Received: September 23, 2024. Revised: October 25, 2024. 106

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Multiple Object Detection and Tracking Using Deep Learning Framework with Non-Maximum Suppression

Vinod Biradar[1]* **Karuna Chandrashekhar Gull[1]**

[1]*Department of Computer Science and Engineering, S. G. Balekundri Institute of Technology, Belagavi, Visvesvaraya Technological University, Belagavi-590018, Karnataka, India*
* Corresponding author's Email: vinnu151986@gmail.com

Abstract: Multiple object detection and tracking involves identifying and locating numerous objects within a sequence of images or video frames and maintaining their identities across frames. This process is significant for applications like surveillance and autonomous vehicles. However accurately detecting and tracking multiple objects in dynamic environments is challenging because of occlusions, overlapping trajectories, and varying object sizes which results in tracking failure and misidentification. In this research, the Convolutional Neural Network with Non-Maximum Suppression (CNN-NMS) is proposed to detect and track multiple objects accurately using the VGG16 model. Using a CNN-NMS enhances object detection by leveraging the CNN's ability to extract detailed features whereas NMS refined the outcomes by removing redundant overlapping bounding boxes. In CNN-NMS, the VGG16 is applied to extract features with incorporating linear layer and sigmoid activation which predicts class labels and bounding box offsets. The proposed CNN-NMS achieves a high MOTA of 80.05%, 85.72%, 79.12%, 81.27%, and 82.47% using MOT17, open image bus truck, MOT15, 16, and 20 datasets compared to existing methods like You Only Look Once Tracker and RetinaMOT.

Keywords: Bounding boxes, Convolutional neural network, Multiple object detection and tracking, Non-maximum suppression, VGG16.

1. Introduction

Multi-Object Tracking (MOT) is a significant research direction in computer vision as it involves identifying, classifying, and tracking various objects in a video and relating their trajectories to establish an entire sequence. This process not only involves detecting objects in every frame but also defining which objects are to be tracked [1]. To achieve effective tracking, it is essential to accomplish the data association in subsequent frames and relate the motion trajectories of the detected objects to evaluate their category [2]. Typically, MOT combines approaches like trajectory prediction, data association, and object detection to effectively track and handle multiple objects in a video sequence. Due to the developments in object detection, most tracking approaches follow a two-phase tracking-by-detection method [3]. This method decouples the tracking and detection tasks by presenting a tailored approach for each which results in a more robust MOT system [4, 5]. The primary goal of MOT is to determine the moving object state and allocate a unique identifier to each moving object [6, 7]. MOT is executed by offline or online processing to determine the trajectories. Online MOT detects objects in the present frame and directly produces trajectories while the near-online model looks ahead for a few frames before associating the objects with trajectories. For offline trajectories, a mini-batch detection is created and managed recursively to finalize trajectories [8].

In the MOT model, rich representations are significant to recognize the visual data as they are derived from layered networks [9]. This tracking ability is vital to provide early warning of abnormal behaviour or aid in efficient vehicle control. Moreover, pedestrian targets are influenced by

various factors like changes in object occlusion, posture variations, and external environment [10]. Initially most of the model detect objects by acquiring their bounding boxes in the present frame [11]. Then, the Reidentification (ReID) features are extracted from every bounding box to match the candidate box with the existing trajectory [12, 13]. In complex scenes, objects are covered by other entities which impedes accurate tracking. Also, changes in illumination affect object attributes which leads to a decline in tracker performance [14]. Examining object tracking and detection increases the capability to manage and understand dynamic scenes. This enhances the accuracy of object identification and tracking across frames which is significant for applications in autonomous vehicles and surveillance. Consequently, it makes more efficient resource allocation and decision-making by generating detailed insights into object behaviour and interactions [15]. However accurately detecting and tracking multiple objects in dynamic environments is difficult due to occlusions, overlapping trajectories, and varying object sizes which leads to tracking failure and misidentification. To solve this issue, the CNN-NMS is proposed for multiple object detection and tracking using the VGG16 model. The CNN effectively detect multiple objects whereas NMS refines this detection by removing the redundant overlapping boxes which enhances tracking accuracy and minimizes misidentification.

The main contribution of this research is represented below:

- In VGG16, the linear layer with a sigmoid activation function is included to extract the features that predict the class labels and bounding box offsets.
- CNN-NMS enhances object detection accuracy by efficiently removing redundant bounding boxes which preserves the most appropriate features.
- Selection search efficiently generates region proposals that focus on potential object position which assists in minimizing the search space for detection. It enhances the accuracy and speed of the detection approach by generating more appropriate candidate regions.

This research paper is organized as follows: Section 2 provides a literature survey of the existing methods and Section 3 illustrates the proposed methodology. Section 4 generates an experimental set-up, and Section 5 explains the overall conclusion.

2. Literature survey

The related work about object detection and tracking using the MOT17 dataset was explained in detail with their findings, advantages, and limitations.

Chan [16] implemented a You Only Look Once Tracker (YOLOTracker) to perform object detection and tracking. Initially, an effective joint detection and tracking approach was determined to obtain instance-level embedding training which achieves high efficiency. Then, the path aggregation network was applied to integrate low-and-high resolution features in semantic and textual data which reduced the ReID feature's miss-alignment. Nevertheless, the YOLOTracker struggles with its lower performance in managing small or fast-moving objects because of its dependence on single-frame object detection which leads to suboptimal performance.

