Big Data and AI for people tracking in video surveillance system in real-time

1st El Hamdi Salma
Faculty of sciences
Mohammed V University in Rabat,
Rabat, Morocco
salma.elhamdi@um5r.ac.ma

2nd Refaat Khaoula
Faculty of sciences
Mohammed V University in Rabat,
Rabat, Morocco
khaoula.refaat@um5r.ac.ma

3rd Abderrazzak Chaimae
Faculty of sciences
Mohammed V University in Rabat,
Rabat, Morocco
chaimae.abderrazzak@um5r.ac.ma

***Abstract*—Context: Real-time surveillance systems are essential for maintaining public safety in crowded and high-risk environments. These systems must address challenges such as high-density crowds, varying lighting conditions, and frequent occlusions. Traditional methods, like motion detection and basic tracking, often fail to meet the demands of modern applications.**
Objectif: To develop an intelligent, scalable framework leveraging Big Data and Artificial Intelligence (AI) to improve detection, tracking, and processing in real-time surveillance.
Methods: The framework integrates YOLOv8 for precise object detection, DeepSORT for robust tracking, and Apache Spark Streaming for distributed data processing. The MOT17 dataset was used to evaluate system performance.
Resultats : The framework effectively handles challenges in dynamic environments, providing accurate detection, reliable tracking, and efficient processing of large-scale video data streams.
Conclusion: By combining state-of-the-art algorithms and distributed computing, this framework demonstrates the potential for intelligent, scalable, and efficient surveillance solutions in real-world scenarios.

***Index Terms*—Big Data, Artificial Intelligence, YOLOv8, DeepSORT, Apache Spark-Streaming, Real-Time Analytics, Computer Vision, Machine Learning, Video Analytics**

I. Introduction

Real-time surveillance systems have become indispensable for monitoring and ensuring public safety, particularly in crowded or high-risk environments. These systems must handle dynamic challenges, such as high-density crowds, varying illumination conditions, and frequent occlusions. Traditional surveillance approaches relied on static methods, such as motion detection and basic object tracking, but they often failed to meet the demands of modern security needs.

The integration of Big Data and Artificial Intelligence (AI) has revolutionized these systems, enabling real-time analysis and proactive decision-making through advanced detection and tracking algorithms. This paper details a comprehensive framework combining state-of-the-art technologies to enhance detection, tracking, and scalability in real-time video surveillance. The framework begins with preprocessing, a critical step to ensure the quality and consistency of input data. Surveillance videos are divided into frames, which are then organized into structured datasets. The MOT17 dataset, widely recognized for its diversity and complexity, is used for training and evaluation. Each frame is annotated with bounding boxes, object IDs, and visibility scores, ensuring that the models are well-prepared for real-world scenarios.[1]

Object detection is handled by YOLOv8, a cutting-edge algorithm that offers a balance between speed and accuracy. YOLOv8 partitions video frames into grids and predicts the locations and classes of objects within them. While YOLO excels in real-time scenarios, it faces challenges such as occlusion, overlapping objects, and varying lighting conditions, which can affect detection accuracy. These limitations are mitigated through careful parameter optimization and the use of robust datasets like MOT17. The tracking component is powered by DeepSORT, which assigns unique identifiers to detected objects and maintains their consistency across frames. DeepSORT leverages appearance-based features to address challenges like occlusions and identity switches. However, in crowded scenes, the algorithm can still struggle with misidentifications, particularly when multiple objects overlap or when individuals move rapidly. Enhancing the feature extraction process and optimizing the data association step are key to overcoming these challenges.[2]

To process the vast amount of video data in real time, the system integrates Apache Spark Streaming. This distributed computing platform enables the parallel processing of high-resolution video feeds from multiple cameras, ensuring scalability and low-latency performance. However, the integration of Spark Streaming with AI models introduces challenges, such as balancing computational loads across nodes and ensuring synchronized data processing. These issues are addressed through the efficient allocation of resources and the adoption of fault-tolerant strategies.[3]

Finally, the framework combines these components to create a cohesive system capable of handling complex real-world scenarios. Preprocessing ensures high-quality inputs, YOLOv8 delivers fast and accurate detections, DeepSORT maintains robust tracking, and Spark Streaming enables large-scale data handling. Together, these technologies form a powerful solution for real-time surveillance applications, addressing key challenges and paving the way for more intelligent and efficient monitoring systems.[4]

II. Literature Review

This section provides an overview of the methodologies and advancements discussed in the literature for object detection and tracking in surveillance systems. The review focuses on the progression from traditional approaches to modern deep learning-based techniques.

The primary goal of these methods is to detect objects within video sequences and track them across successive frames to determine their trajectories. Early approaches relied on motion and observation models, leveraging techniques like Kalman filters and probabilistic methods to predict object positions. Later, machine learning methods such as boosting, random forests, and support vector machines enhanced object classification and tracking accuracy.

Feature-based techniques, including Haar-like features, histograms of oriented gradients, and scale-invariant feature transforms, further improved detection capabilities. With the advent of deep learning, algorithms such as YOLO and recurrent neural networks have revolutionized real-time tracking by combining appearance features and temporal analysis to handle challenges like occlusion and dynamic backgrounds. These advancements have paved the way for more robust and accurate surveillance systems capable of addressing real-world complexities.

A. Reviewing Some Related Work

Review on: Object Detection and Tracking using YOLOv8 and DeepSORT

Recent advancements in object detection and tracking, such as the integration of YOLO (You Only Look Once) and DeepSORT algorithms, have significantly improved real-time applications like surveillance and traffic monitoring. YOLOv8 stands out for its enhanced multi-scale detection capabilities through advanced features like the Feature Pyramid Network (FPN) and Path Aggregation Network (PAN), while DeepSORT's incorporation of appearance features reduces identity shifts during occlusions. Together, these approaches deliver high accuracy and processing speeds, addressing critical challenges in dynamic environments. Despite their effectiveness, challenges like occlusions and complex backgrounds remain, driving ongoing research to develop tightly integrated detection and tracking systems for smart city applications, aiming to improve traffic management and public safety.[5]

Review on: Analyzing Surveillance Videos in Real-Time using AI-Powered Deep Learning Techniques

