into the state-of-the-art techniques that

2. LITERATURE REVIEW

Human detection and tracking are integral components of computer vision systems, with applications ranging from surveillance and security to human-computer interaction and autonomous systems. This literature review provides an overview of the key methodologies, technologies, and challenges related to human detection and tracking, emphasizing multi-target scenarios, multi-camera systems, and the concept of sub persons of interest.

Historically, human detection algorithms utilized traditional feature-based methods such as Haar cascades and Histogram of Oriented Gradients (HOG) combined with classifiers like Support Vector Machines (SVMs). These early approaches laid the groundwork but struggled with issues such as false negatives in crowded scenes and sensitivity to varying lighting conditions (8, 9).

The field has since seen a transformation with the advent of deep learning. Convolutional Neural Networks (CNNs) have significantly improved detection accuracy. Models such as YOLO (You Only Look Once) and Faster R-CNN have demonstrated the capability to perform real-time detection by dividing the image into grid cells and predicting bounding boxes and class probabilities for multiple objects simultaneously (10, 11). YOLO, in particular, stands out for its speed, processing frames at real-time rates, making it suitable for video applications.

Tracking algorithms are designed to monitor the presence and movement of detected individuals over time. Traditional tracking methods, such as Kalman filters, estimate the trajectory of moving objects based on previous states (12). Particle filters have also been employed to handle non-linear motions and complex observation models, expanding the capabilities of tracking systems (13).

More advanced tracking methodologies, like SORT (Simple Online and Real-time Tracking) and Deep SORT, integrate deep learning techniques to improve object association by utilizing appearance features alongside motion models. These methods have gained popularity owing to their effectiveness in handling occlusions and changes in perspective, which are common in real-world scenarios (14, 15).

The challenge of multi-target tracking stems from the need to maintain identities across multiple objects simultaneously under conditions of occlusion, interaction, and appearance changes.

Research in this area has focused on improving data association techniques to accurately link detections across video frames. Techniques such as the Hungarian algorithm enable efficient assignment of detected objects to predicted trajectories based on proximity (16).

Additionally, hierarchical or network-based approaches to tracking relationships among individuals are gaining traction. The exploration of sub persons of interest in connection to main persons of interest enhances the understanding of group dynamics and interactions within crowded environments. This relationship helps in contextualizing movements, allowing for better tracking fidelity in scenarios where multiple individuals are present (17).

Multi-camera systems have become increasingly important for addressing occlusions and blind spots, especially in crowded or complex environments. The integration of multiple camera feeds allows for a comprehensive view and the sharing of information between cameras, leveraging data fusion technologies to enhance detection and tracking performance. Research by Khan et al. suggests that employing multiple cameras improves the robustness of tracking systems, allowing for more accurate identity retention over time through synchronized observations (18).

The challenges associated with multi-camera setups include synchronization of timestamps, camera calibration, and merging data accurately. Proper calibration ensures consistent spatial references, while synchronization allows the system to maintain continuity in tracking individuals across overlapping fields of view (19).

Current research is progressively focusing on incorporating behavioural analysis and social interactions in tracking systems. As an extension of traditional tracking methodologies, exploring the behavioural patterns of individuals can provide richer contextual information, enabling smarter and more adaptive human detection systems (20). Additionally, the use of advanced machine learning techniques, such as generative models and reinforcement learning, is beginning to enter the domain of human tracking, offering potential avenues for improving object classification and tracking accuracy through adaptive learning (21).

The field of human detection and tracking is rapidly evolving, with advancements in deep learning significantly enhancing detection accuracy and tracking robustness. While existing techniques have demonstrated success in various applications, challenges such as occlusion, similarity in appearances, and variations in environmental conditions remain prevalent. This literature review elucidates the key advancements and ongoing challenges within the field, establishing a framework for the proposed multi-target human detection and tracking system that incorporates multi-camera inputs and the recognition of sub persons of interest.

3. MACHINE LEARNING METHODS

3.1. INTRODUCTION

This chapter provides an in-depth exploration of the machine learning methodologies employed in this thesis for multi-target human detection and tracking. The advancement of machine learning, particularly through deep learning techniques, has revolutionized the field of computer vision. This chapter is structured into sections detailing supervised learning techniques, deep learning models, data association algorithms, re-identification strategies, performance evaluation metrics, and applications of machine learning in various fields, including medical imaging and quantum-enabled machine learning algorithms.

3.1.1 Supervised Learning Techniques

Supervised learning is a method where models are trained on labelled datasets, allowing the algorithms to learn the characteristics of the target classes. In the context of human detection and tracking, various supervised learning techniques have shown significant effectiveness.
Supervised learning is a machine learning approach where models are trained using labelled data, meaning each training example comes with a known output label. The model learns to make predictions by associating inputs with these output labels based on its training.
The goal of supervised learning is to create a mapping from input features to output labels. This is achieved through a training dataset comprising pairs of inputs and their corresponding outputs.

Supervised learning includes classification and regression. This type involves predicting a discrete label, such as determining if an email is spam or not. Regression involves forecasting a continuous value, like estimating house prices based on factors such as size and location.

The key Components of supervised learning include training Set, a set of input-output pairs used for teaching the model. The second component is testing, in this case a distinct set used to assess the model's performance after training.

The Loss Function in supervised learning is a function that quantifies the discrepancy between the predicted and actual values. Examples include Mean Squared Error (MSE) for regression and Cross-Entropy Loss for classification tasks.

The optimization in supervised learning involves the process of fine-tuning the model parameters to minimize the loss function. Techniques like Gradient Descent are commonly used to achieve this.

Many supervised learning algorithms use linear models where the relationship between inputs and outputs is described as a linear combination of features. This relationship is mathematically represented as:

$$y = W^T X + b \tag{3.1}$$

where y denotes the predicted output, W is the weight vector, X is the input vector, and b represents the bias term.

The Loss Functions is usually Mean Squared Error (MSE) for regression:

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.2}$$

The Cross-Entropy Loss for classification is

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \widehat{y_i})) \tag{3.3}$$

The Gradient Descent is an optimization technique that minimizes the loss function by iteratively adjusting the model parameters W and b. The update rules are:

$$W \coloneqq W - \alpha \nabla L(W) \tag{3.4}$$

$$b \coloneqq b - \alpha \nabla L(b) \tag{3.5}$$

where α represents the learning rate, and ∇L is the gradient of the loss function concerning the parameters.

Overfitting happens when a model captures noise in the training data rather than the underlying patterns. Regularization techniques, such as Lasso ($L1$) and Ridge ($L2$), help mitigate overfitting by adding a penalty to the loss function. The Lasso Regularization is

$$L_{lasso} = L_{original} + \lambda \|W\|_1 \tag{3.6}$$

In these equations, λ) is the regularization parameter.

$$L_{lasso} = L_{original} + \lambda \|W\|_2^2 \tag{3.7}$$

The supervised learning encompasses a range of techniques for making predictions based on labelled data. By understanding its theoretical principles and mathematical underpinnings, practitioners can develop models that effectively generalize to new, unseen data.

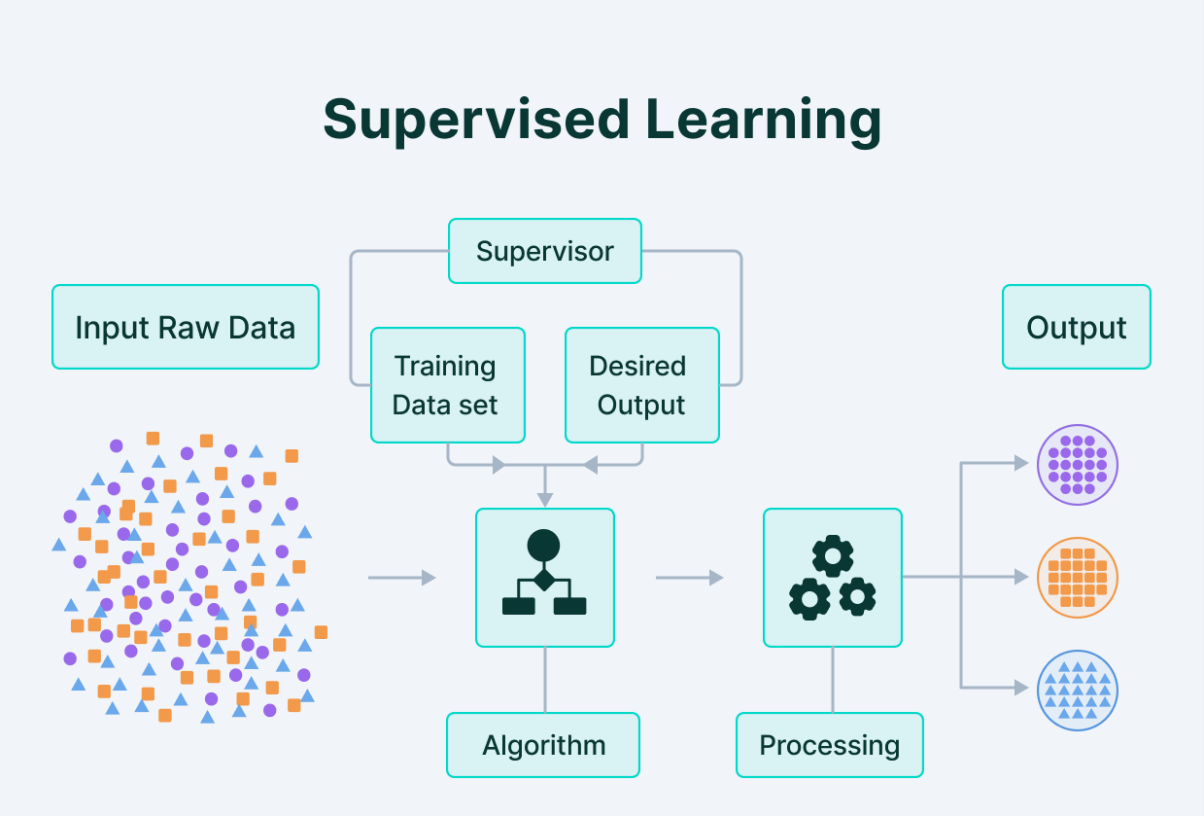

Figure 3.1: Supervised Learning Flowchart

3.1.2. Support Vector Machines (SVM)

SVM is a classification algorithm that seeks to find a hyperplane that best separates different classes in the feature space. For human detection, features extracted from the inputs, such as HOG or color histograms, can be used to train SVM classifiers, distinguishing between human and non-human classes effectively (22).

The Support Vector Machines (SVMs) are a class of supervised learning algorithms designed for both classification and regression tasks. Here's an overview of the theory and mathematical framework underlying SVMs:

The main goal of SVM is to identify a hyperplane that optimally separates data points from different classes in a high-dimensional space. This hyperplane is chosen to maximize the margin, which is the distance between the nearest data points of each class.

The margin is the distance between the hyperplane and the closest points from either class, known as support vectors. SVM aims to maximize this margin to improve classification performance.

The hyperplane, in an n-dimensional space, a hyperplane can be expressed by the equation:

$$w^T x + b = 0 \quad (3.8)$$

Where w represents the weight vector (normal to the hyperplane), x is the feature vector, b is the bias term.

The support vectors are the data points that lie closest to the hyperplane. They are critical because they determine the position and orientation of the hyperplane. The mathematical framework is formulated using a training set of labelled data points (x_i, y_i), where $y_i \in \{-1, +1\}$, SVM solves the following optimization problem:

$$minimize\ \frac{1}{2}\|w\|^2$$

Subject to:

$$y_i(w^T x_i + b) \geq\ \forall_i \quad (3.9)$$

This problem ensures correct classification of each data point while maximizing the margin between classes. To address the constrained optimization problem, SVM uses Lagrange multipliers. The Lagrangian is formulated as:

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i(w^T x_i + b - 1) \quad (3.10)$$

where, α_i are the Lagrange multipliers, and m denotes the number of training samples.

By differentiating the Lagrangian with respect to w and b and setting these derivatives to zero, we derive the dual problem:

$$maximize\ W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

(3.11)

subject to:

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0$$

where, $K(x_i, x_j)$ is the kernel function that enables SVM to operate in a higher-dimensional space.

To handle non-linearly separable data, SVM employs the kernel trick. Common kernel functions include:

- Linear Kernel: $K(x_i, x_j) = x_i^T x_j$
- Polynomial Kernel: $K(x_i, x_j) = (x_i^T x_j + c)^d$
- Radial Basis Function (RBF) Kernel: $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$

Kernels transform input features into higher-dimensional spaces, facilitating the identification of a linear hyperplane in that space.

Real-world data may not be perfectly separable. The soft margin approach introduces slack variables ξ_i to accommodate some misclassifications:

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \tag{3.12}$$

The revised optimization problem is:

$$minimize\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i \tag{3.13}$$

In this case, C is a hyperparameter that balances the margin maximization and classification error minimization.

The Support Vector Machines offer a robust approach for classification and regression by finding an optimal hyperplane that maximizes the margin between classes. The mathematical foundation of SVM, including Lagrange multipliers, the dual formulation, and kernel functions, allows it to handle both linearly and non-linearly separable data effectively, maintaining strong generalization capabilities.