Cao [17] presented a Rethinking anchor-free YOLO5 MOT (RetinaMOT) to track multiple objects. The retinal convolution was used for extracting the features which enhances the model performance. Then, the adaptive cascade pyramid was applied to align features when compensating for the pyramid set issues. The interaction significance of cross-latitude data and location data in attention was verified effectively using the presented approach. However, because of its difficulty in managing occlusions and overlapping objects, the RetinaMOT lead to inaccurate tracking of multiple objects particularly in crowded scenes.

Alagarsamy and Muneeswaran [18] introduced a Reptile Search Optimization Approach with Deep Learning-based Multi-Object Detection and Tracking (RSOADL-MODT) to detect and track the objects. Initially, the introduced approach employed a Path-Augmented RetinaNet (PA-RetinaNet) model to extract the features and a quasi-Recurrent Neural Network (QRNN) was applied for the detection process. The RSOA was used as a hyperparameter optimizer which enhance the PA-RetinaNet's network potential effectively. Nevertheless, the RSOADL-MODT struggled with long-term object tracking when there were significant appearance changes or sudden movements.

Gao [19] suggested a DetTrack to enhance occlusion object detection for MOT. While the object was fully occluded, the object motion track box prediction was applied as a hypothesis box to continue the tracking process. The spatial-temporal features were integrated with track prediction to solve object detection in tracking performance which assists in reestablishing low-scoring object detections and improved data associations. However, track management struggles with frequent occlusions due

to it depends on visual continuity to manage object identities which get disrupted while objects are obscured.

Wang and Mariano [20] developed a YOLOv8s Symmetric Definite Convolution (YOLOv8s SPD) to detect and track the object. The detector was integrated with the ByteTrack tracking approach, Intersection of Unit (IoU), and Binary Cross-Entropy loss function (BCELoss) to obtain a high performance. The fixed retention frame number was adopted in the track deletion process in ByteTrack approach. Nevertheless, the YOLOv8s SPD struggled with managing complex object shapes and fine-grained information because it depends on a symmetric convolutional process which minimized its ability to capture objects effectively.

In the overall analysis, the existing method had limitations like struggle with its lower performance in managing small or fast-moving objects, inaccurate tracking of multiple objects particularly in crowded scenes, struggle with occlusions, overlapping trajectories, and varying object sizes. To solve this issue, the CNN-NMS is used to detect and track the objects effectively which enhances the detection of small and fast-moving objects. NMS refines the outcomes by eliminating redundant overlapping detection. This process increases accuracy in crowded scenes, handling occlusions, and varying object sizes.

3. Proposed methodology

In this research, the CNN-NMS is proposed to detect and track the objects effectively. Initially, open image bus truck, MOT15, 16, 17, and 20 datasets are used to determine mode performance. The selective search is applied to determine potential objects which reduces the search space and VGG16 is performed to extract the features accurately. At last, CNN-NMS is used for object detection and tracking. Fig. 1 depicts the block diagram for the proposed method.

3.1 Datasets

The MOT17 [21], open image bus truck [22], MOT15 [23], MOT16 [24], and MOT20 [25] datasets are used to determine the model performance. MOT17 is a widely utilized dataset that contains 7 sequences for training and the remaining 7 for testing sets. Also, it involves a crowded street scene with the motion of a linear object. The dataset highlights tracking multiple objects by different challenges like varying lighting conditions, occlusions, and various camera angles. The open-image bus truck dataset has 627 images of various vehicle classes for detecting objects. Each frame in the sequences is annotated with object IDs and bounding boxes for pedestrians.

MOT15 is a standard benchmark that presents a fair test for MOT evaluation and involves 11 videos for the performance of tracker analysis. MOT16 contains a 7 training and 7 testing video sequence with numerous challenging tasks like target occlusion and interactions. MOT20 involves an overall of 8 videos and has 13,410 frames. The number of video frames for testing and training are 4479 and 8931 frames. Then, the selective search process is applied in the pre-processing stage.

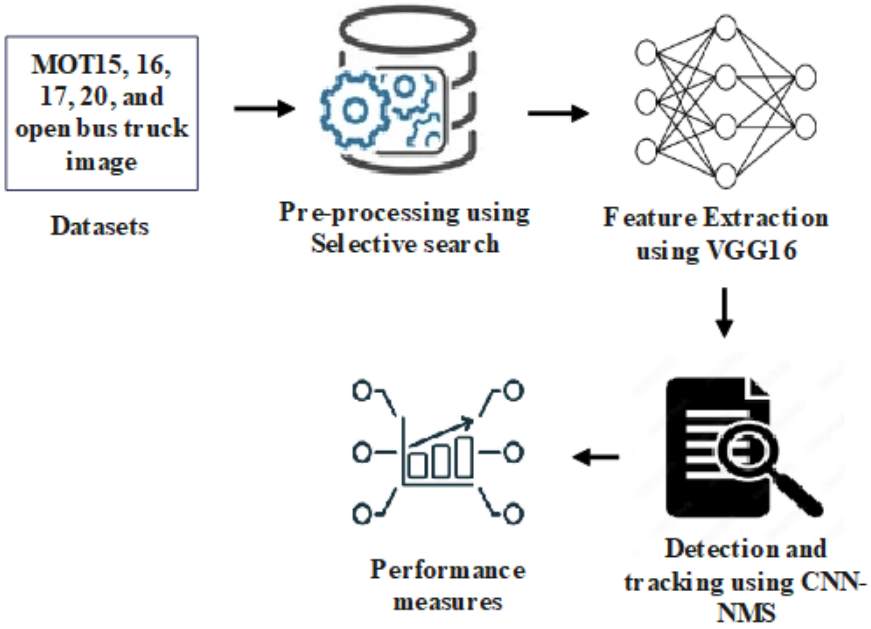

Figure. 1 Block diagram for the proposed method

Open image bus truck

MOT17

Figure. 2 Sample images for datasets

3.2 Pre-processing

After obtaining the images, the pre-processing is performed using a selective search [26] method which generates region proposals by grouping pixels depending on similarities such as size, scale, texture, and color. These region proposals are then utilized to determine potential objects, minimizing the search space and enhancing the detection approach's efficiency. In this method, four similarities are defined and explained below

a) Color similarity $s_{col}(r_i, r_j)$: In this similarity, L_1 is used to normalize the color histograms. Every region r_i with dimensionality contained a color histogram $C_i = \{c_i^1, \dots, c_i^n\}$ which is represented using Eq. (1).