The evolution of surveillance systems, from traditional CCTV setups to AI-driven real-time video analysis, has significantly improved security measures. Research highlights the integration of convolutional neural networks (CNNs) for object recognition and recurrent neural networks (RNNs) for behavior analysis, achieving up to 95% accuracy in detecting and tracking objects. These advancements enable proactive anomaly detection, such as identifying threats like violence or theft. However, challenges persist, including the computational demands of processing high-resolution video data and environmental factors like lighting variations and weather changes, which can affect detection accuracy. Future directions emphasize integrating distributed AI systems for autonomous decision-making and ensuring data security to protect against breaches, making AI-powered surveillance a cornerstone of modern public safety strategies.[6]

Review on: Convolutional neural network–based person tracking using overhead views

One study focused on leveraging convolutional neural networks (CNNs) to enhance person tracking in video surveillance systems by utilizing overhead views to mitigate common challenges such as occlusion, close interactions between individuals, and variations in illumination or background complexity. The researchers proposed a combination of Faster-RCNN for person detection and GOTURN for tracking, achieving robust performance metrics. Faster-RCNN, a region-based detection framework, operates by generating bounding boxes through a region proposal network (RPN) and refining them for object classification. The model achieved a true detection rate (TDR) ranging from 90% to 93% and maintained a false detection rate (FDR) as low as 0.5%. For tracking, the GOTURN algorithm utilized convolutional layers to predict the position of detected individuals across consecutive video frames, resulting in a tracking success rate of up to 94%. The study utilized a diverse dataset specifically designed for overhead views, including both indoor and outdoor environments with significant variations in person appearance, pose, scale, and camera resolution. This dataset presented unique challenges compared to frontal-view datasets; however, the models demonstrated remarkable generalization capabilities despite being pre-trained on standard frontal-view data. By using an overhead perspective, the system addressed limitations of traditional surveillance methods, offering improved visibility and reduced occlusion. Furthermore, the overhead camera setup proved to be a cost-effective solution, minimizing energy consumption, human resources, and installation expenses while maintaining high performance. This study underscores the potential of CNN-based models to revolutionize real-time video surveillance systems and lays a foundation for future research aimed at optimizing tracking performance using specialized datasets tailored for overhead perspectives.[7]

Review on: A Comparison of Multicamera Person-Tracking Algorithms

A.W. Senior, G. Potamianos, S. Chu, Z. Zhang, and A. Hampapur conducted a comparative study of four advanced algorithms for real-time person tracking in indoor environments using multicamera systems. These algorithms include background subtraction, particle filter tracking, face detection, and edge alignment utilizing 3D cylindrical models. Each method showcases unique strengths: background subtraction offers high accuracy but requires a stable background model, while particle filters are robust to background changes but can be misled by unrelated movements. Face detection is effective for speaker localization but depends on the visibility of faces, and edge alignment provides high precision but relies on accurate initialization. Evaluated using data from the CHIL project, the results indicated average errors below 500 mm for background subtraction and face detection. This work underscores the challenges and progress in developing robust algorithms for real-time person tracking in multicamera surveillance systems.[8]

Review on: A Big Data Platform for Real-Time Video Surveillance

Improvements in object detection and tracking, including the incorporation of Kafka and Spark frameworks, have enhanced real-time video surveillance systems. The use of YOLO and libraries like OpenCV ensures high accuracy in object detection, while the integration with distributed databases (HBase) and relational databases (SQL Server) allows efficient storage and querying of processed data.

Systems like the platform proposed by Thi-Thu-Trang Do et al. stand out for their ability to combine real-time and historical video data using distributed technologies such as Spark Structured Streaming. The architecture includes components for data collection, storage, and intelligent video analysis. However, challenges remain, including optimizing AI models and integrating user-friendly interfaces for simplified operations.

These studies emphasize the importance of developing robust and scalable solutions for real-time video surveillance systems, particularly for applications such as urban management and public safety.[9]

Review on: Real-time object detection, tracking, and monitoring framework for security surveillance systems

Recent advancements in object detection and tracking, such as the integration of approximate median filtering and deep learning approaches, have significantly enhanced real-time applications like security surveillance. The proposed framework combines methods such as component labeling, background subtraction, and deep learning for accurate detection and tracking. These algorithms, implemented in Python and integrated with C# for usability, outperform state-of-the-art techniques on MOT15, MOT16, and MOT17 datasets, demonstrating superior accuracy and precision.

The system also integrates user-friendly software development, enabling seamless execution as a standalone application via Microsoft Visual Studio. Despite its effectiveness, challenges remain, such as adapting to overcrowded scenarios and varying environmental conditions. Future improvements will explore dynamic scalability and incorporate advanced object detection techniques, like YOLO and its variants, to address complex surveillance challenges and further enhance security monitoring and public safety.[10]

III. Methodology

The methodology outlined in this section details the steps and components of the proposed system for real-time people tracking in video surveillance. It begins with data preprocessing, followed by a description of the dataset utilized. Subsequently, the proposed model is introduced, including its architecture and specific components. Finally, implementation details are provided to explain the practical aspects of deploying the model in real-world scenarios.

*A. **Preprocessing***

Pre-processing is an important step to prepare surveillance data for analysis using **YOLO, DeepSORT**[11][12], and **Spark Streaming**[13]. This phase ensures optimal organization and quality of the data to enhance model performance.

Extraction and Division of Video Frames

Surveillance videos, often large in size, are divided into individual frames for simplified processing. These frames are extracted at a predefined frequency to ensure consistent coverage of the scenes.[14]

Data Organization

The extracted frames, forming a dataset of approximately 5 GB, are split into two main sets:

- **Train:** Used for model training.
- **Test:** Used for evaluating model performance.

Each set is further divided into two subfolders:

- **Images:** Containing the extracted frames as image files.
- **Labels:** Containing annotations associated with each frame.

These annotations follow the **YOLO** format, with normalized coordinates indicating object classes and their positions.[15]

Data Loading into YOLO

The images and annotations are prepared and loaded into the YOLO model for object detection. This process includes:

Resizing frames to a standard size (e.g., 640x640).
Normalizing pixel values to improve input quality.
Validating and cleaning annotations to ensure compatibility with the model.

Integration with DeepSORT and Spark Streaming

- **DeepSORT:** Uses the frames and bounding boxes generated by YOLO to extract visual features and track objects across successive frames.
- **Spark Streaming:** Processes frames and intermediate results in real-time to enable continuous analysis. Frames may be converted into compatible formats for distributed processing, such as tensors or compressed files.

By combining these steps, the system efficiently prepares the data for precise and rapid processing, enabling intelligent real-time surveillance.