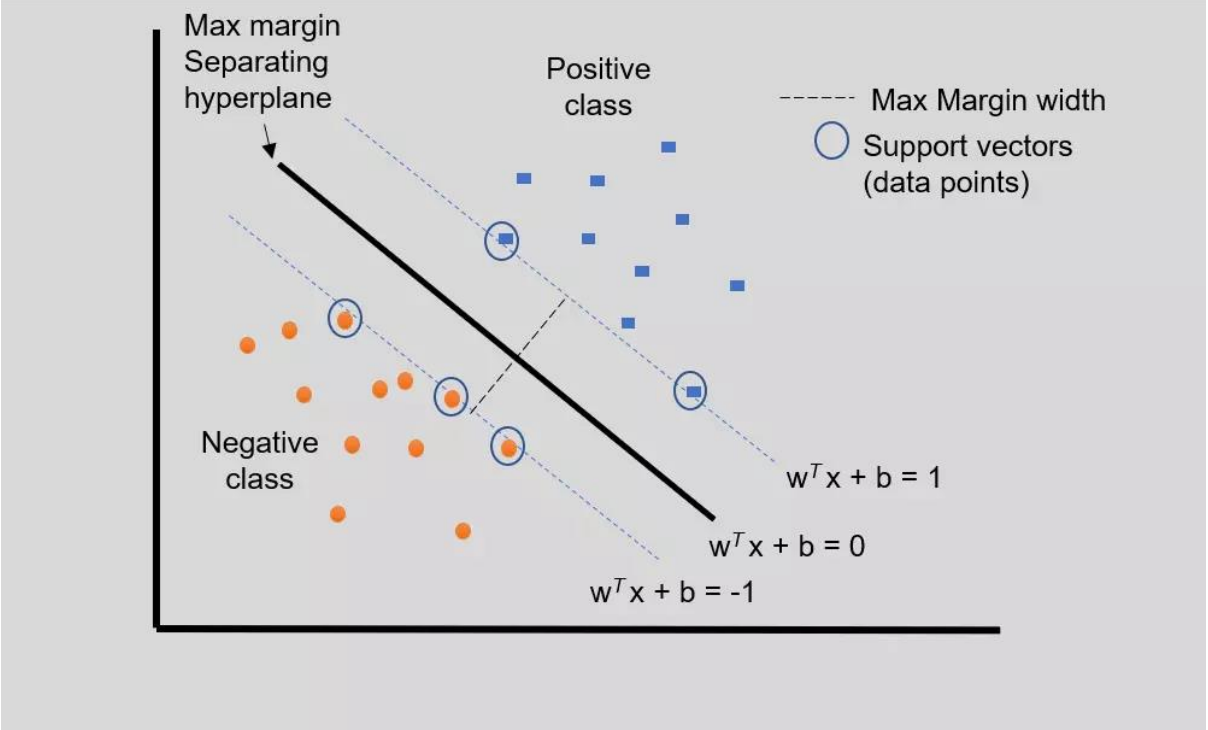

Figure 3.2: Linear SVM for Simple Two-Class Classification with Separating Hyperplane

3.1.3. Decision Trees and Random Forests

Decision trees split the dataset into branches based on feature values, leading to predictions at the leaf nodes. Random forests, an ensemble of decision trees, improve prediction accuracy and robustness by aggregating the results of multiple trees. These techniques can be employed for both detection and classification tasks (23).

A Decision Tree is a model used for both classification and regression tasks, structured like a flowchart. It works by splitting the dataset into subsets based on the feature that provides the greatest information gain or reduces impurity the most, making decisions by traversing the tree from the root to a leaf node.

The structure of a Decision Tree is composed of several key components:

- Root Node: Represents the entire dataset and is the starting point of the tree.
- Internal Nodes: These nodes represent the attributes or features of the dataset. They act as decision points that split the data into subsets.

- Branches: Each branch emerging from an internal node represents the outcome of a decision made on a feature, leading to further splits or outcomes.
- Leaf Nodes: These are the terminal nodes of the tree that provide the final output, such as a class label for classification tasks or a continuous value for regression.

The Decision Trees rely on impurity measures to decide the best way to split data at each node. Common impurity measures including Gini Index and Entropy, the measures can be represented mathematically as:

- The Gini Index

$$Gini(D) = 1 - \sum_{i=1}^{C} p_i^2 \quad (3.14)$$

Where p_i is the probability of class i in dataset$D$, and C is the number of classes. A lower Gini index indicates a better split and

- The Entropy

$$Entropy(D) = -\sum_{i=1}^{C} p_i log_2(p_i) \quad (3.15)$$

Where p_i represents the proportion of examples of class i in the dataset$D$. Entropy measures the disorder or impurity in the data. A lower entropy value indicates higher purity.

Information Gain (IG) is a metric used to quantify the effectiveness of a split. It measures how much uncertainty (or entropy) is reduced after splitting the data based on a particular feature:

$$IG(D, A) = Entropy(D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} Entropy(D_v) \quad (3.16)$$

Where D_v is the subset of D for which attribute A takes the value$v$. A higher IG indicates a more informative split.

The decision tree algorithm recursively splits the dataset on the feature with the highest Information Gain or lowest Gini index, continuing until a stopping criterion is met. These criteria can include reaching a maximum depth, having a minimum number of samples in a node, or achieving a minimal level of impurity.

A Random Forest is an ensemble learning technique that builds multiple decision trees (often thousands) during training. The final output is determined by aggregating the predictions of individual trees, either through majority voting for classification or averaging for regression.

The Random Forests leverage a technique called bootstrapping, where multiple subsets of the training data are created by sampling with replacement. Each decision tree in the forest is trained on a different bootstrap sample, reducing overfitting by decreasing variance across the model ensemble.

Unlike traditional decision trees that consider all features at each split, Random Forests select a random subset of features to find the best split at each node. This randomness introduces diversity among the trees, further reducing correlation and overfitting.

The prediction of a Random Forest is determined by combining the outputs of all the trees in the ensemble:

- For classification, the mode of the predicted classes is chosen.
- For regression, the average of the predictions is taken:

$$Prediction = \frac{1}{N}\sum_{i=1}^{N} T_i(x) \qquad (3.17)$$

Where T_i is the prediction of the i^{th} tree, and N is the total number of trees in the forest.

The Random Forests can measure the importance of each feature by analysing how much a feature contributes to reducing impurity (measured by Gini index or entropy) across all trees. This is done by calculating the decrease in accuracy or the increase in weighted impurity when a feature is randomly permuted.

Decision Trees offer a straightforward, interpretable method for classification and regression, though they may suffer from overfitting. Random Forests address this issue by aggregating multiple trees, enhancing model robustness and accuracy while providing valuable insights into feature importance. The mathematical principles underlying both techniques, such as impurity measures and ensemble methods, contribute to their widespread use and effectiveness in a variety of predictive modelling tasks.

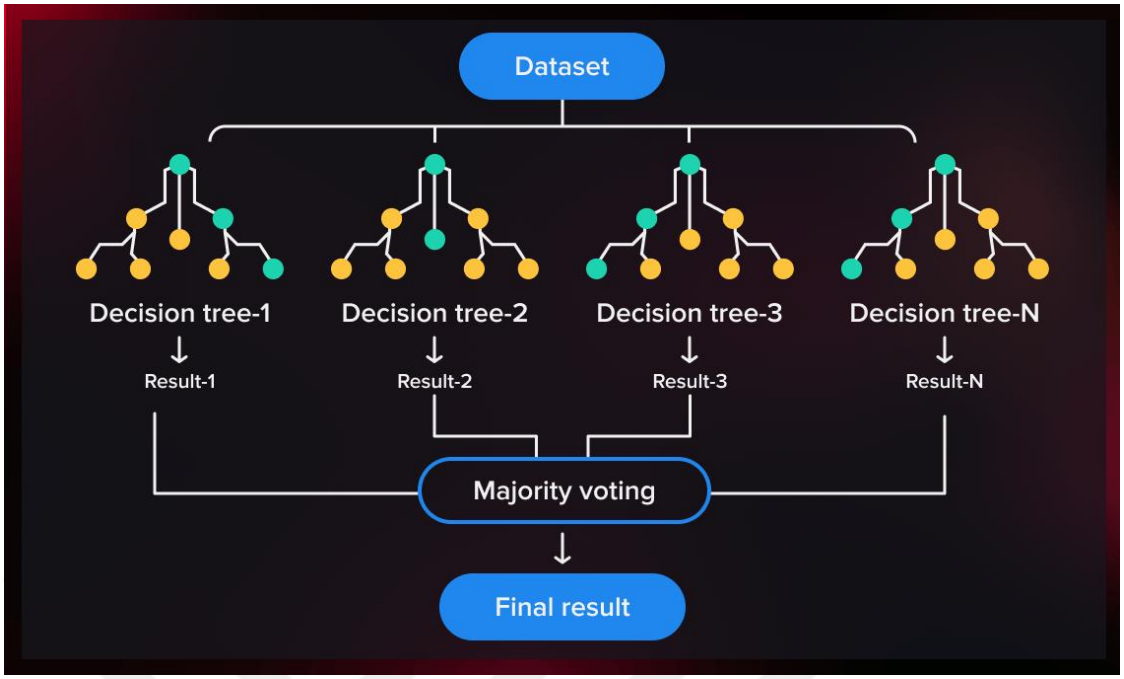

Figure 2.3: Random Forest

3.1.4. Deep Learning Models

Deep learning has gained prominence in human detection and tracking due to its ability to automatically extract features from raw data, thus eliminating the need for extensive manual feature engineering. Several state-of-the-art deep learning models are widely utilized in this research area:

Deep learning is a subset of machine learning that employs neural networks with many layers (deep networks) to model complex data patterns. Here's an overview of the theory and mathematical basis for deep learning models:

At the core of deep learning is the neural network, which consists of interconnected layers of nodes (neurons). Each neuron receives inputs, processes them, and passes the output to the next layer. A typical neural network architecture includes:

- Input Layer: The first layer that receives input data.
- Hidden Layers: Intermediate layers that do not interact with the output directly. There can be multiple hidden layers in a deep network.
- Output Layer: The final layer that produces the desired output (predictions).

Each neuron applies an activation function to its output to introduce non-linearity into the model. Common activation functions include:

- Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (3.18)$$

The sigmoid function maps any input to a value between 0 and 1, often used in binary classification.

- Tanh:

$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (3.19)$$

The tanh function outputs values between -1 and 1, providing stronger gradients than sigmoid in some contexts.

- ReLU (Rectified Linear Unit):

$$ReLU(x) = \max(0, x) \quad (3.20)$$

ReLU is widely used due to its simplicity and efficiency, especially in deep networks, where it mitigates the vanishing gradient problem.

The process of passing the input data through the layers of the neural network to compute the output. For a simple feedforward neural network, the output of a neuron can be calculated as:

$$h = f(W^{T}x + b) \quad (3.21)$$

where, W is the weight vector, x is the input vector, b is the bias, and f is the activation function.

The loss function quantifies how well the neural network's predictions match the actual target values. Common loss functions include:

- Mean Squared Error (MSE) for regression:

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y \quad i - \hat{y}_i)^2 \quad (3.22)$$

- Cross-Entropy Loss for classification:

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \quad (3.23)$$

This loss function is commonly used in classification tasks and measures the dissimilarity between the predicted probability distribution and the true distribution.

Backpropagation is the algorithm used for training neural networks by minimizing the loss function. It involves two main steps:

- Forward Pass: Calculate the output and loss.
- Backward Pass: Compute the gradient of the loss with respect to each weight through the chain rule.
- The weight updates are made using Gradient Descent or its variants (like Adam, RMSprop):

$$W \coloneqq W - \alpha \nabla L \tag{3.24}$$

where α is the learning rate, and ∇L is the gradient of the loss with respect to the weights.

The core of DL is based on Neural networks heavily rely on linear algebra concepts, particularly in operations involving vectors and matrices for representation and transformations of data.

A fundamental optimization algorithm used to minimize the loss function. It iteratively adjusts the parameters (weights and biases) in the direction of the negative gradient of the loss.

The chain rule is utilized in backpropagation to compute the gradient of the loss function with respect to the weights. For a composite function $z = g(f(x))$:

$$\frac{dy}{dx} = \frac{dz}{df} \cdot \frac{df}{dx} \tag{3.25}$$

Techniques such as $L2$ (Ridge) regularization or dropout are used to prevent overfitting by adding a penalty for complex models.

- L2 Regularization:

$$L_{regularized} = L + \lambda \|W\|^2 \tag{3.26}$$

where λ is a hyperparameter controlling the strength of regularization. This adds a penalty proportional to the square of the weights, discouraging overly complex models.

- **Dropout**: Randomly dropping neurons during training to prevent the network from becoming too reliant on any single neuron, thus improving generalization.

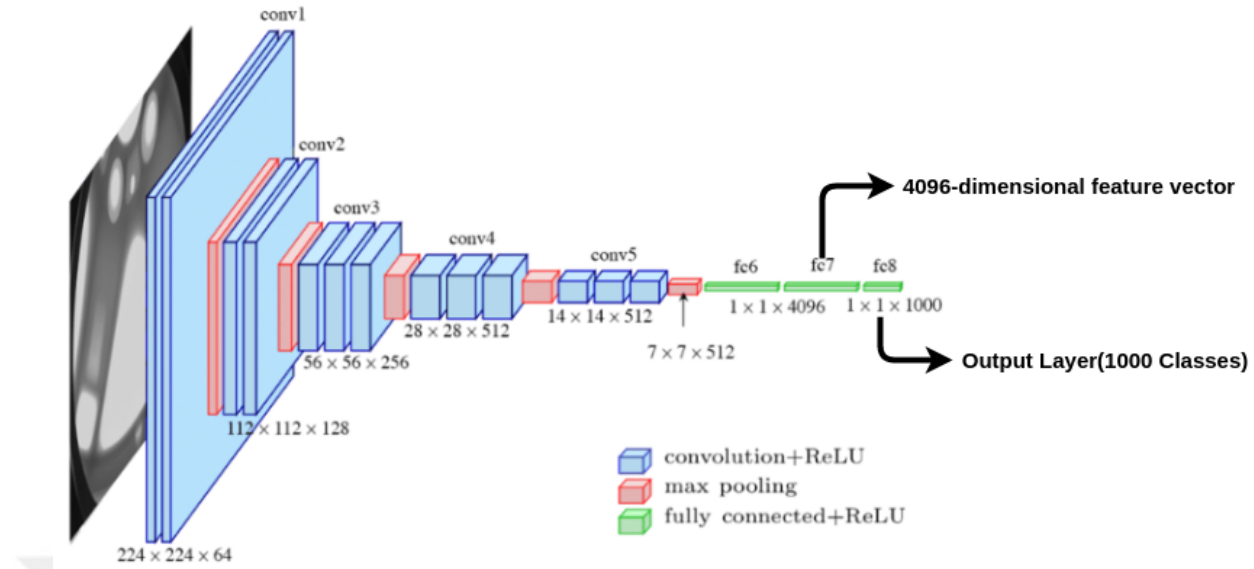

Figure 3.3: Deep Learning Method for Object Detection

3.1.5. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) are specialized for processing grid-like data, such as images. Convolutional layers apply filters to the input to detect features. The convolution operation can be expressed as:

$$(I * K)(x, y) = \sum_m \sum_n I(m, n) K(x - m, y - n) \qquad (3.27)$$

where I is the input, K is the kernel/filter, and $*$ denotes convolution.