$$s_{col}(r_i, r_j) = \sum_{k=1}^{n} min\ (c_i^k, c_j^k\} \quad (1)$$

b) Texture similarity $s_{tex}(r_i, r_j)$: The texture histograms $T_i = \{t_i^1, \dots, t_i^n\}$ are normalized by utilizing the L1 norm which is expressed in Eq. (2).

$$s_{tex}(r_i, r_j) = \sum_{k=1}^{n} min\ (t_i^k, t_j^k\} \quad (2)$$

c) Size similarity $s_{size}(r_i, r_j)$: It primarily focuses on small regions to combine and $size(im)$ and represents image size in pixels which is formulated in Eq. (3).

$$s_{size}(r_i, r_j) = 1 - \frac{size(r_i)+size(r_j)}{size(im)} \quad (3)$$

d) Fill similarity $s_{fill}(r_i, r_j)$: It determines how effectively fits the region r_i and r_j into each other. The BB_{ij} represents a tight bounding box over r_i and r_j is indicated using Eq. (4).

$$s_{fill}(r_i, r_j) = 1 - \frac{size(BB_{ij})-size(r_i)-size(r_j)}{size(im)} \quad (4)$$

Where c_i^k and c_j^k represents color histogram for regions r_i and r_j for k^{th} color channel, and n determines the total number of colour channels, and t_i^k, t_j^k indicates texture histogram. At last, the final similarity measure is an integration of colour, texture, size, and fill similarities are formulated using Eq. (5).

$$s(r_i, r_j) = \varepsilon_1 s_{col}(r_i, r_j) + \varepsilon_2 s_{tex}(r_i, r_j) + \varepsilon_3 s_{size}(r_i, r_j) + \varepsilon_4 s_{fill}(r_i, r_j) \quad (5)$$

Where $\varepsilon\ \epsilon$ (0,1) determines a constant number. The selective search produces a possible object position set in an object detection model. Also, it effectively minimizes the number of candidate regions by combining similar pixels which enhances computational speed. Then, the pre-processing input is fed into the extraction process.

3.3 Feature extraction

After pre-processing, the VGG16 [27] is performed to extract the features which is a significant stage for object detection and tracking. VGG16 can extract a large number of data and provides a better performance in MODT. Different layers are used in the entire model which is convolutional, Batch Normalization (BN), and pooling along with Rectified Linear Unit (ReLU) and SoftMax function to establish the entire framework. The network's input image size is fixed to 224 * 224 pixels with a 3 *3 filter size. Then, the images are fed into the number of convolutional layers and then passed via an initial stack of 2 convolutions. The stride and padding values are fixed into 2. The entire spatial resolution is reserved, and the resulting dimension activation is matched with source images. Hence, the activations are managed to a pooling layer with a 2 x 2 window size, and then those size becomes

half. At the end of the initial stack, the outcomes are 112 *112 *64 in size. Moreover, the outcome is again fed into convolutional layers in the second phase which is then fed into the pooling layer and results in 56 * 56* 128 pixels. This procedure continues until the final stage of pooling and convolutional layer. A flattening layer is located among the 3 Fully Connected (FC) layers which are utilized after a convolution stack. The features are fetched after passing the normalized image via a pre-trained approach. A linear layer with sigmoid activation is attached to the VGG16 model to predict the class corresponding to the region proposal. An additional linear layer is applied to determine the four bounding box offsets. Its pre-trained model on MODT provides a robust generalization for different objects. Also, VGG16's simplicity and effective feature representation enable smoother integration into detection and tracking systems. Then, the extracted features are fed into the detection and tracking process for further processing.

3.4 Detection and tracking

The extracted features are fed into the CNN-NMS method to detect and track the objects. Using CNN-NMS for object detection and tracking enhances performance by removing redundant and overlapping bounding boxes and ensures that only the most appropriate detection is retained. This combination improves the accuracy of detecting distinct objects, minimizes false positives, and enhances tracking of moving objects by focusing on the detections. NMS ensures computational efficiency by filtering out the unnecessary detections which makes the overall system more effective. CNN [28] is a kind of feed-forwarding neural network used for visual images and has various layers like input, BN, pooling, activation, FC, and output layers which are explained below in detail.

Input and Convolutional layer: The input layer is used to insert images into a network and then the convolutional layer convolutes the input data or from the prior layer by shifting a filter to generate a feature map. This progress produces various feature maps for a better understanding of the image. The mathematical formula for the convolution process is expressed in Eq. (6).

$$\tilde{Z} = f(W\hat{Z} + b) \tag{6}$$

Where $\tilde{Z}$ represents convolution process output, W denotes weights, $f(.)$indicates activation function, and b and determines bias. The convolutional layer weight is experienced on update progress to enhance the outcome of image classification in a training phase. The update progress in training is completed on all weights in every convolutional layer. A set of weights used for a region in the image is known as a filter which is formulated in Eq. (7).

$$a\ W_{ij} = U\left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right] \tag{7}$$

Where U determines uniform distribution, n denotes the previous layer size. The number of layers contained in the initial layer is 30, 2nd is 60, and 3rd is 120 whereas the filter size is 5 X 5 for each layer. The padding and stride are determined in the convolution process.