B. *Data Description*

The dataset utilized for this study is the **MOT17 (Multiple Object Tracking 2017)** dataset, sourced from the official MOTChallenge repository. This dataset is a well-established benchmark widely used for evaluating multi-object tracking algorithms. It provides essential resources for developing and testing systems focused on real-time people tracking in video surveillance scenarios. The MOT17 dataset comprises high-quality video sequences, meticulously annotated for pedestrian detection and tracking. These sequences are divided into two parts: a **training set** and a **testing set**.
Given that the dataset exceeds 5 gigabytes in size, it is considered big data and adheres to **the 5V** characteristics: Volume (large data size), Velocity (high frame rates and real-time processing needs), Variety (diverse environments and camera setups), Veracity (detailed annotations ensuring data quality), and Value (crucial for developing tracking systems in real-world surveillance conditions).[16]

The dataset includes 14 video sequences, with 7 allocated for training and 7 for testing. The resolutions of the video sequences vary between 640×480 and 1920×1080 pixels, with frame rates ranging from 30 to 60 frames per second. Each sequence is annotated with bounding boxes, unique object IDs, visibility scores, and occlusion indicators, enabling detailed tracking of pedestrians across frames. The scenes include diverse environments such as urban streets, public squares, and semi-crowded areas, with a mix of static and moving camera setups to simulate real-world surveillance conditions.

Feature Details of MOT17 Dataset: The annotations in each frame provide bounding boxes to define pedestrian regions, unique IDs to maintain tracking consistency, visibility scores ranging from 0 to 1 to indicate the proportion of visible pedestrian area, and flags for partial or full occlusions. These detailed annotations make the MOT17 dataset a robust tool for training and testing algorithms designed for real-world applications.

The dataset poses several challenges that align with the objectives of this study, including high crowd density, frequent occlusions, and varying illumination conditions. Additionally, the presence of both static and dynamic camera motion further tests the robustness of tracking algorithms. To leverage the dataset effectively, the video sequences and annotations were ingested into a big data architecture using Apache Kafka, and the processed data was stored in distributed systems like HBase for scalability.

Feature Name	**Type**	**Detail**
Video Sequences	Integer	14 sequences: 7 for training and 7 for testing
Resolution	Integer (pixels)	Resolutions range between 640×480 and 1920×1080
Frame Rate	Integer (frames/second)	Frame rates vary between 30 and 60
Resting blood pressure	Integer	Patient blood pressure level
Bounding Boxes	Coordinates (x, y, w, h)	Rectangular areas surrounding pedestrians in each frame
Object IDs	Integer	Unique IDs assigned to each pedestrian
Visibility Scores	Float (0 to 1)	Indicates the proportion of the pedestrian visible in a given frame
Occlusion Indicators	Boolean	Indicates whether the pedestrian is partially or fully occluded
Scene Type	Categorical	Urban streets, public squares, or semi-crowded areas
Camera Motion	Categorical	Static or moving camera
Challenges	Categorical	Includes crowd density, occlusions, varying illumination, and scale changes

TABLE I: Attributes & description of data set

The training set was used to train detection and tracking models, such as YOLO for pedestrian detection and DeepSORT for tracking. Real-time analytics were performed using Apache Spark Structured Streaming, ensuring scalability and low-latency processing. This workflow allowed the system to handle the complexity and volume of data effectively, enabling robust real-time tracking for video surveillance applications.

C. *Proposed Model*

The model proposed in this article combines YOLOv8 for object detection, Deep SORT for tracking, and Spark Streaming for handling large-scale data in real-time video surveillance systems. YOLOv8 is used to detect people in video frames by dividing each frame into a grid, predicting bounding boxes, and classifying detected objects with high accuracy [17]. Its efficiency makes it suitable for real-time surveillance, ensuring fast and accurate detection of individuals in dynamic environments.

Once people are detected, Deep SORT is employed to track them across frames. Deep SORT assigns unique identities to each detected person and maintains their identity even in cases of occlusion or overlapping. By using appearance-based features, Deep SORT ensures consistent tracking of individuals, making it ideal for surveillance scenarios where individuals may temporarily disappear or interact with others [18].

To manage the large volume of data from multiple cameras, the system integrates Spark Streaming [19]. Spark Streaming allows for the parallel processing of video streams, distributing the workload across a cluster of machines. This enables real-time tracking and analysis of people across multiple video feeds, ensuring the system can scale efficiently without compromising performance.

Together, YOLOv8, Deep SORT, and Spark Streaming provide a robust and scalable solution for real-time people tracking in video surveillance systems, making it suitable for large-scale applications like security and crowd monitoring.

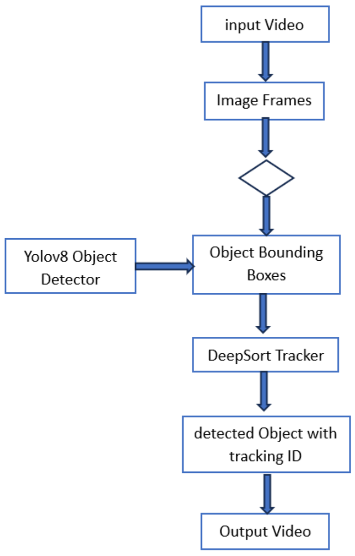

Fig. 1: Proposed Model for Real-Time People Tracking Using YOLOv8 and Deep SORT

D. Model Description

The key metrics used to evaluate the performance of the YOLO and DeepSORT models in the project. These metrics provide insights into the model's detection and tracking capabilities during training and validation phases.

Epochs and Time

- Epoch: Represents the number of complete passes through the training dataset. Each epoch involves forward and backward passes for all training samples.[20]

- Time: Indicates the duration (in seconds) taken for each epoch. This metric reflects computational efficiency and helps identify bottlenecks during training.