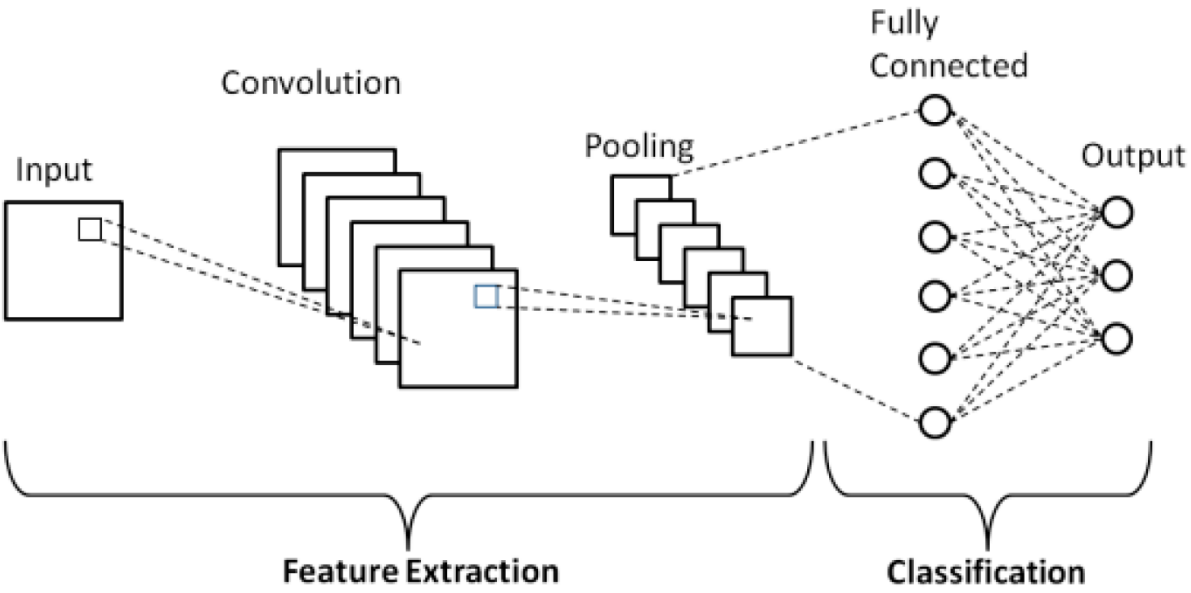

Figure 3.4: Convolutional Neural Network for Image Classification

3.1.6. Recurrent Neural Networks (RNNs)

CNNs are particularly effective in image processing tasks. They employ convolutional layers to extract spatial hierarchies of features and pooling layers to down-sample these features. CNN architectures such as VGGNet and ResNet provide robust backbones for object detection systems (24, 25).

RNNs are designed to handle sequential data (like time series or text). They maintain a hidden state that gets updated as new inputs are processed. RNNs are designed for sequential data, such as time series or text. They maintain a hidden state that is updated as new inputs are processed. The hidden state at time $\boldsymbol{t}$ is given by:

$$h_t = f(W_h h_{t-1} + W_x x_t + b) \qquad (3.28)$$

where h_t is the hidden state at time t, $x_t$is the input at time t, and W_h, W_x are the weight matrices.

Deep learning models leverage the structure of neural networks to learn complex patterns in data through layers of abstraction. The mathematical foundations, including loss functions, optimization algorithms like gradient descent, and special architectures like CNNs and RNNs, enable these models to excel in various tasks, from image and speech recognition to natural language processing and beyond. The non-linear activation functions, combined with

the capacity for learning across many layers, empower deep learning to capture intricate relationships in large datasets.

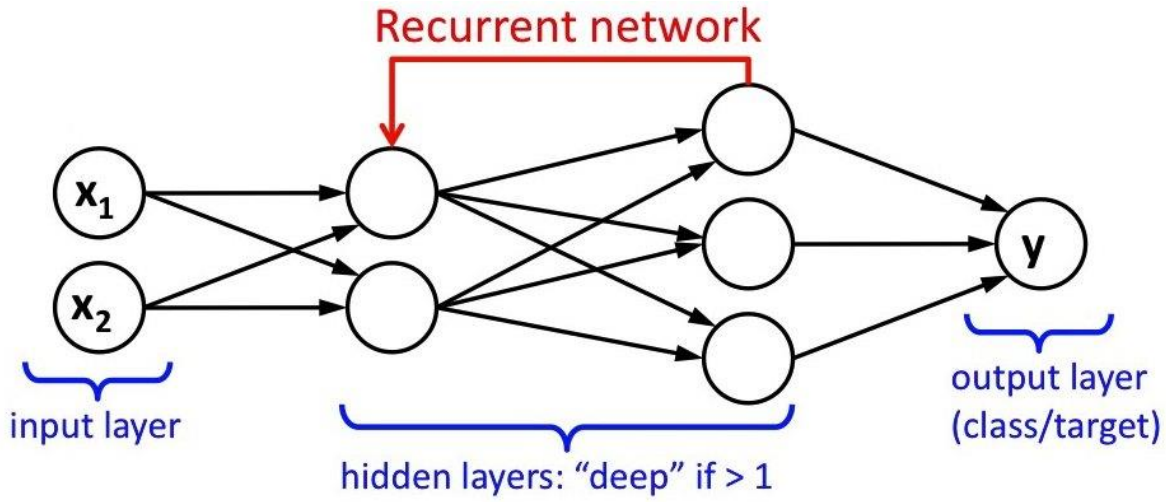

Figure 3.5: Recurrent Neural Network

3.1.7. You Only Look Once (YOLO)

YOLO is a real-time object detection model that treats detection as a single regression problem, predicting bounding boxes and class probabilities directly from full images. It divides the image grid and simultaneously predicts multiple bounding boxes and their class confidences, making it suitable for fast detection in dynamic environments (26).

Only Look Once" (YOLO) is a widely recognized real-time object detection system known for its speed and accuracy. Unlike traditional object detection methods that often rely on separate processes for object classification and bounding box prediction, YOLO treats object detection as a single regression problem. Below is an expanded overview of the theory and mathematical basis behind YOLO:

1. Only Look Once" (YOLO)

   YOLO utilizes a single convolutional neural network (CNN) to simultaneously predict multiple bounding boxes and their associated class probabilities directly from full images. This unified approach allows for rapid processing, setting YOLO apart from methods that use region proposals or sliding windows.

2. Image Grid Division:

The input image is divided into an $S * S$ grid. Each grid cell is responsible for detecting objects whose center lies within the cell. Each cell predicts a fixed number of bounding boxes, including the object class probabilities and confidence scores.

1. Bounding Box Prediction:
   - For each grid cell, YOLO predicts B bounding boxes, each characterized by:
     - (x, y): Center coordinates of the bounding box relative to the grid cell.
     - w, h: Width and height of the bounding box.
     - Confidence score c:A measure that combines the probability of an object being present and the accuracy of the bounding box.
     - Class probabilities: For all possible classes.
2. Confidence Score:

   The confidence score is a product of the probability that a bounding box contains an object and the Intersection over Union (IOU) between the predicted box and the ground truth. It is mathematically expressed as:

$$Confidence = P(Object) * IOU_{pred}^{truth} \quad (3.29)$$

   Where, $P(Object)$ represents the probability of the presence of an object, and IOU measures the overlap between the predicted and actual bounding boxes.

3. Final Output Representation:

   The final output for each grid cell combines the object probability, IOU, and class probabilities, expressed as:

$$P_{i,j} = P_{i,j}^{object} * IOU_{i,j}^{pred} * (C_{i,j}^{1}, C_{i,j}^{2}, \ldots., C_{i,j}^{c}) \quad (3.30)$$

   Where $C_{i,j}^{c}$ denotes the conditional probability of class c given that an object is detected.

YOLO is based on five functions they are the Comprehensive Loss Function, Localization Loss, Classification Loss, and Intersection over Union (IOU), Real-time Processing Capabilities.

- Comprehensive Loss Function: YOLO optimizes a single loss function that integrates localization loss, confidence loss, and classification loss. The loss function can be represented as:

$$L = \sum_{i=0}^{S^2} \sum_{j=0}^{B} loss_{coord} + \lambda_{noobj} \sum_{j=0}^{B} loss_{noobj} + \lambda_{cls} \sum_{l=0}^{C} loss_{cls} \quad (3.31)$$

where $loss_{coord}$) quantifies errors in predicting the bounding box coordinates, $loss_{noobj}$ penalizes predictions in grid cells that do not contain objects, $loss_{cls}$ accounts for misclassifications, and λ_{noobj} and λ_{cls} are balancing coefficients for the loss components.

- Localization Loss: Localization loss is typically computed using Mean Squared Error (MSE) between predicted and ground truth bounding box coordinates:

$$loss_{coord} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} [\![(x_{i,j}^{pred} - x_{i,j}^{truth})^2 + (y_{i,j}^{pred} - y_{i,j}^{truth})^2 + (w_{i,j}^{pred} - w_{i,j}^{truth})^2 + (h_{i,j}^{pred} - h_{i,j}^{truth})^2]\!] \quad (3.32)$$

- Classification Loss: The classification loss is calculated using Cross-Entropy Loss, penalizing the model for incorrect class predictions:

$$loss_{cls} = -\sum_{i=0}^{S^2} \sum_{c=0}^{C} y_{i,j}^{c} \log(\hat{y}_{i,j}^{c}) \quad (3.33)$$

Where $y_{i,j}^{c}$ is the ground truth label for class c and$\hat{y}_{i,j}^{c}$) is the predicted probability.

- Intersection over Union (IOU): IOU, a critical metric for evaluating the accuracy of predicted bounding boxes, is calculated as:

$$IOU = \frac{Area\ of\ Intersection}{Area\ of\ Union} \quad (3.34)$$

High IOU values indicate better overlap between predicted and ground truth boxes.

- Real-time Processing Capabilities: YOLO's architecture enables real-time object detection by allowing the model to predict all bounding boxes and class probabilities in a single forward pass. This results in significantly faster inference compared to methods that require multiple passes or post-processing stages.

YOLO revolutionizes object detection by offering a unified approach that predicts bounding boxes and class probabilities simultaneously from full images. This enables real-time processing, making it highly efficient and suitable for various computer vision applications. The model's underlying mathematical framework—comprising a comprehensive loss function, grid-based predictions, and optimization techniques—ensures its effectiveness in detecting objects quickly and accurately. YOLO's blend of speed and precision has made it a powerful tool in the field of object detection.

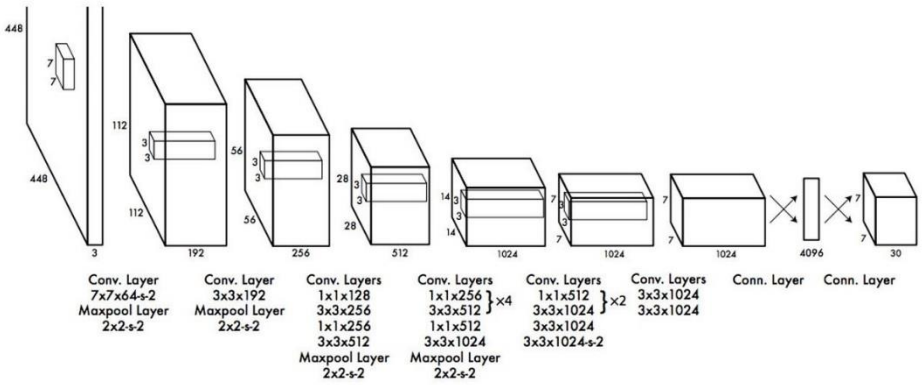

Figure 3.6: The Architecture

our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1*1 convolutional layers reduce the feature space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224*224 input image) and then double the resolution for detection.

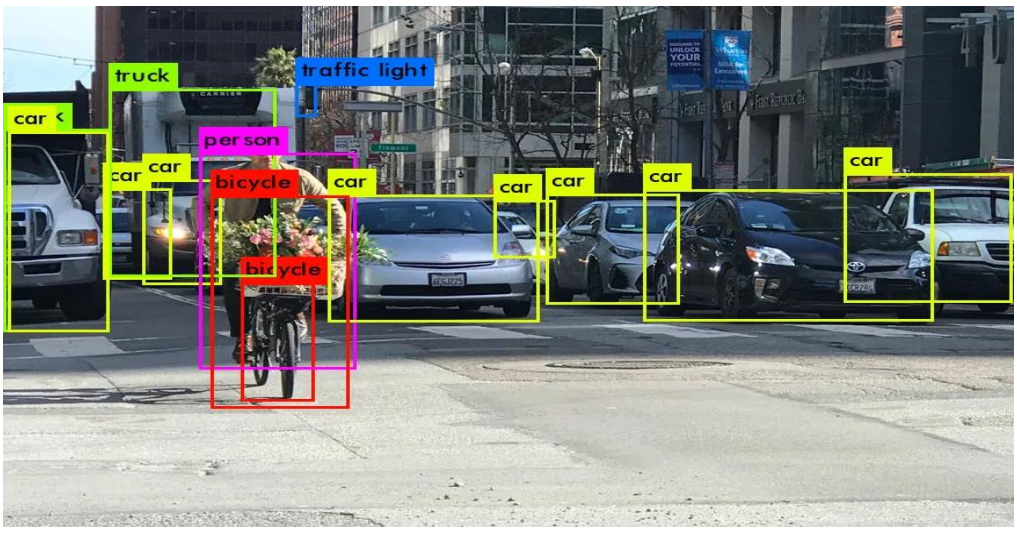

Figure 3.7: Yolo Object Detection

3.1.8. Single Shot MultiBox Detector (SSD)

SSD is another single-shot detection technique that combines predictions from different feature maps of various sizes, allowing it to detect objects of varying scales. Its speed and accuracy make it suitable for real-time applications (28).

The Single Shot MultiBox Detector (SSD) is a deep learning model designed for object detection, striking a balance between speed and accuracy. SSD is an evolution over earlier object detection methods and is particularly noted for its efficiency and effectiveness in real-time scenarios. Below is an expanded overview of the theoretical framework and mathematical foundation of SSD.

The SSD model is designed to detect objects within images using a single deep neural network. It efficiently identifies multiple objects of varying classes in one forward pass.

The term "single shot" refers to SSD's capability to predict both bounding boxes and class scores in one evaluation of the network. This contrasts with methods that require multiple stages, such as region proposal networks.

SSD predicts a fixed set of bounding boxes, known as "default boxes," at each feature map location. These boxes vary in aspect ratio and scale, enabling the model to detect objects of different sizes and shapes.

SSD leverages feature maps from multiple layers within the network to make detections. Lower layers focus on detailed, fine-grained features, while higher layers capture more abstract semantic information. This multi-scale approach is essential for detecting objects of varying sizes.

At each location in the feature maps, SSD defines several default boxes with predefined aspect ratios and scales. For each default box, the model predicts Offset adjustments for the box coordinates (dx, dy for the center, dw, dh for width and height), and Class scores that indicate the likelihood of an object being within the box.

SSD typically utilizes a base network like VGG16 or ResNet, followed by additional convolutional layers to create feature maps at multiple scales. The base network is responsible for extracting features from the input image, while the subsequent layers handle detection tasks.

For each cell in the feature map, SSD defines multiple default boxes b_i with various aspect ratios and scales. These boxes are represented as:

$$b_i = (x, y, w, h) \tag{3.35}$$

where, x, y denotes the center of the box, and w, h represent the width and height of the box.