BN: It is used to normalize each input channel in a mini-batch and increase the training on CNN to minimize the initialization of the network. BN is applied among convolution and ReLU which helps to attain the desired outcomes. The initial layer of a process is to normalize every channel's activation by subtracting every channel from the average mini-batch and dividing them by the mini-batch standard deviation.

ReLU and pooling: An activation layer utilizes an unsaturated activation function to increase the nonlinear decision-function nature. Here, ReLU is employed using Eq. (8).

$$\widetilde{Z_R} = \begin{cases} \hat{F}, & \hat{F} \geq 0 \\ 0, & \hat{F} < 0 \end{cases} \tag{8}$$

Where $\hat{F}$ indicates convolutional process output. Then, the pooling layer is used to minimize the spatial representation size. Also, it reduces the number of calculations and manages overfitting. The max-pooling function is used which divides the input layer into rectangular sets with an output of maximum region.

FC and Output: In this layer, the feedback progress is executed by refining the prior layer's bias and weight. It minimizes the feature information loss and the outcome of this layer is fed into the output layer for detection. The output layer represents detection outcome and the most used activation functions are sigmoid and SoftMax. In this research, the SoftMax function is applied as an output layer which is formulated in Eq. (9).

$$y_k(Z^*) = \frac{exp\ (Z_k^*)}{\sum_{j=1}^{C} exp\ (Z_j^*)}, k = 1, \ldots, C \tag{9}$$

Where y_k represents SoftMax output in class k, Z^* indicates FC layer output, and C determines the number of classes (labels).

This research eliminates NMS in detection and hence each detection boxes acquired by tracking are utilized as a detection outcome for tracking. In the tracking phase, the detection boxes compute IOU based on the position predicted by the Kalman filter. A detection box with an IOU greater than the threshold is selected as a candidate box. ReID is then utilized to determine if it matches an existing target. Whether it is not a match, NMS is employed to filter out a box with error detection. Then, a detection box that is either an existing target or an unfiltered one is observed as a new target and allocated a new ID. The input frame is fed as input to the detection model and acquire a detection series boxes $B_t = \{B_t^1, B_t^2, B_t^3, , \ldots, B_t^n$. Between them $B_t^n(x_1, y_1, x_2, y_2)$ indicates i^{th} the detection frame and forecast, the target place is represented as $P(x_1', y_1', x_2', y_2')$. Based on the coordinates of predicted position and detection box, the IOU value c_i is computed which is formulated in Eq. (10-11).

$$c_i = \frac{Intersection(B_t^i, P)}{Union(B_t^i, P)} \tag{10}$$

$$\widehat{B_t} = \{B_t^i \, \epsilon B_t | c_i > \theta\} \tag{11}$$

Based on the computed detection box c_i and predicted location, boxes with an IOU value smaller than the IOU threshold θ are filtered out, and those $\widehat{B_t}$ closer to the predicted location are selected. Then, the NMS is performed on $\widehat{B_t}$ box uses Eq. (12) which filters out inappropriate detection boxes and acquires last candidate box $\widehat{B_t}$ a candidate choice to match the trajectory.

$$B_t' = NMS(\widehat{B_t}) \tag{12}$$

Fig. 3 represents the sample image for detected objects. Using CNN with NMS increases object detection and tracking by minimizing overlapping bounding boxes which ensures more accurate localization. NMS reduces the occurrence of multiple detections for similar objects which improves the reliability and efficiency of tracking systems. CNN effectively detects objects from images whereas NMS refines the output to reduce false positives which leads to more accurate detection. This combination increases the system performance like video surveillance in detection and tracking.

4. Experimental results

The proposed CNN-NMS is simulated using a Python 3.7 environment with 16 GB RAM, an Intel core i7 processor, and a Windows 10 (64 bit) operating system. The performance metrics like Multiple Object Tracking Accuracy (MOTA), Identification F1-score (IDF1), Mostly Tracked Targets (MT), Most Lost Target (ML), ID Switches (IDS), and Hz are used to determine the model performance. The mathematical formula for MOTA and IDF1 are represented using Eqs. (13) and (14).

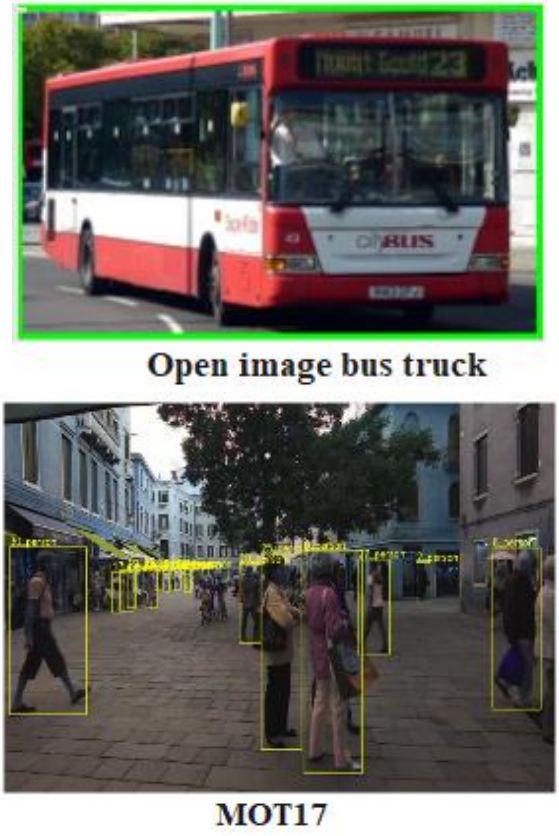

Figure. 3 Sample images for detected objects
Bottom of Form

- MOTA – It is a measure of total performance of a tracker for MOT using equation (13)

$$MOTA = 1 - \frac{\sum_t (FP_t + FN_t + IDS_t)}{\sum_t GT_t} \tag{13}$$

- IDF1 – It is a ratio of accurately identified detection over average number of ground truth and calculated detections using equation (14)

$$IDF1 = \frac{2 \times IDP \times IDR}{IDP + IDR} \tag{14}$$

- MT – It is a high integrity tracking that defines the proportion of tracks with a tracking length of over 80%