Loss Metrics

Loss metrics measure the deviation between predictions and ground truth, guiding model optimization:

- **train/box_loss:** Measures the error in bounding box predictions during training.
- **train/cls_loss:** Reflects the error in object classification during training.
- **train/dfl_loss** The distribution focal loss addresses class imbalance by weighting harder-to-detect classes more heavily.[21]

Validation losses are calculated on the validation dataset to monitor generalization:

- **val/box_loss:** Bounding box loss on the validation set.
- **val/cls_loss:** Classification loss on the validation set.
- **val/dfl_loss:** Distribution focal loss on the validation set.[22]

Evaluation Metrics

These metrics evaluate the model's detection accuracy and reliability:

- **Precision (PPP):** Indicates the proportion of true positive detections among all positive detections: High precision reflects fewer false positives.
- **Recall (RRR):** Reflects the proportion of true positive detections among all actual positives:High recall indicates fewer missed detections.[23]

Learning Rates

Learning rates are crucial for optimizing the model during training. Different parameter groups in the model have distinct learning rates:

- **lr/pg0:** Learning rate for the first parameter group, typically responsible for backbone layers.
- **lr/pg1:** Learning rate for the second parameter group, often for additional layers.
- **lr/pg2:** Learning rate for specialized parameters like object detection heads.[24]

E. Implementation Detail

The project was implemented using the MOT17 dataset, a 5GB collection of video sequences from real-world environments. These videos were converted into individual frames for easier processing and compatibility with object detection and tracking tasks. The YOLO algorithm was utilized for its real-time, high-accuracy object detection capabilities, specifically to identify pedestrians in complex scenarios. Following detection, the DeepSORT tracking algorithm assigned unique identifiers to individuals, ensuring consistent tracking across frames, even in challenging conditions such as occlusions and crowded scenes. The system output consisted of a processed video demonstrating real-time detection and tracking, with bounding boxes and unique IDs for each pedestrian.

For real-time processing, Apache Spark Streaming was integrated to handle the continuous flow of frames efficiently, enabling scalable, low-latency analysis of the detection and tracking pipeline. The computational demands of training and running YOLO and DeepSORT necessitated the use of a GPU, which was provided by Google Colab. This platform facilitated both training and inference, ensuring optimal performance while highlighting the system's dependency on GPU-enabled environments. The combination of the MOT17 dataset, YOLO, DeepSORT, and Spark Streaming demonstrated the feasibility and effectiveness of real-time people tracking in video surveillance systems.

IV. Results and Discussion

This section provides a comprehensive analysis of the outcomes obtained from our real-time surveillance system for person detection and tracking. The proposed system combines three main components: YOLO for object detection, DeepSort for object tracking, and Spark Streaming for processing and scalability.

The results highlight the strengths and limitations of each component, offering a detailed evaluation of detection accuracy, tracking performance, and real-time processing capabilities. This discussion aims to demonstrate the effectiveness of the integrated system in addressing the challenges of real-time surveillance in dynamic environments, ensuring both precision and scalability

A. *Yolo*

The application of YOLOv8 enables the evaluation of object detection efficiency in real-time, leveraging its fast and accurate prediction capabilities. For this task, the model was trained with specific parameters, including a set number of epochs (30), an auto-scaled batch size based on GPU memory, and an image size of 640 pixels, optimizing both detection accuracy and computational efficiency. The use of a GPU facilitated the optimization of training and execution times. The results are presented through visual analyses and metrics such as the confusion matrix, providing a detailed view of the model's performance in real-world scenarios. This approach helps to understand how YOLOv8 detects and localizes objects of interest while considering the parameters that influence its performance.

Performance Analysis of YOLO Model

The model training was conducted over 30 epochs, and here is a comprehensive analysis of the key observations and trends:

- **Model Precision Evolution:**
  The mAP50 (mean Average Precision at 50% IoU threshold) shows a steady improvement from 0.42 to 0.45, indicating effective learning from training data.
  The mAP50-95 metric increases from 0.26 to 0.32, demonstrating robust detection capabilities across various IoU thresholds.
  Precision(B) stabilizes at approximately 0.90, while Recall(B) converges around 0.78, suggesting a good balance between accurate detections and false positives.

- **Loss Performance:**
  Box loss shows significant improvement, decreasing from 1.1 to 0.5 in training data, indicating enhanced localization accuracy.
  Classification loss (cls_loss) demonstrates steady convergence from 1.3 to 0.9.
  DFL (Distribution Focal Loss) decreases from 0.95 to 0.82, suggesting improved feature learning.
  Validation losses follow similar trends, confirming stable model optimization.

- **Model Convergence Analysis:**
  All training losses show consistent downward trends without significant plateaus, indicating effective learning.
  Validation metrics demonstrate stable performance with minor fluctuations, suggesting good generalization.
  The smooth convergence curves suggest appropriate learning rate and optimization settings.

- **Generalization and Stability:**
  The parallel trends between training and validation metrics indicate good generalization without overfitting.
  Performance metrics show stability in later epochs, suggesting the model has reached a reliable state.
  The consistent gap between training and validation losses remains within acceptable bounds, confirming proper model regularization.

This analysis suggests that the model is well-suited for real-time person tracking in video surveillance systems, achieving a good balance between accuracy and generalization. The steady improvement in precision metrics, combined with the consistent decrease in loss values, indicates that the model has successfully learned the key features necessary for reliable person detection and tracking.

This analysis provides a detailed understanding of the model's performance and is crucial for further fine-tuning or drawing comparisons with other models in the study.

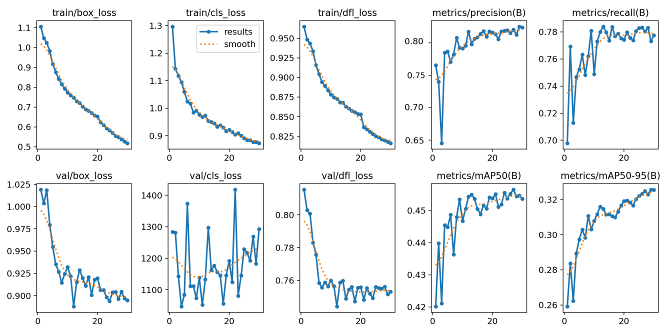

Fig. 2: The results

TABLE II: Summary of Training Metrics for First 5 Epochs

Metric	Ep.1	Ep.2	Ep.3	Ep.4	Ep.5
Training Metrics					
Box Loss	1.106	1.047	1.024	0.982	0.917
Class Loss	1.297	1.143	1.118	1.095	1.059
DFL Loss	0.965	0.949	0.943	0.934	0.916
Performance Metrics					
Precision (B)	0.761	0.742	0.646	0.781	0.783
Recall (B)	0.698	0.769	0.713	0.747	0.752
mAP50 (B)	0.420	0.440	0.421	0.445	0.445
mAP50-95 (B)	0.259	0.284	0.262	0.289	0.297
Validation Metrics					
Val_Box_Loss	1.019	1.004	1.019	0.980	0.955
Val_Class_Loss	1284.0	1280.9	1142.9	1047.3	1084.3
Val_DFL_Loss	0.815	0.803	0.801	0.783	0.776
Training Parameters					
Time (s)	472.6	945.9	1419.5	1890.5	2346.8
LR (pg0)	0.00066	0.00129	0.00186	0.00180	0.00174

- **Loss Functions Evolution**

  Box Loss (train/box_loss): Shows steady decrease from 1.1 to 0.5, indicating good convergence in object detection.
  Classification Loss (train/cls_loss): Decreases from 1.3 to 0.9, showing improved classification accuracy.
  DFL Loss (train/dfl_loss): Gradual reduction from 0.95 to 0.82, suggesting refinement in feature learning.