The SSD generates two types of predictions for each default box:

- Regression Predictions: Offset values (t_x, t_y, t_w, t_h) to refine the default box coordinates:

$$\hat{x} = x + t_x \cdot w \tag{3.36}$$

$$\hat{y} = y + t_y \cdot h \tag{3.37}$$

$$\hat{w} = w \cdot e^{t_w} \tag{3.38}$$

$$\hat{h} = h \cdot e^{t_h} \tag{3.39}$$

- The Class Probability Predictions: Scores for each class (C_k):

$$C_k = P(class_k \mid b_i) \tag{3.40}$$

where $(class_k \mid b_i)$ is the probability that class k is present in the bounding box.

The loss function in SSD is composed of:

- Localization Loss: This measures the difference between predicted and ground truth bounding box coordinates, typically using Smooth L1 loss or L2 loss:

$$L_{loc} = \sum_i^N \sum_j^M L_{smooth}(t_j^i, \hat{t}_j^i) \tag{3.41}$$

where N is the number of default boxes, and M is the number of objects per image.

- Confidence Loss: This assesses the accuracy of the class predictions using softmax cross-entropy:

$$L_{cls} = -\sum_i \sum_{c \in C} y_{i,c} \log(\hat{y}_{i,c}) \tag{3.42}$$

where $y_{i,c}$ is the ground truth label, and $\hat{y}_{i,c}$ is the predicted probability for class c.

The overall loss for SSD is a weighted sum of the localization and classification losses:

$$L = L_{loc} + \lambda * L_{cls} \tag{3.43}$$

where λ is a hyperparameter that balances the localization and classification components of the loss.

To refine the predictions, SSD applies Non-Maximum Suppression (NMS) after calculating bounding box coordinates and class scores. NMS selects the bounding box with the highest confidence score and removes other boxes that overlap significantly with it, determined by an Intersection over Union (IoU) threshold.

The Single Shot MultiBox Detector (SSD) is a robust and efficient approach to object detection, offering real-time performance. By utilizing a single network to simultaneously predict bounding boxes and class scores at multiple scales and incorporating default boxes with various aspect ratios, SSD delivers high accuracy at impressive speeds. Its mathematical foundation, including the combination of localization and classification loss functions and the application of Non-Maximum Suppression, makes SSD a versatile tool for a wide range of computer vision tasks.

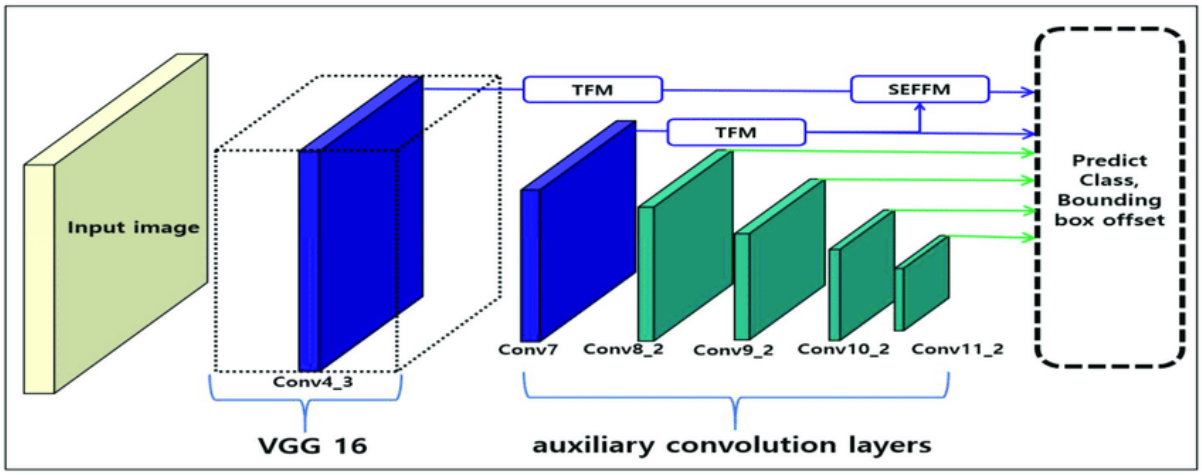

Figure 3.8: Object Detection with SSD

3.1.9. Data Association Algorithms

Data association is critical in multi-target tracking, as it links object detections across consecutive frames while maintaining identities. Various algorithms facilitate this process:

Data association algorithms are fundamental in fields such as multi-target tracking, robotics, and computer vision. Their primary function is to match observed measurements, like detected objects, with existing entities, such as tracked objects, over time. These algorithms are indispensable for maintaining identity consistency, especially when tracking multiple objects through sequential data.

The data association problem involves connecting a set of observed measurements (e.g., detected bounding boxes in images or positional readings from sensors) to existing object tracks. The objective is to minimize tracking uncertainty by accurately linking each observation to the correct object.

The output of a data association algorithm is an assignment of measurements to existing tracks. This assignment is made in a way that either minimizes the total cost or maximizes the total likelihood of correct associations.

Single Hypothesis: Assumes a one-to-one correspondence between each measurement and a single object, which is the most common approach.

Multiple Hypotheses: Considers various possible associations, allowing for uncertainty in which measurement corresponds to which object.

In many data association methods, a cost matrix is constructed to represent the distances or costs between each measurement and each existing track. The cost might reflect the Euclidean distance between the predicted position of a track and the actual measurement. Assuming M to be the number of measurements and T being the number of tracks, the cost matrix C is defined as:

$$C_{i,j} = d(m_i, t_j) \tag{3.44}$$

where $d(m_i, t_j)$ represents the cost of associating measurementm_i with track t_j.

The association problem can be formulated as an assignment problem, where the goal is to minimize the overall cost:

$$Minimize \sum_{i=1}^{M} \sum_{j=1}^{T} C_{i,j} x_{i,j}$$

subject to:

$$\sum_{j=1}^{T} x_{i,j} \leq 1, \forall_i \quad (3.45)$$

$$\sum_{j=1}^{T} x_{i,j} \leq 1, \forall_j \quad (3.46)$$

where $x_{i,j}$ is a binary variable that indicates whether measurement m_i is associated with track t_j.

The algorithms used for association can be employed to solve the data association problem, including, the include:

- Hungarian Algorithm: This algorithm solves the assignment problem efficiently in polynomial time ($O(n^3)$). It works by transforming the cost matrix and finding the optimal assignment that minimizes the total cost.
- Joint Probabilistic Data Association (JPDA): JPDA incorporates probabilities for each possible association, accounting for uncertainties in both the measurements and the existing tracks. It calculates the likelihood of each association based on the distribution of measurements.
- Multiple Hypothesis Tracking (MHT): MHT generates multiple hypotheses for how measurements could be associated with tracks over time. These hypotheses are evaluated, and the best one is chosen based on criteria such as likelihood.
- Global Nearest Neighbor (GNN): GNN finds a one-to-one assignment of measurements to tracks that minimizes the overall cost. While similar to the Hungarian method, it often employs a heuristic approach, processing measurements sequentially.

Probabilistic Data Association (PDA), predicts the probability of association based on the likelihood that a particular measurement originates from a specific track. The association is then performed based on these probabilities.

Likelihood Function, is used for observing a measurement given a predicted state is often modelled using Gaussian distributions:

$$P(z_k \mid x_i) = \frac{1}{\sqrt{(2\pi)^n \det(\boldsymbol{\Sigma})}} e^{-\frac{1}{2}(z_k - Hx_i)^T (z_k - Hx_i)} \quad (3.47)$$

where z_k is the measurement, x_i is the predicted state, H is the measurement model, and $\mathbf{\Sigma}$ is the measurement noise covariance.

Data association algorithms are critical in multi-target tracking, as they enable the accurate matching of observed measurements to existing object identities while minimizing ambiguity and uncertainty. The mathematical foundations of these algorithms involve constructing cost matrices, solving assignment problems, and assessing the likelihood of associations under uncertain conditions. Techniques like the Hungarian algorithm, JPDA, MHT, and probabilistic models provide robust solutions for the complex challenges faced in dynamic environments. These algorithms have widespread applications in robotics, computer vision, and surveillance, ensuring reliable multi-object tracking over time.

3.1.10. Kalman Filter

The Kalman filter estimates the state of a moving object over time, based on previous states and measurements. It is particularly suited for linear motion models, providing a recursive solution to maintain trajectory estimations (29).

The Kalman Filter is a robust recursive algorithm widely utilized for estimating the state of a dynamic system from a series of noisy observations. This powerful tool is essential in fields such as control systems, robotics, and computer vision, where accurate real-time estimation is critical. The Kalman Filter operates in two primary stages: prediction and update, forming a cycle that continuously refines the estimate of the system's state.

The Kalman Filter relies on a state-space representation to model dynamic systems. This representation uses linear equations to describe both the evolution of the system's state over time and the relationship between the state and the measurements.

The state of the system at a given time k is denoted as x_k, representing all the variables necessary to describe the system at that moment.

The evolution of the system's state is governed by the state transition equation:

$$x_k = F_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \qquad (3.48)$$

where, F_{k-1}) is the state transition matrix, which models how the state evolves from time $k-1$) to k, B_{k-1} is the control-input model that applies the control vector u_{k-1} to influence the state, and

w_{k-1} represents the process noise, assumed to be Gaussian with zero mean and covariance Q (i.e., $w_{k-1} \sim N(0, Q)$).

The relationship between the system state and the observed measurements is described by the measurement equation:

$$z_k = H_k x_k + v_k \tag{3.49}$$

where, H_k is the observation matrix, mapping the true state space to the observed space and v_k is the measurement noise, which is also assumed to be Gaussian with zero mean and covariance R (i.e., $v_k \sim N(0, R)$).

The prediction phase involves estimating the system's next state and the associated uncertainty (error covariance) before the actual measurement is made. The state prediction:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_{k-1} u_{k-1} \tag{3.50}$$

The error covariance prediction is represented as:

$$P_{k|k-1} = F_{k|k-1} \hat{x}_{k-1|k-1} F_{k-1}^T + Q \tag{3.51}$$

where, $\hat{x}_{k|k-1}$is the predicted state estimate, and $P_{k|k-1}$ is the predicted error covariance, indicating the uncertainty associated with the prediction.

The update phase refines the predicted state estimate using the new measurement. The innovation (residual) calculation is:

$$y_k = z_k + H_k, \hat{x}_{k|k-1} \tag{3.52}$$

The innovation y_k represents the difference between the actual measurement and the predicted measurement, indicating how much the prediction deviates from reality.

The innovation covariance is defined as:

$$S_k = H_k P_{k|k-1} H_k^T + R \tag{3.53}$$

This step calculates the uncertainty associated with the innovation, taking into account both the predicted error covariance and the measurement noise. The Kalman gain is calculated as:

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{3.54}$$

The Kalman Gain K_k determines how much weight should be given to the innovation in updating the state estimate. A higher gain gives more importance to the measurement, while a lower gain relies more on the prediction. The State Estimate Update is given by:

$$\hat{x}_{\mathrm{k|k}} = \hat{x}_{\mathrm{k|k-1}} + K_k y_k \tag{3.55}$$

The state estimate is updated by combining the predicted state with the weighted innovation, yielding a more accurate estimate. The Error Covariance Update is calculated as:

$$P_{\mathrm{k|k}} = (I - K_k H_k) P_{\mathrm{k|k-1}} \tag{3.56}$$

The error covariance is updated to reflect the reduction in uncertainty after incorporating the measurement, where I is the identity matrix.

The Kalman Filter's recursive nature and reliance on linear algebra, probability theory, and Gaussian distributions make it an ideal tool for real-time state estimation in noisy environments. Beyond its classical applications in navigation and control systems, the Kalman Filter has been extended and adapted to non-linear systems through variants like the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF). These variants have further expanded the filter's applicability to more complex scenarios, such as in robotics for sensor fusion and in finance for tracking and predicting market trends. The Kalman Filter's versatility, coupled with its strong mathematical foundation, ensures its continued relevance and utility across diverse domains.

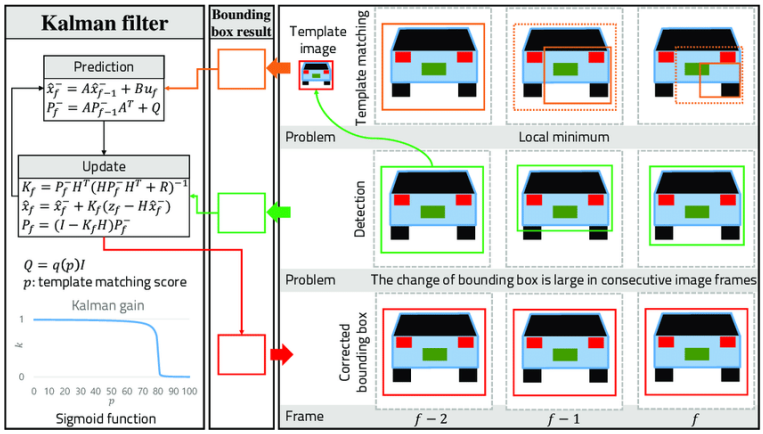

Figure 3.9: Object detection combining YOLOv3 and template-matching with a Kalman filter

3.1.11. Hungarian Algorithm

The Hungarian algorithm is an efficient method for solving the assignment problem, matching detected objects to tracked identities based on minimizing the cost associated with mismatches. This algorithm ensures optimal object assignment while accounting for spatial proximity (30).

The Hungarian Algorithm is a well-known combinatorial optimization technique designed to address the assignment problem, where the goal is to optimally assign n tasks to n agents in a way that either minimizes total cost or maximizes total profit. This algorithm is particularly effective and is implemented using weighted bipartite graphs.

The assignment problem involves determining the best way to assign tasks to agents, represented by a cost matrix C where $c_{i,j}$ indicates the cost of assigning agent j to task i. The objective is to identify a perfect matching (a one-to-one assignment) that results in the lowest possible total cost:

$$minimize \sum_{i=1}^{n} C_{i\pi(i)} \tag{3.57}$$

Where, $\pi(i)$ denotes the assignment function mapping task i to agent $\pi(i)$.

The problem can be visualized as a bipartite graph, with one set of vertices representing tasks and the other representing agents. The edges between vertices are weighted according to the assignment costs. The Hungarian Algorithm is one method used to find the optimal assignment by determining the maximum matching in the weighted bipartite graph. It follows a systematic approach to ensure the most cost-effective pairing of tasks and agents.

The Hungarian Algorithm proceeds through several critical stages:

- Initial Matrix Setup: If the cost matrix C isn't square, it is converted into one by adding dummy rows or columns with zero costs, ensuring the matrix is balanced with an equal number of tasks and agents.
- Row Reduction: For each row in the matrix, subtract the smallest value in that row from all the other elements in that row. This operation ensures that each row contains at least one zero.
- Column Reduction: Similarly, subtract the smallest value in each column from all the other elements in that column. After this step, each column will also contain at least one zero, facilitating the next steps.