- ML – It is a high missing tracking degree that represents the trajectories proportion with tracking length less than 20%
- IDS – It calculates the total number of transition ID during the tracking process
- Hz – Processing speed in Frames Per Second (FPS).

Where FP_t and FN_t indicates False Positive and False Negative at time step t, GT_t represents Ground Truth at time t, as well as IDP and IDR defines accurate rate and recall rate of identity.

4.1 Performance analysis

Table 1 represents a performance analysis of different detection methods. The existing techniques like YOLO, Faster Region-based CNN (R-CNN), and EfficientNet are compared with the proposed CNN-NMS. When compared to these methods, the CNN-NMS achieves a better MOTA of 80.05%, 85.72%, 79.12%, 81.27%, and 82.47% using MOT17, open image bus truck, MOT15, 16, and 20 datasets due to CNN-NMS having the ability to more effectively suppress redundant bounding boxes which results in reduced FP. Also, it maintains accurate localization and minimizes overlap errors effectively.

Table 2 depicts a different detection method without NMS. CNN without NMS achieves less performance due to it producing multiple overlapping bounding boxes for the same object. NMS is significant for eliminating these redundant detections by choosing the most appropriate bounding box. Without NMS, the approach output contains various overlapping boxes which results in duplicated detection and minimized overall performance in object detection and tracking. Therefore, NMS is essential to use with CNN to achieve better performance.

Table 3 determines a performance analysis of different feature extraction methods. The existing techniques like ResNet, MobileNet, and DenseNet are compared with VGG16 which achieves a better performance. Due to VGG16's deep architecture and consistency, it effectively captures complex features and patterns in images which obtains robustness and reliability. Also, it captures detailed hierarchical features and spatial patterns effectively which distinguish objects and maintain tracking accuracy effectively which leads to better MOTA of 80.05%, 85.72%, 79.12%, 81.27%, and 82.47% using MOT17, open image bus truck, MOT15, 16, and 20 datasets compared to existing methods.

Table 4 determines a performance analysis of different detection methods without selective search. Without using selective search, CNN-NMS performs less performance because selective search generates

Table 1. Performance analysis of different detection methods

Methods	Datasets	MOTA (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	IDS ↓	Hz ↑ (FPS)
YOLO	MOT17	75.32	74.21	56.12	17.56	2354	32.25
	Open Image Bus Truck	72.56	71.32	54.78	15.67	2332	30.43
	MOT15	73.45	72	42.1	14.5	1200	29.5
	MOT16	76.78	75.3	57.2	16.25	1800	31.1
	MOT20	74.89	73.98	52.9	15.3	1500	31
Faster RCNN	MOT17	68.45	66.8	53.67	15.84	2201	29.4
	Open Image Bus Truck	76.24	75.12	58.15	14.12	2228	29.78
	MOT15	70.55	69.45	45.75	13.9	1100	27.5
	MOT16	79.12	78.04	56.45	15.7	1250	32
	MOT20	72.35	71.68	51.8	11.2	1412	25.5
EfficientNet	MOT17	70.9	69.15	54.2	14.45	2123	30
	Open Image Bus Truck	82.11	81.54	63.49	10.98	1902	40.76
	MOT15	76.65	75.9	47.15	13.6	850	29
	MOT16	78.9	77.1	57.45	14.3	1150	33.2
	MOT20	81.2	80.4	62	12.1	1500	36.8
CNN-NMS	MOT17	80.05	79.65	58.32	11.21	1983	40.02
	Open Image Bus Truck	85.72	84.01	63.33	9.51	1848	43.89
	MOT15	79.12	78.54	50.75	7.36	910.5	39.7
	MOT16	81.27	80.29	61.48	13.09	1100.5	36.8
	MOT20	82.47	82.13	65.03	7.59	1403.2	28.6

Table 2. Different detection methods without NMS

Methods	Datasets	MOTA (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	IDS ↓	Hz ↑ (FPS)
YOLO	MOT17	63.3	62.15	47.12	18.56	2421	25.2
	Open Image Bus Truck	70.42	69.56	52.41	17.34	2387	27.85
	MOT15	61.5	60.45	46.12	17.5	1200	24.3
	MOT16	62.8	61.15	49.02	16.8	1250	25
	MOT20	63.8	62	50.7	17.2	1500	26
Faster RCNN	MOT17	65.5	64.4	50.67	17.2	2302	28.1
	Open Image Bus Truck	77.92	76.48	59.23	14.67	2167	31.89
	MOT15	61	60.05	47.5	16.2	1100	27.4
	MOT16	62	61.1	50.67	15.6	1150	28
	MOT20	63	62.15	52	14.8	1600	27.7
EfficientNet	MOT17	67.85	66.7	52.3	15.8	2153	30.5
	Open Image Bus Truck	79.43	78.32	60.87	13.23	2135	33.12
	MOT15	64.45	63.15	49.8	14.1	1100	28.9
	MOT16	65	64.15	51	15.9	1150	29.5
	MOT20	67.5	66.2	53.3	12.7	1700	28
CNN	MOT17	78.54	76.22	54.36	12.56	2001	39.75
	Open Image Bus Truck	80.32	79.87	60.33	12.51	2148	39.89
	MOT15	79.12	78.54	50.75	7.36	910.5	39.7
	MOT16	81.27	80.29	61.48	13.09	1100.5	36.8
	MOT20	82.47	82.13	65.03	7.59	1403.2	28.6