- **Validation Metrics:**

  Box Loss (val/box_loss): Stabilizes around 0.9 after initial decrease.
  Classification Loss (val/cls_loss): Shows some fluctuation (800-1400 range).
  DFL Loss (val/dfl_loss): Settles around 0.75 with minor oscillations.

- **Performance Metrics:**

  Precision(B): Improves and stabilizes at 0.80, indicating good accuracy in detection **Recall(B):** Reaches 0.78 with some fluctuations.
  mAP50(B): Achieves 0.45, showing decent mean Average Precision.
  mAP50-95(B): Steady increase to 0.32, indicating robust performance across various IoU thresholds.

• The values in the table for each epoch show a significant decrease in losses and an improvement in metrics, which is confirmed by the graphs. This indicates that the training is on track and the model is in continuous improvement, which is essential for practical outcomes.

CONFUSION MATRIX

The confusion matrix is a fundamental evaluation tool in machine learning used to assess the performance of classification models. It provides a detailed summary of the predictions made by a model, categorizing them into true positives, true negatives, false positives, and false negatives. Represented as an N x N matrix, where N is the number of target classes, the confusion matrix is particularly useful for binary and multi-class classification tasks. Each row of the matrix represents the instances of the actual class, while each column represents the instances of the predicted class. By examining the diagonal elements of the matrix, one can identify the number of correct predictions, which is essential for estimating the model's accuracy. Additionally, the matrix facilitates the computation of other performance metrics such as precision, recall, F1-score, and specificity, offering a comprehensive understanding of the model's strengths and weaknesses[25]. This makes it a vital tool for identifying areas where the model may require improvement or adjustment.

Performance Metrics Calculation

Performance metrics are used to evaluate the quality of a classification model's predictions. They are derived from the values in the confusion matrix:

1. Precision (PPV)

$$PPV = \frac{TP}{TP + FP} \tag{1}$$

Precision evaluates the proportion of correctly predicted positive instances among all positive predictions.

2. Recall (Sensitivity)

$$TPR = \frac{TP}{TP + FN} \tag{2}$$

Recall measures the ability to find all positive instances.

3. F1-Score

$$F1 = 2 \times \frac{PPV \times TPR}{PPV + TPR} \tag{3}$$

The F1-Score is the harmonic mean of precision and recall, useful when balancing these two metrics is crucial.

Legend of Terms

- **TP (True Positives):** Number of positive instances correctly predicted.
- **TN (True Negatives):** Number of negative instances correctly predicted.
- **FP (False Positives):**Number of negative instances incorrectly predicted as positive.
- **FP (False Positives):** Number of positive instances incorrectly predicted as negative.

These metrics provide a comprehensive view of a model's performance, enabling comparisons with other models and identifying areas for improvement.

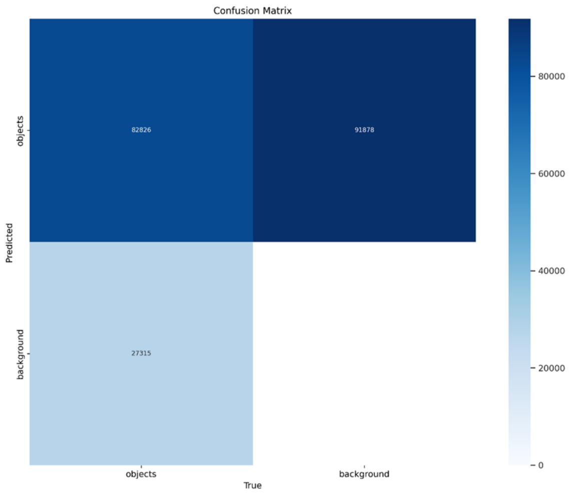

Fig. 3: confusion matrix precision

The confusion matrix is used to evaluate the performance of the real-time video analysis pipeline based on YOLO for object detection and Deep SORT for person tracking. It provides an overall view of correct and incorrect predictions for the system's main classes: **objects** and background. In this confusion matrix **(Fig3)**, the x-axis (True) represents the actual classes, while the y-axis (Predicted) corresponds to the model's predictions. This 2×2 matrix is designed for two primary classes: Objects (detected as people or objects of interest) and **Background** (areas without objects of interest) [26]. The key values in the matrix include **True Positives (TP)**: 82,826 correct predictions for the "objects" class, **False Positives (FP)**: 91,878 instances incorrectly classified as "objects," and **False Negatives (FN)**: 27,315 objects misclassified as "background."

The following performance metrics were calculated from the confusion matrix values to evaluate the object detection and tracking system. The values of true positives (TP), false positives (FP), and false negatives (FN) provide a general idea of the model's performance. The **precision** (PPV), which measures the proportion of correct positive predictions, is 80.4%, indicating that a significant proportion of predictions for the "objects" class are correct. **The recall** (or sensitivity, TPR) is 75.2%, showing that the system successfully detects the majority of objects in the scene, although some are still missed. The **F1-score**, which balances precision and recall, is 0.578 (57.8%), reflecting an equilibrium between the two metrics but also suggesting room for improvement in the model. Finally, **specificity** (TNR), which measures the model's ability to correctly classify background areas, cannot be calculated due to the absence of information on true negatives (TN), but it would be crucial to assess the model's effectiveness in classifying the background. These metrics provide valuable insights into the strengths, such as the high recall (75.2%), meaning the system correctly detects a large proportion of objects. This is especially critical in surveillance systems, where missing objects can be crucial. These metrics also highlight the system's weaknesses, allowing for the identification of areas for improvement in future optimizations. [27]

ANALYSIS RESULTS

This section presents a detailed examination of the results obtained from the people detection task using YOLOv8. The analysis focuses on key metrics, visualizations, and patterns observed in the dataset and model performance, providing insights into the behavior of bounding box annotations and detection outcomes.