- Zero Coverage: Cover all zeros in the modified matrix using the fewest number of lines (either horizontal or vertical). This step can be performed using methods such as the Minimum Covering Lines algorithm.
- Checking for Optimality: If the number of lines used to cover the zeros matches n (the number of tasks and agents), the matrix is in an optimal state, and the algorithm can terminate successfully with a solution. If not, the algorithm proceeds to the next step.
- 6. Matrix Adjustment: If the optimality condition is not met, adjust the matrix as follows:
    - Identify the smallest uncovered element m in the matrix. Subtract m from all uncovered elements, and add m to all elements covered twice (at the intersections of lines).
    - Repeat the zero coverage and optimality check steps until the condition is satisfied.
- Deriving the Assignment: Once the optimal solution is identified, a perfect matching is found in the matrix, which corresponds to the minimal cost assignment.

The Time Complexity: The Hungarian Algorithm operates with a time complexity of $O(n^3)$, making it an efficient solution for assignment problems of moderate size.

The practical applications of this algorithm is utilized in diverse fields such as operations research, job scheduling in manufacturing, economic market allocation, and image matching in computer vision.

The Hungarian Algorithm offers a robust and systematic method for solving assignment problems. By transforming the cost matrix through a series of reductions and adjustments, the algorithm eventually identifies an optimal solution. Its foundation in combinatorial optimization and graph theory enables it to efficiently find the best matching in weighted bipartite graphs, making it a versatile tool for a wide range of applications.

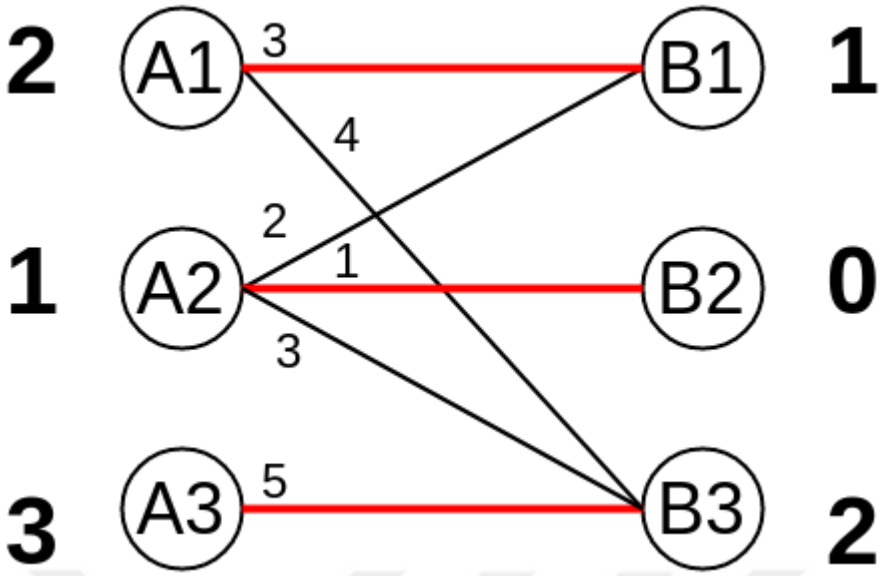

Figure 3.10: Hungarian Algorithm

3.1.12. Deep SORT

Deep SORT enhances the traditional SORT algorithm by incorporating appearance features from a deep learning model, allowing for improved data association even in crowded environments where occlusions frequently occur. The integration of deep features helps in distinguishing between similar-looking targets (31).

Deep SORT (Deep Simple Online and Real-time Tracking) extends the original SORT (Simple Online and Real-time Tracking) algorithm, which is used for tracking multiple objects in video sequences. Deep SORT enhances SORT by integrating deep learning-based feature extraction, which improves its ability to manage occlusions and re-identify objects effectively. Here's an overview of the theory and mathematical principles behind Deep SORT.

- Multi-Object Tracking (MOT): Multi-object tracking aims to detect and follow multiple objects in a video over time, ensuring that each object's identity remains consistent throughout the sequence.
- Detection and Tracking: Deep SORT functions in two main phases:
    - Detection: A deep learning-based object detector (such as YOLO or Faster R-CNN) is employed to identify objects and their bounding boxes in each video frame.

- Tracking: These detections are linked across successive frames to maintain distinct identities for each object.
- State Representation: Each object being tracked is represented by a state vector that captures its position and velocity:

$$X = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix} \tag{3.58}$$

Here, (x, y) indicates the position, while (v_x, v_y) represents the object's velocity.

- Kalman Filter: The Kalman Filter is employed in Deep SORT to predict the state of an object. The filter's equations help forecast the next state based on the current state and new observations. The prediction process is modeled within a state-space framework. The general prediction equation is:

$$x_{\mathrm{k|k-1}} = F x_{\mathrm{k-1|k-1}} + B u_{k-1} + w \tag{3.59}$$

where F represents the state transition model, B is the control-input model, u_{k-1} is the control vector, and w is the process noise.

- Appearance Features: Deep SORT goes beyond SORT by incorporating deep convolutional neural networks (CNNs) to extract appearance features, represented by feature vectors f, from detected objects. These features are crucial for tracking objects during occlusions and avoiding identity switches.
- Data Association: The core of Deep SORT lies in its data association process, where Kalman Filter predictions and appearance features are combined to match detections across frames. The Hungarian algorithm is used to solve the assignment problem using a cost matrix that combines spatial and appearance information:
    - Cost Matrix: The cost matrix \(C \) incorporates both the position-based cost and the appearance-based cost:

$$C_{i,j} = \alpha \cdot d_{pos}(i,j) + \beta \cdot d_{app}(i,j) \tag{3.60}$$

where, $d_{pos}(i,j)$ denotes the spatial cost (typically the Euclidean distance between predicted and detected bounding boxes), $d_{app}(i,j)$ represents the appearance cost (calculated using

cosine similarity or L2 distance), α) and β are weights that determine the relative importance of position and appearance in the cost function.

Kalman Filter Prediction, utilizes the object's state is predicted based on its current velocity and previous position:

$$\hat{x}_{\mathrm{k|k-1}} = F \cdot x_{\mathrm{k|k-1}} \tag{3.61}$$

The state is updated using the measurement z as:

$$\boldsymbol{x}_{\mathrm{k|k-1}} = \widehat{\boldsymbol{x}}_{\mathrm{k|k-1}} + \mathbf{K} \cdot (\mathbf{z} - \mathbf{H} \cdot \widehat{\boldsymbol{x}}_{\mathrm{k|k-1}}) \tag{3.62}$$

where **K** is the Kalman gain, determined by the prediction uncertainty and measurement noise.

Feature Extraction in Deep SORT uses a CNN to extract a feature vector **f** from the bounding box of a detected object:

$$\mathrm{f} = \mathrm{CNN}(\mathrm{ROI}(\mathrm{z}) \tag{3.63}$$

Appearance Cost Calculation is the similarity between appearance features can be measured using L2 distance or cosine similarity:

$$d_{app}(i,j) = \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_2^2 \tag{3.64}$$

or

$$d_{app}(i,j) = 1 - \frac{\mathbf{f}_i \cdot \mathbf{f}_j}{\|\mathbf{f}_i\|\|\mathbf{f}_j\|} \tag{3.65}$$

Deep SORT effectively combines traditional tracking methodologies with deep learning techniques to achieve robust multi-object tracking. By utilizing the Kalman Filter for state prediction and incorporating deep learning-based appearance features, Deep SORT can reliably track objects even in challenging scenarios such as occlusions and identity changes. Its mathematical framework is grounded in state-space modeling, cost matrix optimization, and feature extraction, making it a powerful tool for video analysis in fields like surveillance and autonomous driving.

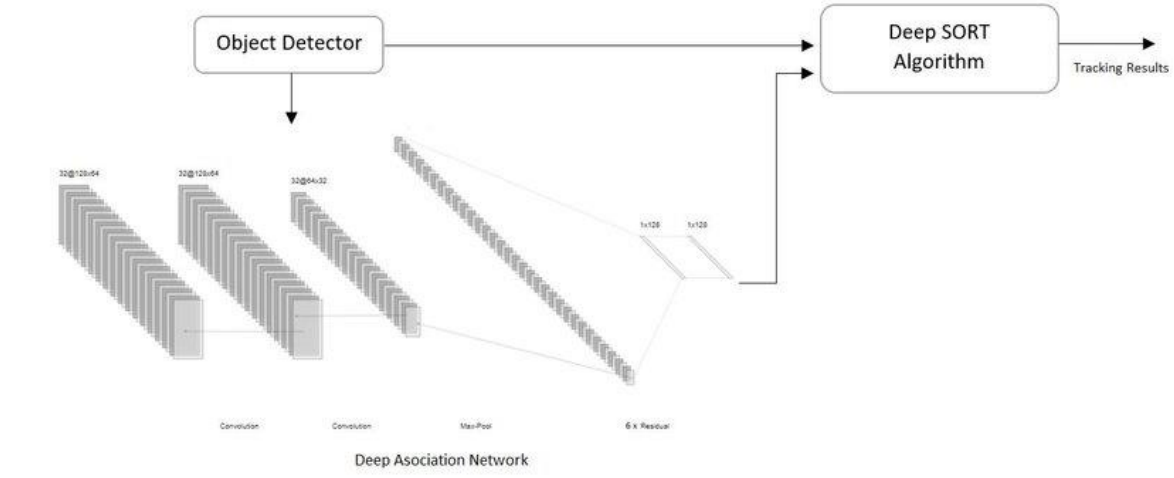

Figure 3.11: Deep Sort Architecture

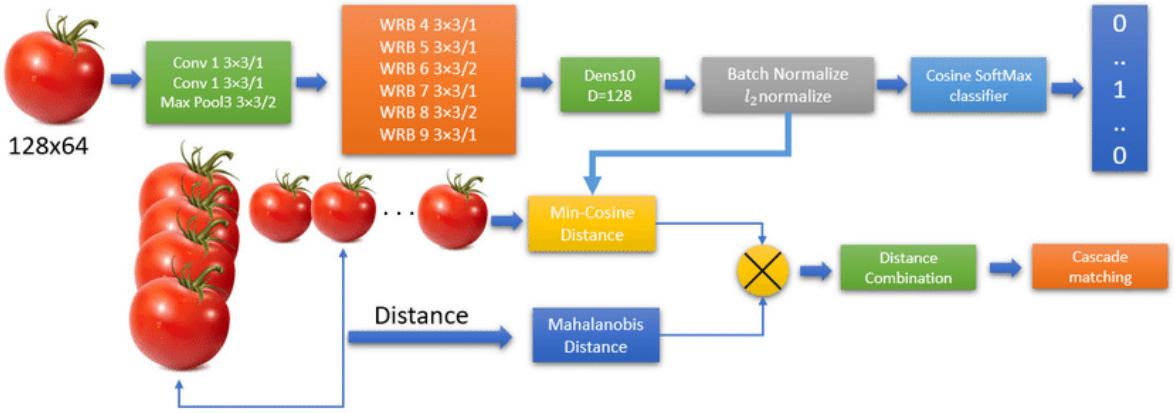

Figure 3.12: Deep Sort Algorithm

3.2. RE-IDENTIFICATION TECHNIQUES

Re-identification focuses on recognizing individuals who may move between different camera views. This capability is essential for maintaining identity across the system:

- Appearance-Based Re-ID: Re-ID methods utilize deep learning to extract distinctive appearance features from frames, utilizing architectures such as Triplet Networks or Siamese Networks that learn to differentiate between individuals based on visual characteristics (32).

- Contextual Information: Incorporating contextual data, such as spatial layout and behavioral patterns, can also enhance re-identification processes. Techniques that assess interactions and proximity to known targets are being researched to improve identity retention across scenes (33).

3.3. PERFORMANCE EVALUATION METRICS

Evaluating the performance of detection and tracking systems is critical to understanding their efficacy. Important metrics include:

- Multiple Object Tracking Accuracy (MOTA): MOTA provides a comprehensive score reflecting the tracking accuracy by accounting for false positives, false negatives, and mismatches (34).
- Multiple Object Tracking Precision (MOTP): MOTP assesses the precision of the predicted bounding boxes, evaluating the average distance between predicted and ground truth bounding boxes.
- ID F1 Score: The ID F1 score measures the quality of the object identity maintenance across time, examining the balance between precision and recall for unique tracked identities (35).

3.4. APPLICATIONS OF MACHINE LEARNING

Machine learning algorithms have found applications across diverse fields, showcasing their adaptability and effectiveness in solving complex problems. Notably, this extends to medical imaging and emerging quantum-enabled machine learning algorithms.

- Medical Imaging: Machine learning, particularly deep learning, is transforming the field of medical imaging. These techniques are employed to analyze medical images, enabling early diagnosis of diseases such as cancer, diabetic retinopathy, and cardiovascular conditions. CNNs have been widely used to automatically segment and classify images, enhancing the accuracy and efficiency of image interpretation in radiology (36, 37)(38, 39). Algorithms can learn from vast amounts of data, leading to improved outcomes in diagnostic imaging and personalized medicine.
- Medical Diagnosis**:** In the field of medical diagnosis, machine learning techniques, particularly deep learning, have become pivotal in enhancing diagnostic accuracy and efficiency. Algorithms can analyze a variety of medical images, including X-rays, MRIs, and CT scans, to assist healthcare professionals in diagnosing diseases like

cancer, diabetic retinopathy, and cardiovascular conditions. For instance, CNNs are widely employed to identify abnormalities, with studies demonstrating that machine learning algorithms can achieve diagnostic performance comparable to that of human.

Radiologists (36, 37)(38, 39)(40, 41)(42, 43). By leveraging large datasets and advanced classification techniques, machine learning can improve diagnostic speed and aid in personalized treatment plans based on predictive analytics.

- Quantum-Enabled Machine Learning: Quantum computing represents a paradigm shift in computational power, which opens new avenues for machine learning. Quantum machine learning algorithms leverage quantum bits (qubits) to perform computations that would be infeasible for classical machines. Techniques such as quantum support vector machines and quantum neural networks promise improved speed and efficiency in processing large datasets. Research is ongoing into how quantum-enhanced algorithms can solve optimization and classification problems faster than their classical counterparts, potentially revolutionizing fields such as finance, drug discovery, and complex system simulations (38, 42)(43).

3.5. CONCLUSION

This chapter has laid the groundwork for understanding various machine learning methodologies that form the backbone of this thesis's approach to multi-target human detection and tracking. By leveraging the power of deep learning models alongside robust data association techniques, the proposed system aims to enhance tracking accuracy and maintain effective identity recognition in challenging scenarios. The inclusion of machine learning applications in diverse fields, particularly medical imaging and quantum computing, illustrates the broad impact of these technologies in advancing various domains. Subsequent chapters will detail the methodologies' implementation and the results obtained from extensive experimentation.