Table 3. Different feature extraction methods

Methods	Datasets	MOTA (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	IDS ↓	Hz ↑ (FPS)
ResNet	MOT17	68.3	67.5	52.12	15.56	2234	29.5
	Open Image Bus Truck	74.83	73.67	56.34	15.72	2245	29.12
	MOT15	69.45	68.12	43.89	14.91	1000	25.8
	MOT16	72.36	71.04	54.65	14.75	1250	28.6
	MOT20	75.78	74.34	55.19	16.04	1800	27.4
MobileNet	MOT17	76.45	75	50.67	16.3	2311	28.4
	Open Image Bus Truck	81.76	80.23	62.18	11.43	1923	39.94
	MOT15	77.32	76.15	46.22	12.67	1100	30.15
	MOT16	78.95	77.85	60.15	13.89	1350	31.8
	MOT20	80.1	79.54	63.01	11.67	1450	26.5
DenseNet	MOT17	70.9	69.8	54.2	14.8	2107	32
	Open Image Bus Truck	78.67	77.42	60.01	12.56	2109	33.67
	MOT15	73.88	72.45	49.56	13.22	1150	30.4
	MOT16	79.34	78.12	61.34	13.71	1250	34.2
	MOT20	82.01	81.23	64.67	10.12	1420	24.5
VGG16	MOT17	80.05	79.65	58.32	11.21	1983	40.02
	Open Image Bus Truck	85.72	84.01	63.33	9.51	1848	43.89
	MOT15	79.12	78.54	50.75	7.36	910.5	39.7
	MOT16	81.27	80.29	61.48	13.09	1100.5	36.8
	MOT20	82.47	82.13	65.03	7.59	1403.2	28.6

high-quality object proposals by focusing on meaningful regions of the image. This process significantly minimizes the number of candidate regions on CNN. Without selective search, CNN-NMS relies on less refined proposals which leads to a high number of FP and missed detections which impacts the overall performance. Hence, the selective search is used as the pre-processing step to determine potential objects which minimizes the search space and enhances the detection approach's efficiency.

Table 4. Different detection methods without selective search

Methods	Datasets	MOTA (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	IDS ↓	Hz ↑ (FPS)
YOLO	MOT17	74.2	73.1	55.12	16.9	2458	26
	Open Image Bus Truck	75.95	75.18	57.12	13.89	2190	31.03
	MOT15	72.88	71.44	48.01	15.67	1300	29.9
	MOT16	73.55	72.05	52.78	15.2	1400	28.75
	MOT20	74.33	73.6	55.56	16	1450	25.2
Faster RCNN	MOT17	66.8	65.75	50.9	15.5	2258	28.6
	Open Image Bus Truck	82.03	81.37	63.21	10.75	1897	40.82
	MOT15	64.8	63.05	48.5	14.2	1200	27.4
	MOT16	65.35	64.1	51.15	14.75	1250	29.15
	MOT20	66.1	65.2	52.6	13.6	1600	26.5
EfficientNet	MOT17	69.45	68.5	53.1	14.2	2185	31.5
	Open Image Bus Truck	71.84	70.91	53.78	16.12	2365	27.43
	MOT15	66.55	65.3	43.8	14	1150	29
	MOT16	67.25	66.1	52.3	13.75	1180	30.1
	MOT20	69	68.25	54.5	12.9	1520	24.7
CNN-NMS	MOT17	76.05	76.65	43.32	15.21	2783	38.02
	Open Image Bus Truck	81.72	76.01	36.33	14.51	2548	37.89
	MOT15	79.12	78.54	50.75	7.36	910.5	39.7
	MOT16	81.27	80.29	61.48	13.09	1100.5	36.8
	MOT20	82.47	82.13	65.03	7.59	1403.2	28.6

Fig. 4 represents the analysis of training loss and validation loss for open open-image bus truck. The graph depicts that the object detection and tracking model is learning efficiently from the training data. The training loss decreases constantly over epochs which demonstrates that the model is enhancing its capability to track and detect objects across time. The validation loss remains stable comparatively and maintains a better performance on unseen data. From the overall analysis, it obtains better performance and robustness in object detection and tracking tasks.

Fig. 5 determines the analysis of training and validation accuracy using an open image bus truck over five epochs. The training accuracy is increased which means that the model is better at generating correct predictions on training data. The validation accuracy evaluates the model performance on unseen data effectively. It represents that the model is better at predicting the training data and remains stable.