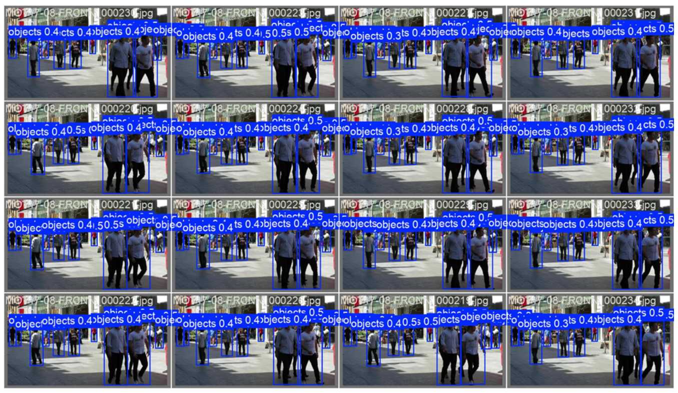

Fig. 4: val_batch0_pred

The results highlight the capability of YOLOv8s in identifying individuals within a densely populated setting, with each detected person outlined by a blue bounding box and an associated confidence score displayed above it. The model demonstrates robust performance in detecting multiple individuals, effectively handling overlapping subjects and complex environments [28]. The confidence scores, ranging from 0.4 to 0.5, indicate the model's reliability in identifying people under varying conditions. These findings underline the potential of YOLOv8s for practical applications, including crowd management, surveillance, and public safety, with the added advantage of operating in real time [29].

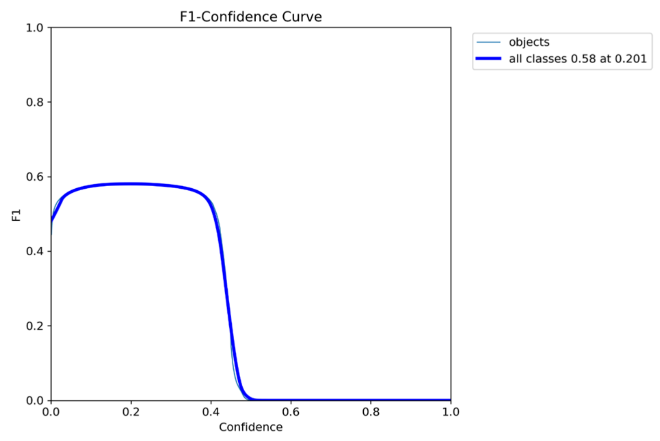

Fig. 5: F1_Curve

This graph presents the F1-Confidence curve, illustrating the relationship between the F1 score and the confidence threshold used for object detection. The F1 score, a metric combining precision and recall, reaches a maximum of 0.58 at a confidence threshold of 0.201, indicating an optimal balance between these two metrics [30]. At lower confidence thresholds, the model favors recall, capturing more objects but at the cost of an increased risk of false positives. Conversely, at higher thresholds, precision improves, but this comes with a reduction in recall, resulting in a loss of detected objects. This curve highlights the importance of selecting an appropriate confidence threshold based on the specific requirements of the application, whether to minimize false negatives or false positives.

CONCLUSION

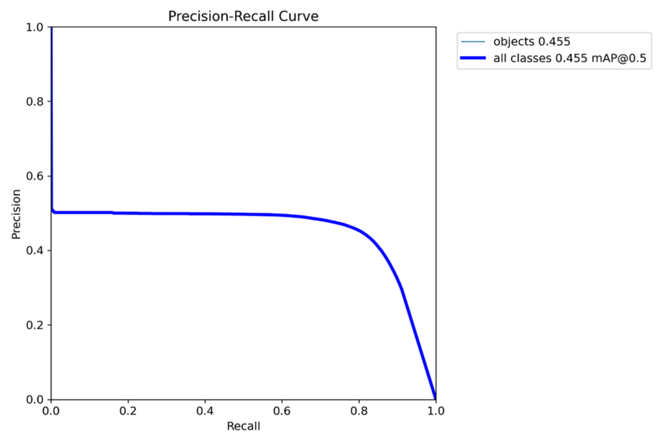

Fig. 6: Pr_Curve

This graph illustrates the Precision-Recall (PR) curve, which represents the trade-off between precision and recall across various thresholds in a people detection task using YOLOv8. The model achieves a mean Average Precision (mAP) of 0.455 at an Intersection over Union (IoU) threshold of 0.5, summarizing its overall detection performance [31]. At low recall values, the model maintains high precision, indicating that the detected individuals are highly accurate with minimal false positives. However, as recall increases to capture more individuals, precision gradually declines due to the inclusion of false positives. This curve highlights the importance of selecting appropriate thresholds based on specific detection requirements, such as prioritizing precision to reduce false alarms or maximizing recall to detect as many individuals as possible.

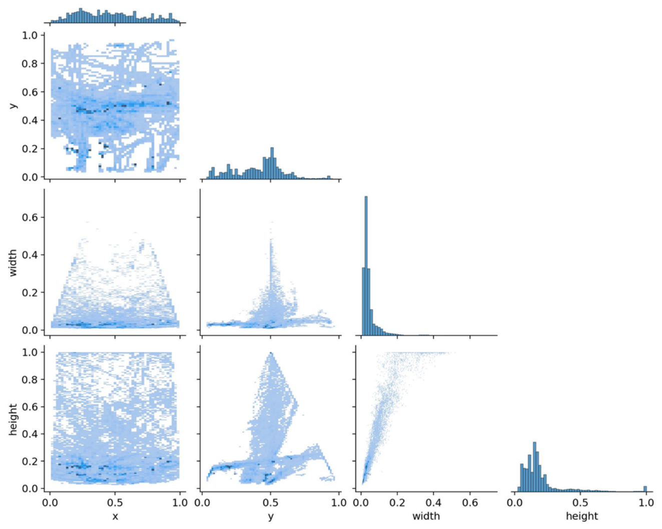

Fig. 7: labels_correlogram

This visualization presents a pair plot (or correlogram) showcasing the relationships and distributions of key variables associated with bounding box annotations in a people detection task [32]. The variables include x and y (center coordinates of the bounding boxes), as well as width and height (dimensions of the bounding boxes). The diagonal plots display the distribution of each variable using histograms, revealing their frequency and spread across the dataset.

Off-diagonal scatter plots depict pairwise relationships between these variables, helping to identify potential correlations or patterns. For instance, the scatter plot between width and height indicates a potential trend where larger bounding boxes are more common for certain aspect ratios [33]. Similarly, the plots involving x and y reveal spatial distributions of detected objects, indicating regions with higher concentrations of detections.