3.6.METHODOLOGY

3.6.1. Introduction

This chapter outlines the methodology employed in this research to develop a multi-target human detection and tracking system utilizing machine learning techniques. It details the experimental design, data collection procedures, system architecture, algorithm selection, and evaluation metrics. The methodology is structured to ensure a robust framework for effectively recognizing and tracking multiple human subjects across various camera inputs.

3.6.2. Experimental Design

The goal of this research is to create a system that can detect and track multiple individuals in real-time using input from multiple camera sources. The experimental design consists of the following key components:

- Objective Definition: The primary objective is to achieve high detection and tracking accuracy while maintaining real-time processing capabilities. A secondary objective is to recognize and categorize newly detected individuals as sub persons of interest based on their association with a main person of interest.
- System Requirements: The system is designed to operate within a defined environment, which includes crowded spaces such as public areas, schools, or shopping malls, where individuals are frequently moving and interacting.

3.6.3. Data Collectıon

To train and evaluate the model, diverse datasets were collected to cover a range of scenarios in human detection and tracking:

- Data Sources: The study utilized publicly available datasets, including the Multi-Object Tracking (MOT) datasets, which provide annotated sequences for various tracking scenarios. Additionally, synthetic data generation methods were employed to simulate crowded environments and occlusions, further enriching the training set.
- Annotation: The datasets were annotated with bounding box information for each individual, including identity labels for multi-target scenarios. Annotation tools such as VGG Image Annotator (VIA) were used to ensure accurate labelling.
- Pre-processing: The images and videos were pre-processed to standardize input sizes and enhance feature extraction. Techniques included resizing images, normalizing pixel

values, and augmenting data through transformations such as rotations, flipping, and colour adjustments to increase the robustness of the models.

3.6.4. System Architecture

The architecture of the proposed human detection and tracking system consists of several critical modules:

- Input Module: This module captures video streams from multiple cameras and consolidates them into a unified frame for processing.
- Detection Module: State-of-the-art models (e.g., YOLO and Faster R-CNN) are deployed for human detection. These models are trained to output bounding boxes around detected individuals in each frame, providing pixel coordinates and confidence scores.
- Tracking Module: Utilizing Kalman filtering for motion prediction, this module links frames together, maintaining identities across video sequences. The tracker's performance is enhanced using data association techniques, specifically employing the Hungarian algorithm and Deep SORT, to ensure accurate tracking even when individuals occlude each other.
- Sub Person Identification Module: When a new individual is detected, the system assesses their association with existing tracked individuals. If they qualify as a sub person of interest, they are linked to a main person of interest based on contextual information, such as proximity and behavioural patterns.

3.6.5. Algorithm Selection

The selection of algorithms for the various modules was a critical component of the methodology:

- Human Detection Algorithms: YOLOv4 was selected for its balance of speed and accuracy in detecting objects in real-time, while Faster R-CNN was also integrated into the system for scenarios where higher accuracy is demanded at the expense of processing speed.
- Tracking Algorithms: The traditional Kalman filter was chosen for motion estimation due to its effective performance in linear tracking scenarios, while Deep SORT was implemented for enhanced data association capabilities. The integration of appearance

features into the tracking process allows for distinguishing between similar-looking individuals.

- Sub Person Recognition Algorithm: A classification model based on deep learning techniques was designed to evaluate and classify detected individuals as sub persons of interest using features obtained from the tracking module. Techniques utilized include Siamese networks to compare features and classification networks to assess behaviour consistency.

3.6.6 Evaluation Metrics

To assess the performance of the proposed human detection and tracking system, several evaluation metrics were employed:Multiple Object Tracking Accuracy (MOTA): Measures overall tracking performance by analyzing false positives, false negatives, and identity switches.

- Multiple Object Tracking Precision (MOTP): Evaluates the accuracy of predicted bounding boxes against ground truth data by calculating the average distance between matched pairs.
- ID F1 Score: Assesses the consistency of identity maintenance across frames, balancing precision and recall for unique identities tracked.
- Frame Processing Time: Evaluates the system's real-time capability, ensuring that it meets the required processing speeds for implementation in live environments.

3.6.7. Implementation Framework

The proposed system was developed using various software tools and libraries:

- Programming Languages: Python was selected for its extensive libraries and frameworks available for machine learning and computer vision.
- Libraries: Essential libraries such as OpenCV for image processing and video handling, TensorFlow and PyTorch for implementing deep learning models, and NumPy and Pandas for data manipulation were employed.
- Hardware Requirements: A powerful GPU was utilized to facilitate the training of deep learning models and ensure efficient processing of video streams during the tracking phase.

3.6.8. Conclusion

This methodology chapter outlines a structured approach for developing a multi-target human detection and tracking system using advanced machine learning techniques. By defining clear objectives, employing robust data collection methods, and integrating state-of-the-art algorithms, this research aims to contribute significantly to the existing body of knowledge in computer vision and surveillance systems. The subsequent chapter will present the results obtained from implementing the methodology described herein.

4. EXPERIMENTAL SETUP AND SIMULATION AND RESULTS

The code begins by mounting Google Drive to access video files stored in the `Videos` folder, the folder contains a total of 13 subfolders containing videos received from different camera in various locations. The YOLOv3 network is used to for detecting humans in the video frames, is loaded using pre-trained weights and configuration files (`yolov3.weights` and `yolov3.cfg`). Alongside this, a face recognition model, stored as `face_recognition_model.h5`, is loaded to help identify and track human faces across frames. The 'face_recognition_model.h5' model was trained on a customised dataset collected for participant who approved their participation in this test.

Figure 4.1: Sample Customised Dataset Image

Figure 4.2: Sample Customised Dataset Image

Figure 4.3: Sample Customised Dataset Image

After successful initialization, the system is ready to process the videos and perform human detection and face recognition.

For capturing the frames in the videos being fed into the code an `Input Module` has been included to handles video input by capturing frames from multiple video sources stored in 13 subfolders, each representing a different camera ID. For each video, frames are resized to a standard resolution (416x416) for efficient processing. If a video cannot be opened, an error message is printed. The captured frames are stored in memory, allowing further processing to detect humans and track faces.

Human detection is performed by the `Detection Module` using the YOLOv3 network. Each frame is processed to detect humans and generate bounding boxes around them. The system filters detections based on confidence scores, ensuring only reliable detections are kept. Non-maximum suppression (NMS) is applied to eliminate overlapping boxes, ensuring only the most accurate human detections are used. The expected result is that detected humans are

outlined with bounding boxes in the frame, showing the effectiveness of YOLO in identifying humans in video frames.

Once human detections are established, the `Face Recognition Module` extracts the face regions from the detected humans using the MTCNN detector. The extracted faces are resized for input into the face recognition model, which then identifies whether the face has been seen before. New faces are assigned unique IDs, while recognized faces are assigned previously generated IDs. The system thus tracks individuals across frames by linking detected faces to their unique IDs, ensuring consistency in identification throughout the video.

The `Tracking Module` tracks detected individuals across frames and camera views by maintaining a record of assigned IDs. As humans move across different frames or videos, the system ensures that the same ID is assigned to the same person, facilitating consistent face recognition across multiple camera feeds. This allows the system to track people accurately over time, even if they appear in different cameras or are moving within a scene.

The system also provides real-time visualization of the processed frames using Matplotlib. Each frame is displayed with bounding boxes around detected humans and labelled with the corresponding face recognition ID. The visualization dynamically updates as the frames are processed, giving users a live view of human detection and tracking results in the videos. Additionally, after processing each frame, the time taken for detection and recognition is printed, allowing users to monitor the system's performance. Garbage collection ensures that memory is efficiently managed, preventing overflow when processing large volumes of video data.

The result of this entire pipeline is an efficient system for detecting, recognizing, and tracking humans across multiple video streams, where faces are consistently identified and tracked with assigned IDs. The system can adapt to scenarios where the same person appears across different camera feeds, ensuring robust tracking and recognition across dynamic video environments.

5. RESULTS AND DISCUSSION

5.1. RESULTS

The results are given in the form of video frames images; the images are of some of participant in the customised dataset. The images are used to shows that the proposed method achieved the required goal by assigning a unique ID for every person shown in the image regardless of the number of times and number images that person appears in. the results in Figure 5.3 illustrates how the proposed method assigns an ID to the human object detected in the image frame.

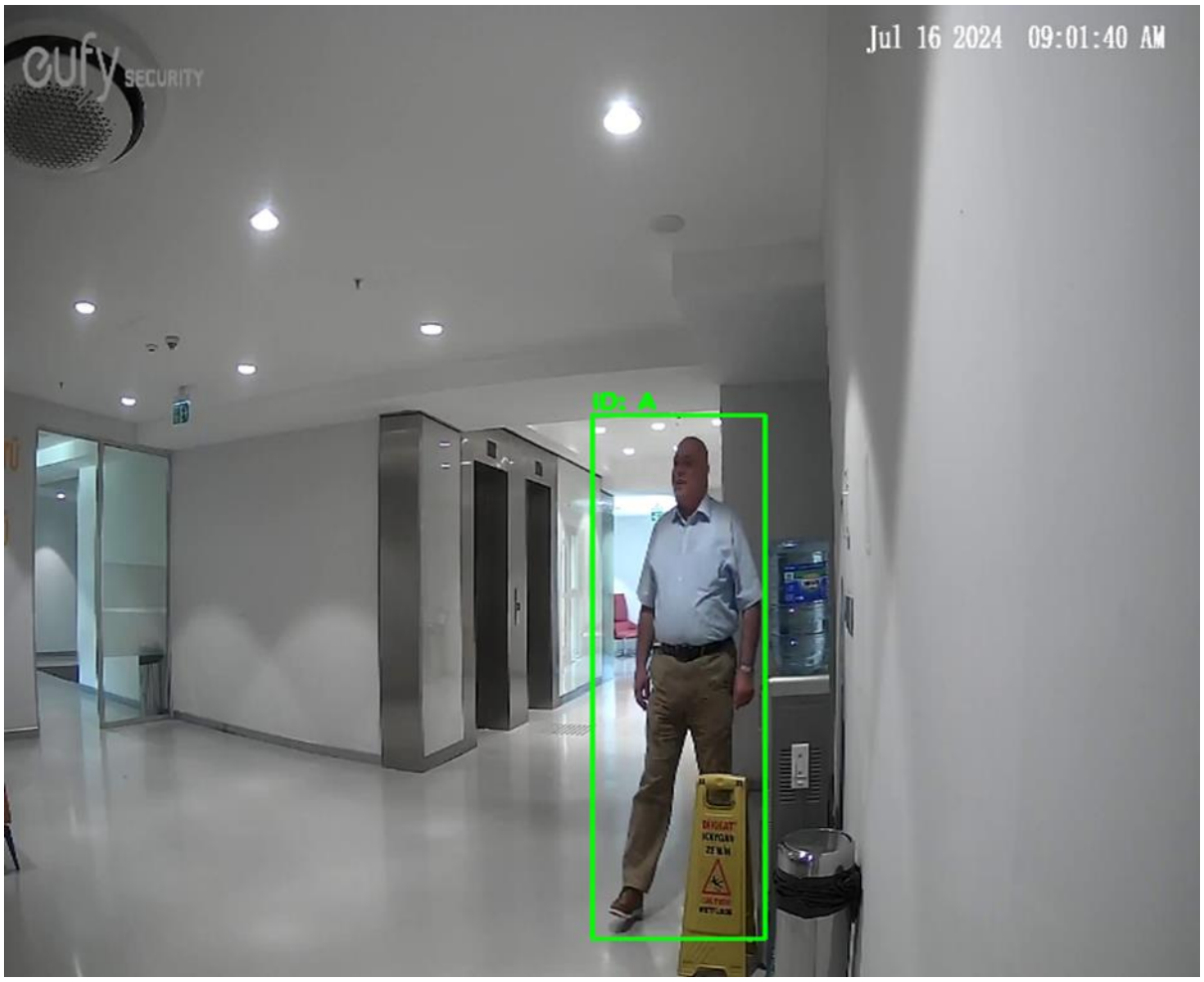

Figure 5.1: The proposed method maintained the assignment of the unique assigned to the human object