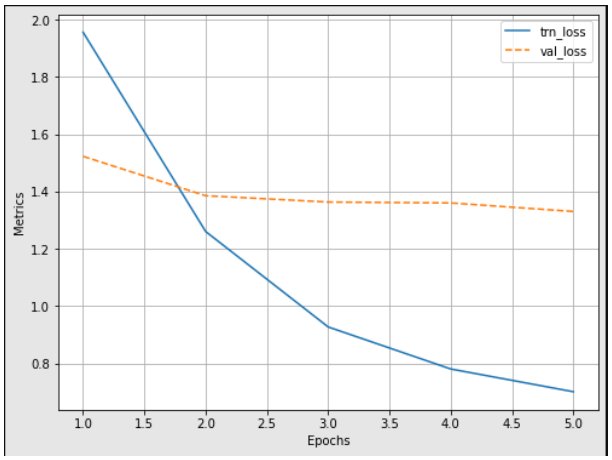

Figure. 4 Training and validation loss for open image bus trucks

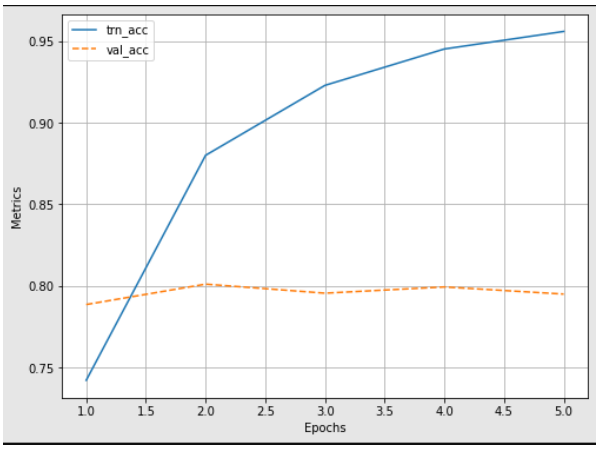

Figure. 5 Training and validation accuracy for open-image bus trucks

4.2 Comparative analysis

Table 5 denotes the comparative analysis of existing methods using MOT17 dataset. Table 6 presents the comparative analysis of existing ethods using MOT15, 16, and 20 datasets. The existing techniques like YOLOTracker [16], RetinaMOT [17], RSOADL-MODT [18], DetTrack [19], and YOLOv8s SPD [20] are compared with the proposed CNN-NMS method. The proposed CNN-NMS

Table 5. Comparative analysis with existing methods using the MOT17 dataset

Methods	MOTA (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	IDS ↓	Hz ↑ (FPS)
YOLOTracker [16]	67.1	65.1	41.50	19.20	4983	24.9
RetinaMOT [17]	74.1	70.9	41.1	19.4	3786	22.0
RSOADL-MODT [18]	74.6	72.3	N/A	N/A	N/A	N/A
DetTrack [19]	78.0	77.6	54.2	14.6	4878	35.5
YOLOv8s SPD [20]	74.0	75.1	42.3	23.1	2214	10.2
Proposed CNN-NMS	80.05	79.65	58.32	11.21	1983	40.02

Table 6. Comparative analysis with existing methods using the MOT15, 16, 20 datasets

Methods	Datasets	MOTA (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	IDS ↓	Hz ↑ (FPS)
YOLOTracker [16]	MOT15	56.3	57	46.60	11	1018	32.2
	MOT16	68.4	65.3	44.70	15.50	1632	24.9
RetinaMOT [17]	MOT16	74.8	73.2	41.5	18.3	1104	22.2
	MOT20	66.8	67.5	58.6	15.1	1739	18.6
DetTrack [19]	MOT16	78.0	78.3	54.7	14.1	1538	22.3
	MOT20	75.6	75.3	70.40	9.3	1640	16.9
YOLOv8s SPD [20]	MOT20	66.8	75.5	57.1	12.0	2337	21.3
Proposed CNN-NMS	MOT15	79.12	78.54	50.75	7.36	910.5	39.7
	MOT16	81.27	80.29	61.48	13.09	1100.5	36.8
	MOT20	82.47	82.13	65.03	7.59	1403.2	28.6

achieves a better MOTA of 80.05%, 79.12%, 81.27%, and 82.47% using MOT17, MOT15, 16, and 20 datasets because it minimizes FP by effectively suppressing redundant detections. This enhances the overall tracking performance by managing more reliable and consistent object tracking which results in higher MOTA scenes. Also, CN-NMS optimizes the balance between detection and tracking which provides a more robust tracking result.

4.3 Discussion

The advantages of the proposed CNN-NMS and the limitations of existing methods are explained in this section. The limitation of existing methods like YOLOTracker [16] struggle with their lower performance in managing small or fast-moving objects because of their dependence on single-frame object detection which leads to suboptimal performance. RetinaMOT [17] leads to inaccurate tracking of multiple objects particularly in crowded scenes. RSOADL-MODT [18] struggles with long-term object tracking under significant appearance changes or sudden movements due to its search optimization based on historical data. YOLOv8s SPD [20] struggled with managing complex object shapes and fine-grained information due to its dependence on the symmetric convolutional process which limited its ability to capture objects effectively. The proposed CNN-NMS addresses the limitations of existing methods by improving performance in object detection. CNN-NMS enhances object detection accuracy by efficiently managing small, fast-moving objects, and ensures better tracking performance in crowded scenes. Also, CNN-NMS enhance long-term tracking by adjusting to vital appearance changes and sudden movements. Furthermore, it captures intricate object shapes and fine-grained information which enables it more robust for handling diverse tracking scenarios.

5. Conclusion

This research proposed CNN-NMS for accurate object detection and tracking. Using CNN for initial object detections provides an advanced feature learning capability which leads to more accurate performance. NMS removed redundant overlapping bounding boxes and refined the object detection output model by minimizing FP which ensures that each object is demonstrated by a single bounding box. VGG backbone for extraction leverages the pre-trained weights to improve the model performance on region proposals. In VGG16, a linear layer with sigmoid activation is added which enables it for simultaneous prediction of class labels and bounding box offsets. This outcome ensures accurate detection and tracking of objects within images and enhances overall performance and reliability. By performing all these processes, the proposed CNN-NMS

achieves a better MOTA of MOTA of 80.05%, 85.72%, 79.12%, 81.27%, and 82.47% using MOT17, open image bus truck, MOT15, 16, and 20 datasets than existing methods like YOLOTracker and RetinaMOT. In the future, an improved DL method will be considered to enhance the model performance.