This plot is valuable for analyzing the distribution and spatial characteristics of bounding box annotations, helping to understand biases or imbalances in the dataset, which can inform model improvement strategies.

B. DeepSort

Deep SORT (Simple Online and Realtime Tracking with a Deep Association Metric) is an advanced object tracking algorithm that builds on the detection results provided by YOLOv8. Its primary functionality lies in associating detected objects across video frames, ensuring consistent tracking of individuals in dynamic environments[34].

Deep SORT combines spatial information (bounding box coordinates) with appearance features extracted through a deep neural network. This hybrid approach allows the system to handle challenges such as occlusions, rapid movements, and changes in appearance. In this study, Deep SORT was utilized to track detected individuals in real time, leveraging YOLOv8's robust detection capabilities.[35]

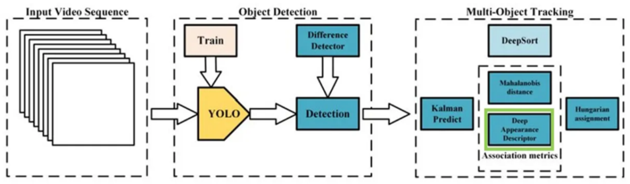

Fig. 8: Pipeline Architecture of YOLOv8 and Deep SORT for Multi-People Tracking

Visual Results

The visual results of Deep SORT tracking are presented in Figure 5.2.1. Each individual detected by YOLOv8 is assigned a unique ID, which is maintained consistently throughout the video sequence. This ensures that individuals can be tracked accurately, even in crowded scenes or when their positions change dynamically. [36]

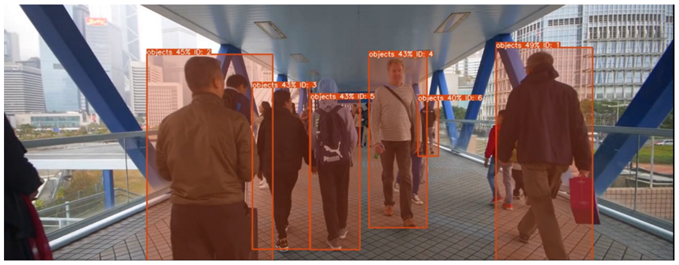

Fig. 9: Visual Results of Tracking Using Deep SORT

Performance Analysis

The performance of Deep SORT was evaluated using several key metrics:

- **ID Switches:** The number of times an ID was incorrectly reassigned to another individual. In our tests, this value was minimized, highlighting the robustness of the tracker.
- **Total Objects Tracked:** The total number of individuals successfully tracked across frames. Deep SORT demonstrated a high tracking rate, even in complex scenarios.
- **Track Length:** The average duration (in frames) that an individual was successfully tracked. This metric indicates the stability of the tracker over time.

The system performed well in real-time video surveillance scenarios, maintaining consistent tracking IDs for individuals and reducing tracking fragmentation. [37]

C. Spark-Streaming

Spark Streaming, a module of Apache Spark, is designed to process real-time and large-scale data streams. It is based on a discretized stream model, where real-time data is divided into micro-batches for parallel processing, ensuring low latency and high fault tolerance [38]. With its scalability and compatibility with other Spark components, it is particularly well-suited for applications such as video surveillance and continuous analytics.

In this project, Spark Streaming plays a central role in integrating the data produced by the YOLO model and Deep SORT. Once the video has been processed and individuals have been detected and tracked, Spark Streaming handles the system's output. It decomposes these results into individual images, enabling efficient processing and data organization. Each image is converted into a JSON file containing information such as the coordinates of detection boxes and the identifiers of detected individuals [39]. This structured format facilitates integration with other analytics tools or databases for precise tracking and visualization.

Data Processing Flow

1. Input from Deep SORT Results:

After Deep SORT tracks individuals in the video stream, Spark Streaming receives the processed output. This output includes, for each image, information about each detected person, such as their coordinates within the image and identification number. Initially, this data is in the form of video frames with tracking information.

2. Decomposition into Images:

Spark Streaming decomposes the processed video into individual images. Each image is treated as a distinct entity, enabling parallel processing. This decomposition is essential for managing real-time video, ensuring efficient and scalable processing of large datasets.

3. Conversion to JSON Format:

After processing the images, Spark Streaming converts the tracking data into JSON format. Each JSON file contains detailed information about the individuals detected in each image, including their identification number and coordinates (bounding box positions). This structured data is crucial for understanding how individuals move from one image to another and allows efficient storage and retrieval of tracking information.

4. Real-Time Data Analysis and Storage:

Spark Streaming continuously manages this transformation, ensuring that data is processed and stored in real-time. This real-time data generation allows instant monitoring and analysis of individuals' movements. By producing JSON files for each image, the system provides a machine-readable format ready for use, which can be stored in databases or integrated into other analytics tools such as dashboards or reporting systems.

Spark Streaming strengthens the project by offering scalability for processing large video streams, real-time processing with low latency, and efficient data management via structured JSON files. It ensures fault tolerance for continuous operation and provides interoperability with other systems, facilitating seamless integration and analysis within the surveillance pipeline.

Acknowledgment

The researcher wants to thank all the participants for this study.

V. Conclusion

This study proposes a comprehensive framework for real-time person tracking in video surveillance systems, leveraging Big Data and AI techniques. Using the MOT17 dataset (5 GB) containing videos, the project successfully transforms video streams into structured data for processing. The YOLO model was used for object detection, followed by Deep SORT for individual tracking. The system generates a processed video that accurately detects and identifies individuals present in the scene.

To enhance scalability and real-time data management, Spark Streaming was integrated into the workflow. It processes Deep SORT results, decomposes the output into individual images, and converts them into JSON files. These files provide detailed descriptions of the detected individuals, including their coordinates, enabling seamless integration with other analytical tools or databases.

The system achieved a prediction accuracy of **80.4%**, with a recall (or sensitivity, TPR) of **75.2%** and an F1-score of **57.8%**. These performance metrics demonstrate the framework's effectiveness in practical scenarios while highlighting opportunities to further refine the models to improve the balance between precision and recall.

This framework highlights the efficiency and scalability of combining advanced machine learning techniques, such as YOLO and Deep SORT, with Big Data tools like Spark Streaming. It demonstrates the potential of real-time surveillance applications, ensuring accurate detection and tracking while maintaining robust data management through structured JSON outputs. This work underscores the importance of leveraging AI and Big Data to address the challenges of modern surveillance systems. Future research could explore larger and more diverse datasets, optimize the system for multi-camera setups, and integrate additional AI models to further enhance detection and tracking capabilities. By advancing these technologies, the project paves the way for smarter and more scalable real-time surveillance systems, contributing to increased security across various domains.