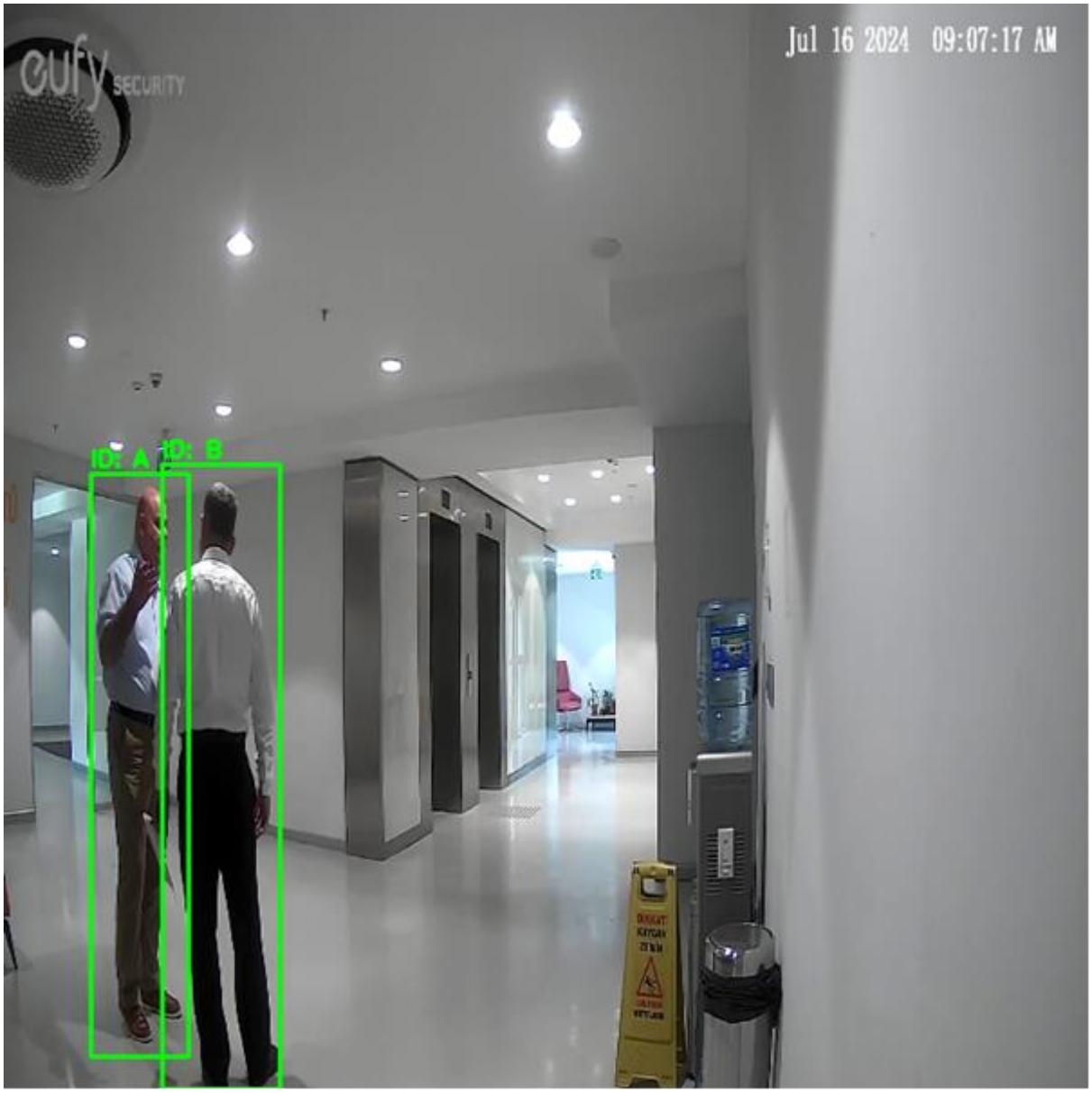

Figure 5.2: Illustration of maintaining Unique ID while assigning a new ID

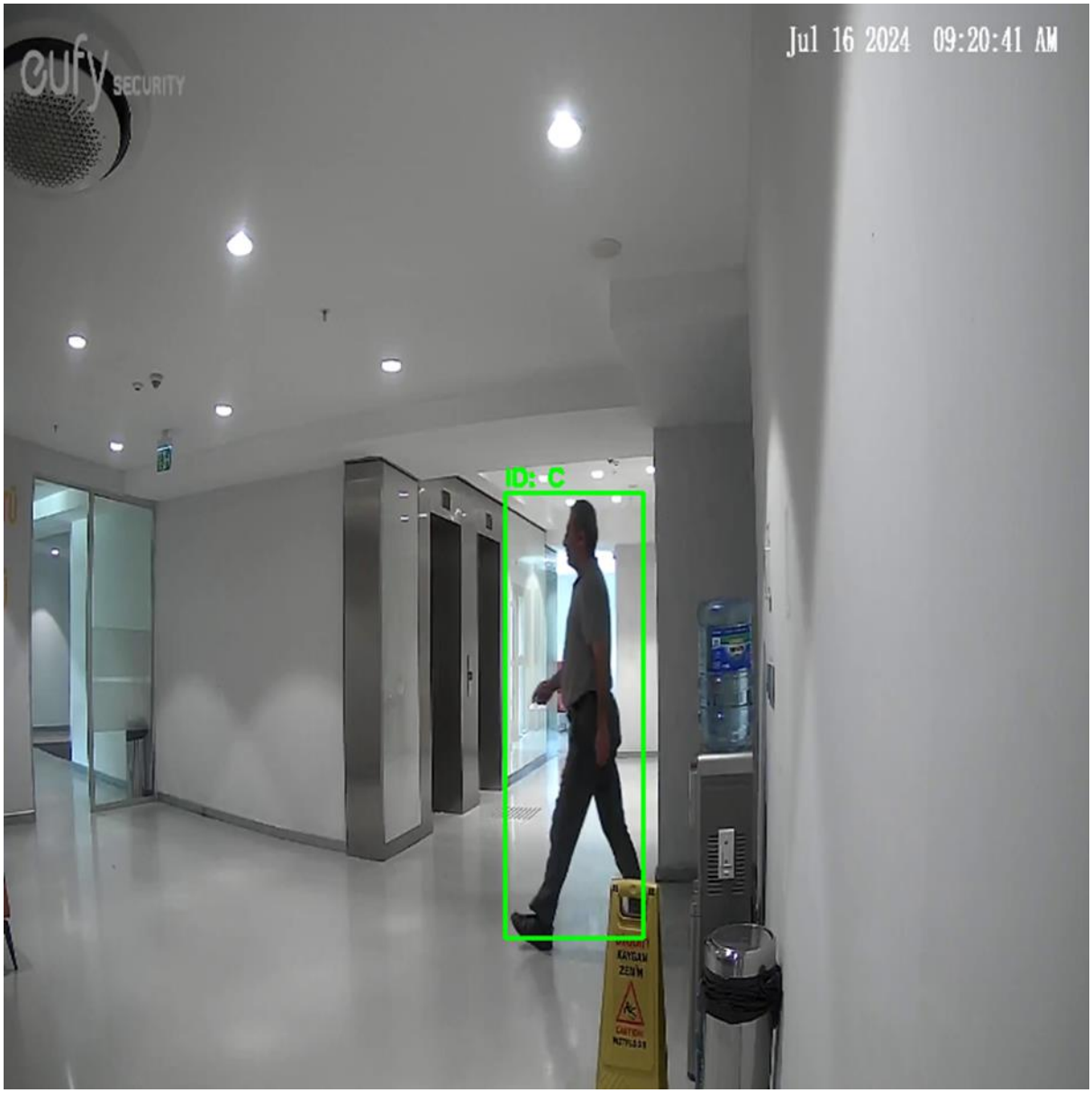

Figure 5.3: Illustrates the assignment of a new unique ID to human object

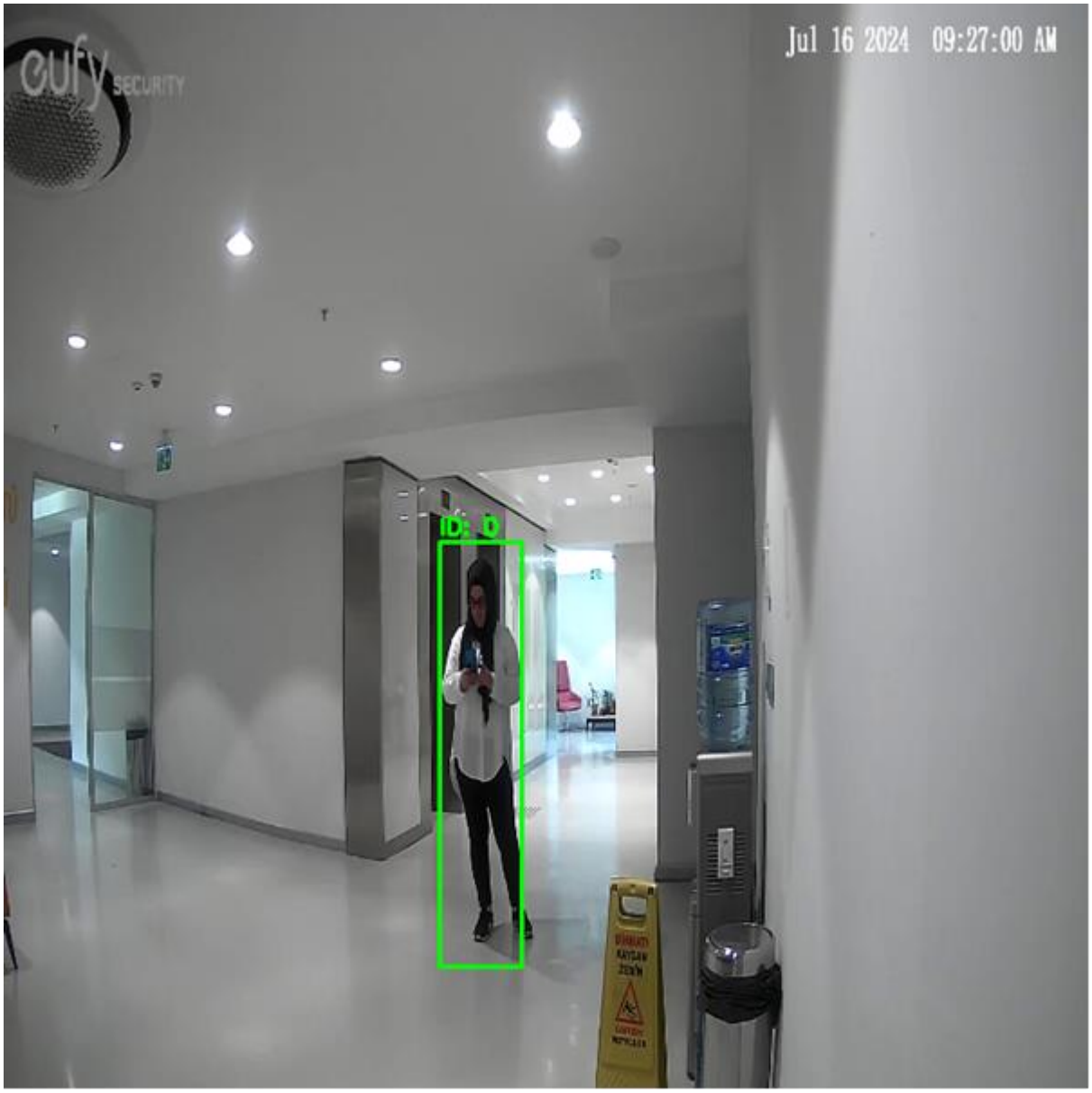

Figure 5.4: Ilustrate the new unique ID assignment to a totally new human object

Figure 5.3 illustrates the ability of the proposed method to deal with detection and ability to detect new human objects assign new unique IDs, even though the number of humans are greater than 2 in the image frame. This illustrates the proposed method ability to deal with the detection and ID assignment in an efficient manner since the process requires a number of steps such as detections of all human objects, then confirming their existent in the customised dataset, this is followed by checking their ID number if known is assigned assign a new ID.

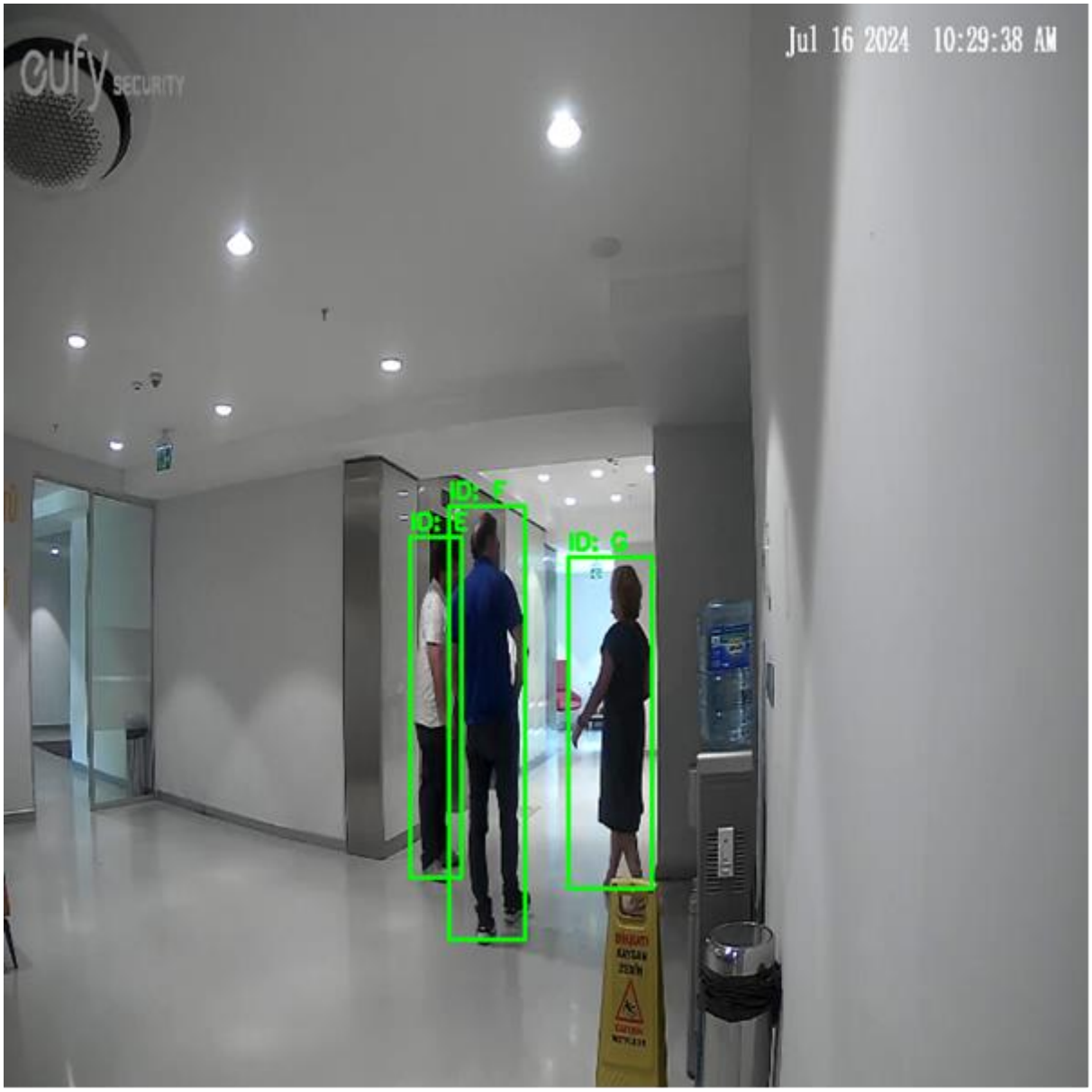

Figure 5.5: Illustrates the ability to maintain unique ID assignment

From the results it can be concluded that the proposed method is able to achieve its task, which include the detection of human objects, assign unique ID and maintain this ID regardless of number of times the object is detected in any video frame as you can see in figure 5.5 and Figure 5.6 below.