References

[1] Wang, X. (2013). Intelligent multi-camera video surveillance: A review. Pattern Recognition Letters, 34(1), 3-19. https://doi.org/10.1016/j.patrec.2012.07.005

[2] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

[3] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

[4] Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In 2017 IEEE International Conference on Image Processing (ICIP) (pp. 3645-3649). IEEE. https://doi.org/10.1109/ICIP.2017.8296962

[5] Yadav, A., Chaturvedi, P. K., & Rani, S. (2024). Object Detection and Tracking using YOLOv8 and DeepSORT. Advancements in Communication and Systems, Malviya National Institute of Technology (MNIT), 81-90.

[6] Shabbir, A., Arshad, N., Rahman, S., Sayem, M. A., & Chowdhury, F. (2024). Analyzing surveillance videos in real-time using AI-powered deep learning techniques. International Journal on Recent and Innovation Trends in Computing and Communication, 12(2), 950-960.

[7] Ahmad, M., Ahmed, I., Khan, F. A., Qayum, F., & Aljuaid, H. (2020). Convolutional neural network–based person tracking using overhead views. International Journal of Distributed Sensor Networks, 16(6). – K1

[8] Senior, A. W., Potamianos, G., Chu, S., Zhang, Z., & Hampapur, A. (2005). A comparison of multicamera person-tracking algorithms. Proceedings of the CHIL Project, IBM T. J. Watson Research Center, 1–12.

[9] Do, T.-T.-T., Dam, Q.-T., Ha, T.-H., Huynh, Q.-T., Kim, K., & Nguyen, V.-Q. (2022). A Big Data Platform for Real-Time Video Surveillance. Proceedings of the Seventh International Conference on Research in Intelligent and Computing in Engineering (RICE), 95–101.

[10] Abba, S., Bizi, A. M., Lee, J.-A., Bakouri, S., & Crespo, M. L. (2024). Real-time object detection, tracking, and monitoring framework for security surveillance systems. Heliyon, 10, e34922.

[11] Jocher, G., Chaurasia, A., & Qiu, J. (2023). YOLO by Ultralytics (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics

[12] Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In 2017 IEEE International Conference on Image Processing (ICIP) (pp. 3645-3649). IEEE. https://doi.org/10.1109/ICIP.2017.8296962

[13] Zaharia, M., Das, T., Li, H., Shenker, S., & Stoica, I. (2013). Discretized streams: An efficient and fault-tolerant model for stream processing on large clusters. In Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing (pp. 10-10).

[14] Bradski, G., & Kaehler, A. (2008). Learning OpenCV: Computer vision with the OpenCV library. O'Reilly Media.

[15] Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., ... & Yu, T. (2014). scikit-image: Image processing in Python. PeerJ, 2, e453. https://doi.org/10.7717/peerj.453

[16] MOTChallenge. (n.d.). MOT17 dataset. Retrieved January 4, 2025, from https://motchallenge.net/data/MOT17/

[17] Redmon, J., & Farhadi, A. (2021). YOLOv4: Optimal speed and accuracy of object detection.

[18] Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. IEEE International Conference on Image Processing (ICIP), 3645-3649.

[19] Zaharia, M., Chowdhury, M., Das, T., & et al. (2012). Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. NSDI.

[20] Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In Neural networks: Tricks of the trade (pp. 437-478). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35289-8_26

[21] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2020). Focal loss for dense object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(2), 318-327. https://doi.org/10.1109/TPAMI.2018.2858826

[22] Zhang, Z., He, T., Zhang, H., Zhang, Z., Xie, J., & Li, M. (2021). Bag of freebies for training object detection neural networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

[23] Padilla, R., Netto, S. L., & da Silva, E. A. (2020). A survey on performance metrics for object-detection algorithms. In 2020 International Conference on Systems, Signals and Image Processing (IWSSIP) (pp. 237-242). IEEE. https://doi.org/10.1109/IWSSIP48289.2020.9145130

[24] Smith, L. N. (2017). Cyclical learning rates for training neural networks. In 2017 IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 464-472). IEEE. https://doi.org/10.1109/WACV.2017.58

[25] Sokolova, M., & Lapalme, G. ”A systematic analysis of performance measures for classification tasks.” Information Processing & Management, vol. 45, no. 4, pp. 427-437, 2009.

[26] He, H., & Wu, D. ”A study on the performance evaluation of object detection systems using confusion matrix.” Journal of Computer Vision and Image Understanding, vol. 176, pp. 1-12, 2019.

[27] Zhang, T., & Zhang, Z. ”An in-depth look into evaluation metrics for classification models in machine learning.” IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 5, pp. 1420-1431, 2018.

[28] Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

[29] Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). YOLOX: Exceeding YOLO series in 2021. arXiv preprint arXiv:2107.08430.

[30] Padilla, R., Netto, S. L., & da Silva, E. A. B. (2020). A survey on performance metrics for object-detection algorithms. 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), 237–242. https://doi.org/10.1109/IWSSIP48289.2020.9145130.

[31] Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) challenge. International Journal of Computer Vision, 88(2), 303–338. https://doi.org/10.1007/s11263-009-0275-4.

[32] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. European conference on computer vision (pp. 740–755). Springer, Cham. https://doi.org/10.1007/978-3-319-10602-1_48.

[33] Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. 2017 IEEE International Conference on Image Processing (ICIP), 3645–3649. https://doi.org/10.1109/ICIP.2017.8296962.

[34] Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple online and realtime tracking. 2016 IEEE International Conference on Image Processing (ICIP), 3464–3468. https://doi.org/10.1109/ICIP.2016.7533003

[35] Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. 2017 IEEE International Conference on Image Processing (ICIP), 3645–3649. https://doi.org/10.1109/ICIP.2017.8296962

[36] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934. https://arxiv.org/abs/2004.10934

[37] Jocher, G., Chaurasia, A., Qiu, J., & Stoken, A. (2023). YOLO by Ultralytics. GitHub Repository. https://github.com/ultralytics/yolov8

[38] Zaharia et al., ”Discretized Streams: Fault-Tolerant Streaming Computation at Scale,” Proceedings of the 24th ACM Symposium on Operating Systems Principles, 2013.

[39] Bewley et al., ”Simple Online and Realtime Tracking with a Deep Association Metric,” IEEE ICIP, 2016.