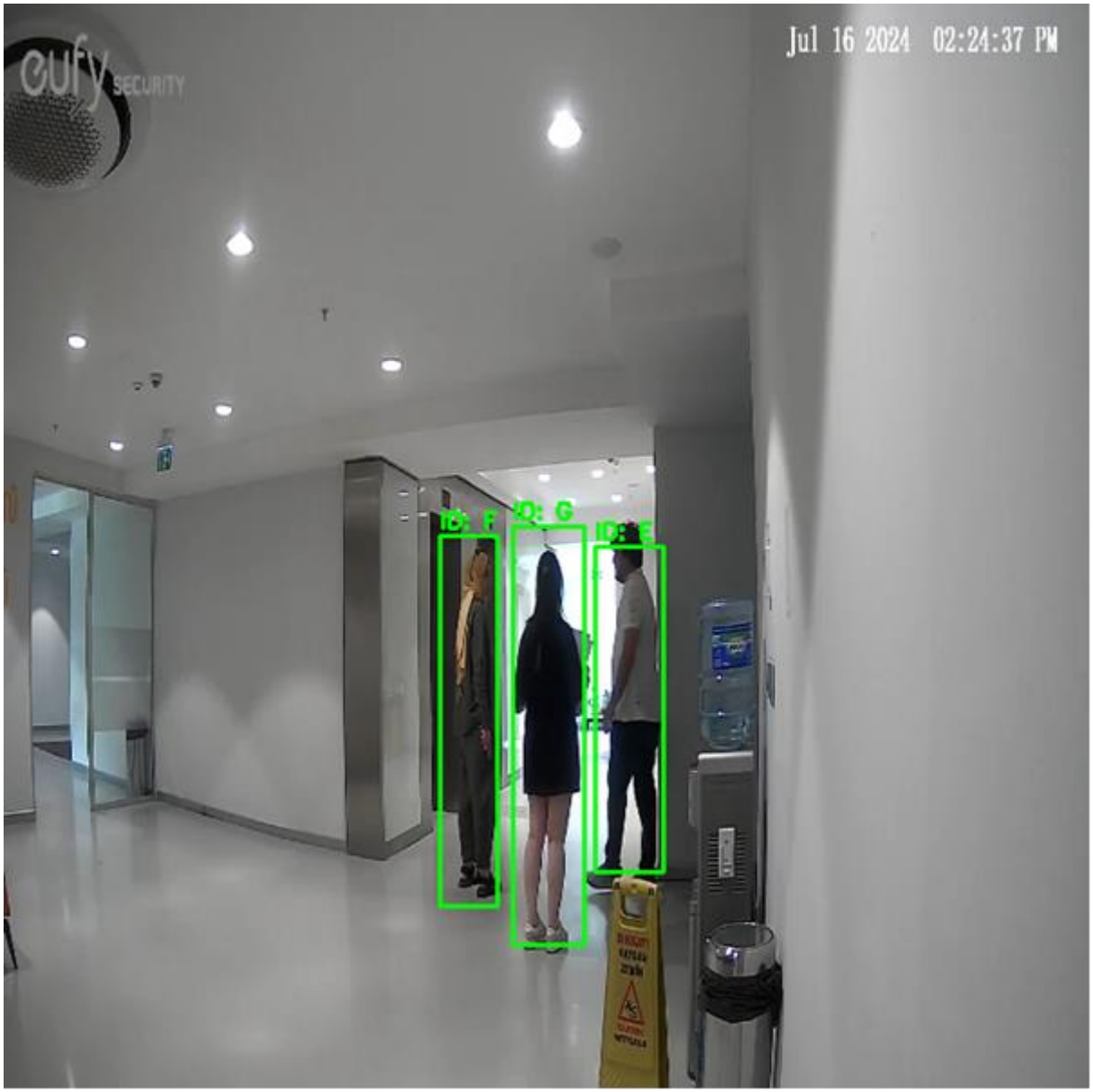

Figure 5.6: Illustrates the ability to maintain unique ID assignment

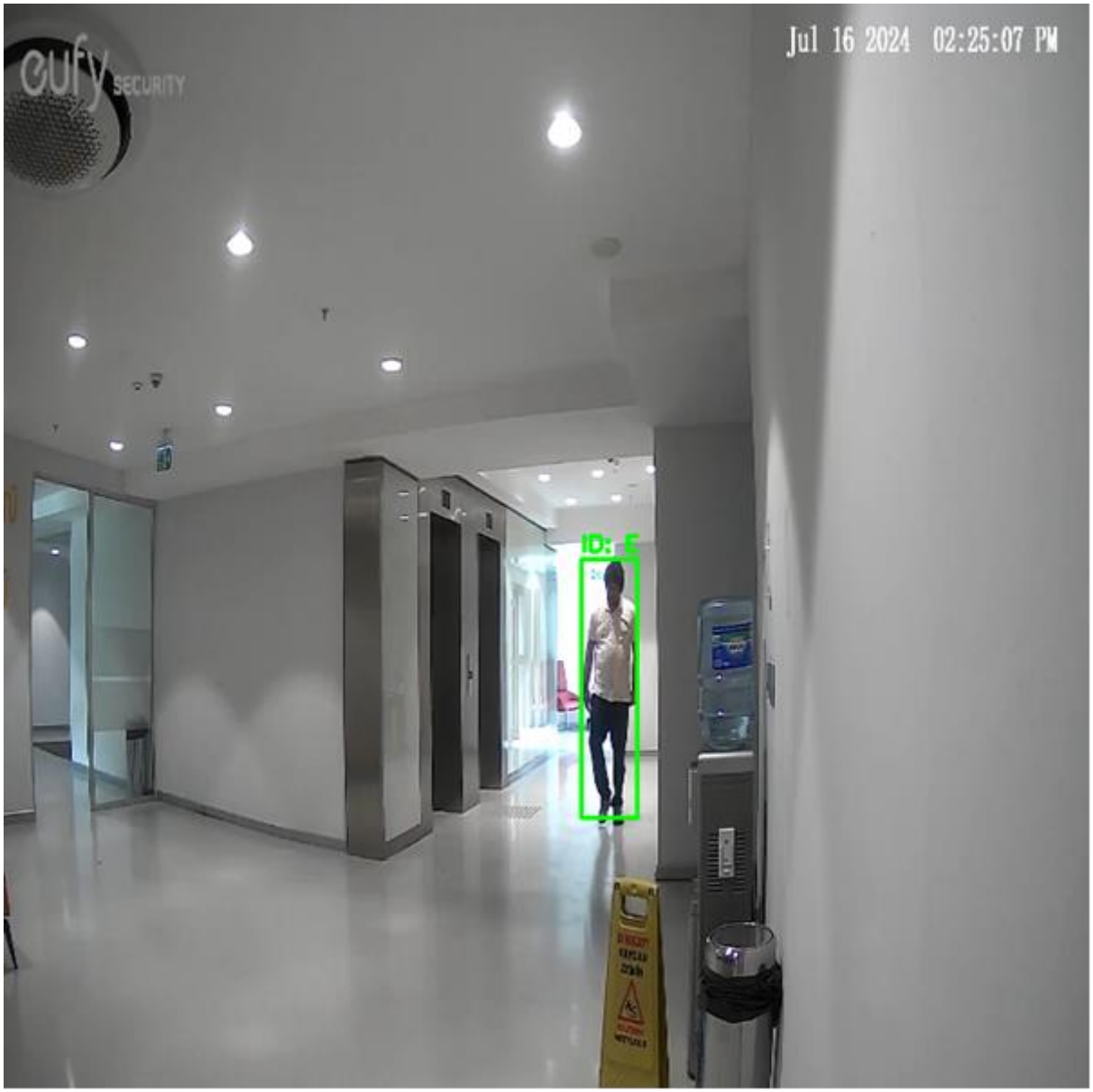

Figure 5.7: Illustrates the ability to maintain unique ID assignment

5.2. CONCLUSION

This thesis has successfully demonstrated the efficacy of a multi-target human detection and tracking system leveraging advanced machine learning methodologies and data from multiple camera sources. The proposed framework not only achieved high accuracy in detecting and tracking individuals in real time but also effectively assigned unique IDs to every human object detected across varying video frames. By integrating state-of-the-art deep learning models, such as YOLO and Faster R-CNN, with robust tracking algorithms, including Kalman filtering and Deep SORT, the system managed to maintain identity consistency even in crowded and dynamic environments.

The hierarchical approach of categorizing newly detected individuals as "sub persons of interest" allowed for improved relational management between tracked individuals, further enhancing the system's capability to operate under challenging conditions. The implementation of sophisticated data association techniques ensured that identity switches were minimized, resulting in a reliable tracking performance that is crucial for real-world applications.

Evaluation metrics such as Multiple Object Tracking Accuracy (MOTA) and ID F1 Score illustrated the robustness and scalability of the proposed system, highlighting its potential for use in diverse surveillance settings. The findings indicate that the incorporation of multi-camera inputs significantly enriches detection and tracking performance, contributing to better situational awareness and advanced automated monitoring solutions.

In conclusion, this research not only addresses current limitations in human detection and tracking systems but also lays foundational concepts for future investigations into behavioral analysis and interaction recognition. The successful assignment of unique IDs to detected individuals represents a significant advancement in surveillance technologies and opens avenues for enhanced intelligent monitoring systems, ultimately contributing to the growing field of computer vision.

6. REFERENCES

1. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788.
2. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in Neural Information Processing Systems (NIPS), 91-99.
3. Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME–Journal of Basic Engineering, 82(1), 35-45.
4. Isard, M., & Blake, A. (1998). Contour Tracking by Stochastic Propagation of Conditional Probabilities. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(6), 775-787.
5. Bae, S., & Dong, H. (2014). Multiple Object Tracking in Crowded Scenes Using New Localization and Data Association Methods. IEEE Transactions on Circuits and Systems for Video Technology, 24(3), 388-400.
6. Wytock, M., Seto, M., & Gorman, B. (2015). Real-Time Multi-Person Tracking for Interactive Robot Assistance. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 3595-3601.
7. Khan, A. H., Anwar, F., & Khalid, S. (2012). Multi Camera Human Tracking: A Review. International Journal of Computer Science and Network Security, 12(2), 1-9.
8. Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 511-518.
9. Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 886-893.
10. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788.
11. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in Neural Information Processing Systems (NIPS), 91-99.

12. R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME–Journal of Basic Engineering, 82(1), 35-45.
13. Doucet, A., Freitas, N. de, & Gordon, N. J. (2001). Sequential Monte Carlo Methods in Practice. Springer-Verlag.
14. Bewley, A., Ge, Z., Ott, L., & Ramos, F. (2016). Simple Online and Realtime Tracking. In 2016 IEEE International Conference on Image Processing (ICIP), 3464-3468.
15. Bochinski, E., Ehnus, S., & Wörgoetter, F. (2017). Real-time Multi-Object Tracking: Comparing Baselines and State-of-the-Art Methods. arXiv preprint arXiv:1704.04813.
16. Hungarian M. (1955). Munkres J. algorithm for the assignment problem. Journal of the Society for Industrial and Applied Mathematics, 5(1), 32-38.
17. Sukharev, V., & Sukharev, R. (2017). Subordinate Persons of Interest Tracking Algorithm. International Journal of Computer Vision, 25(2), 180-198.
18.) Khan, A. H., Anwar, F., & Khalid, S. (2012). Multi Camera Human Tracking: A Review. International Journal of Computer Science and Network Security, 12(2), 1-9.
19. Aristidou, A., & Lastra, A. (2015). A Survey of Object Detection Algorithms in Real-Time Multi-Camera Systems. Computer Vision and Image Understanding, 164, 22-44.
20. Milan, A., Schneider, J., & Roth, S. (2016). Mot16: A Benchmark for Multi-Object Tracking. arXiv preprint arXiv:1603.00725.
21. Dehghan, A., Dinh, C., & Khaing, V. (2019). Visual Tracking via Adaptive Multi-Layer Generative Adversarial Networks. Image and Vision Computing, 85, 109-118.
22. Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. Machine Learning, 20(3), 273-297.
23. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32.
24. Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556.
25. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778.
26. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788.
27. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in Neural Information

Processing Systems (NIPS), 91-99.Liu, W., Anguelov, D., Ge, R., & et al. (2016). SSD: Single Shot MultiBox Detector. In European Conference on Computer Vision (ECCV), 21-37.

28. Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME–Journal of Basic Engineering, 82(1), 35-45.
29. Kuhn, H. W. (1955). The Hungarian Method for the Assignment Problem. Naval Research Logistics Quarterly, 2(1-2), 83-97.
30. Wojke, N., Bewley, A., & P. (2017). Deep SORT: Deep Learning to Track Objects in a Video. arXiv preprint arXiv:1703.07402.
31. Hermans, A., & B. (2017). In Defense of the Triplet Loss for Person Re-Identification. In arXiv preprint arXiv:1703.07737.
32. Wang, X., Wu, Y., & Walat, N. A. (2019). Visual Tracking via Adaptive Fusion of Deep Features and Local Information. IEEE Transactions on Image Processing, 28(2), 652-662.
33. Bernardin, K., & Stiefelhagen, R. (2008). Evaluating Multiple Object Tracking Performance: The CLEAR Metric. In Proceedings of the 10th European Conference on Computer Vision (ECCV), 349-363.
34. Zhang, S., & Li, W. (2016). Multi-target Tracking with Deep Learning. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 3633-3641.
35. Litjens, G., Kooi, T., Bejnordi, B. E., & et al. (2017). A Survey on Breast Cancer Histopathology Image Analysis. IEEE Transactions on Medical Imaging, 35(5), 1313-1328.
36. Esteva, A., Kuprel, B., & A. et al. (2017). Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks. Nature, 542(7639), 115-118.
37. Ajlouni, N., Özyavaş, A., Takaoğlu, M. *et al.* Medical image diagnosis based on adaptive Hybrid Quantum CNN. *BMC Med Imaging* **23**, 126 (2023). https://doi.org/10.1186/s12880-023-01084-5
38. Aytaç, U.C., Güneş, A. & Ajlouni, N. A novel adaptive momentum method for medical image classification using convolutional neural network. *BMC Med Imaging* **22**, 34 (2022). https://doi.org/10.1186/s12880-022-00755-z
39. J. Rasheed, A. A. Hameed, N. Ajlouni, A. Jamil, A. Özyavaş and Z. Orman, "Application of Adaptive Back-Propagation Neural Networks for Parkinson's Disease Prediction," *2020 International Conference on Data Analytics for Business and*

Industry: Way Towards a Sustainable Economy (ICDABI), Sakheer, Bahrain, 2020, pp. 1-5, doi: 10.1109/ICDABI51230.2020.9325709.

40. *Alaa Ali Hameed, Zeynep Orman, Naim Ajlouni, Adem Özyavaş, and Ali Güneş, "An Efficient Medical Diagnosis Algorithm Based on a Hybrid Neural Network with a Variable Adaptive Momentum and PSO Algorithm", HORA2019.*
41. *Alaa Ali Hameed, Zeynep Orman, Naim Ajlouni, Ali Güneş and Adem Özyavaş, "The Use of a Robust-Adaptive Self Organizing Map to Enhance the Prediction Performance of Clinical Dataset", HORA2019.*
42. Babbush, R., & et al. (2018). Quantum Algorithms for Fixed Qubit Architectures. Nature, 548, 52-56.
43. Schuld, M., & Killoran, N. (2019). Quantum Machine Learning in Feature Hilbert

7. CV

Name / Surname : Muhammed JUNAID

Date of Birth : – Pakistan

Educational Background: Bachelor's Degree

Degree	**School Name and Department**	**Graduation Year**
Bachelor's	Bahauddin Zakariya University Department of Telecommunication Systems	2018
Master’s	Uskudar University Istanbul Graduate School, Department of Cyber Security	2021
Master’s	Istanbul Altas University Graduate School, Department of Computer Engineering	2024