Notation Description

Symbol	Description
$s_{col}(r_i, r_j)$	Color similarity
$s_{tex}(r_i, r_j)$	Texture similarity
$T_i = \{t_i^1, \ldots, t_i^n\}$	texture histograms
$s_{fill}(r_i, r_j)$	Fill similarity
BB_{ij}	Tight bounding box over r_i and r_j
r_i and r_j	Region i, j
c_i^k and c_j^k	Color histogram for regions r_i and r_j for k^{th} color channel
n	Total number of colour channels
t_i^k, t_j^k	Texture histogram
$\varepsilon \in (0,1)$	Constant number
$\tilde{Z}$	Convolution process output
W	Weights
$f(.)$	Activation function
a and b	Bias
U	Uniform distribution
n	Previous layer size
$\hat{F}$	Convolutional process output
y_k	SoftMax output in class k
Z^*	FC layer output
C	Number of classes (labels)
c_i	Detection box
$\widehat{B_t}$	Predicted location

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

The paper conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization, have been done by 1st author. The supervision and project administration, have been done by 2nd author.

References

[1] X. Hu, Y. Jeon, and J. Gwak, "FFTransMOT: Feature-Fused Transformer for Enhanced Multi-Object Tracking", *IEEE Access*, Vol. 11, pp. 130060-130071, 2023.

[2] V.D. Stanojevic, and B.T. Todorovic, "BoostTrack: boosting the similarity measure and detection confidence for improved multiple object tracking", *Machine Vision and Applications*, Vol. 35, p. 53, 2024.

[3] D.C. Bui, H.A. Hoang, and M. Yoo, "AFMtrack: Attention-Based Feature Matching for Multiple Object Tracking", *IEEE Access*, Vol. 12, pp. 82897-82910, 2024.

[4] R. Mostafa, H. Baraka, and A. Bayoumi, "LMOT: Efficient Light-Weight Detection and Tracking in Crowds", *IEEE Access*, Vol. 10, pp. 83085-83095, 2022.

[5] Y. Wu, H. Sheng, Y. Zhang, S. Wang, Z. Xiong, and W. Ke, "Hybrid Motion Model for Multiple Object Tracking in Mobile Devices", *IEEE Internet of Things Journal*, Vol. 10, No. 6, pp. 4735-4748, 2023.

[6] C. -J. Liu and T. -N. Lin, "DET: Depth-Enhanced Tracker to Mitigate Severe Occlusion and Homogeneous Appearance Problems for Indoor Multiple-Object Tracking", *IEEE Access*, Vol. 10, pp. 8287-8304, 2022.

[7] L. Ye, W. Li, L. Zheng, and Y. Zeng, "Lightweight and Deep Appearance Embedding for Multiple Object Tracking", *IET Computer Vision*, Vol. 16, No. 6, pp. 489-503, 2022.

[8] A. Boragule, H. Jang, N. Ha, and M. Jeon, "Pixel-guided association for multi-object tracking", *Sensors*, Vol. 22, No. 22, p. 8922, 2022.

[9] S. Bilakeri and A.K. Karunakar, "Multi-object tracking by multi-feature fusion to associate all detected boxes", *Cogent Engineering*, Vol. 9, No. 1, p. 2151553, 2022.

[10] X. Feng, X. Jiao, S. Wang, Z. Zhang, and Y. Liu, "SCGTracker: object feature embedding enhancement based on graph attention networks for multi-object tracking", *Complex & Intelligent Systems*, Vol. 10, No. 4, pp. 5513-5527, 2024.

[11] Z. Ni, C. Zhai, Y. Li, and Y. Yang, "A Multi-Object Tracking Method With Adaptive Dual Decoder and Better Motion Affinity", *IEEE Access*, Vol. 12, pp. 20221-20231, 2024.

[12] W. Yang, Y. Jiang, S. Wen, and Y. Fan, "Online multiple object tracking with enhanced Re-identification", *IET Computer Vision*, Vol. 17, No. 6, pp. 676-686, 2023.

[13] S.A. Qureshi, L. Hussain, Q.u.a. Chaudhary, S.R. Abbas, R.J. Khan, A. Ali, and A. Al-Fuqaha, "Kalman filtering and bipartite matching based super-chained tracker model for online multi object tracking in video sequences", *Applied Sciences*, Vol. 12, No. 19, p. 9538, 2022.

[14] X. Xiao, and X. Feng, "Multi-object pedestrian tracking using improved YOLOv8 and OC-SORT", *Sensors*, Vol. 23, No. 20, p. 8439, 2023.
[15] Y. Li, L. Wu, Y. Chen, X. Wang, G. Yin, and Z. Wang, "Motion estimation and multi-stage association for tracking-by-detection", *Complex & Intelligent Systems*, Vol. 10, No. 2, pp. 2445-2458, 2024.
[16] S. Chan, Y. Jia, X. Zhou, C. Bai, S. Chen, and X. Zhang, "Online multiple object tracking using joint detection and embedding network", *Pattern Recognition*, Vol. 130, p. 108793, 2022.
[17] J. Cao, J. Zhang, B. Li, L. Gao, and J. Zhang, "RetinaMOT: Rethinking anchor-free YOLOv5 for online multiple object tracking", *Complex & Intelligent Systems*, Vol. 9, No. 5, pp. 5115-5133, 2023.
[18] R. Alagarsamy, and D. Muneeswaran, "Multi-Object Detection and Tracking Using Reptile Search Optimization Algorithm with Deep Learning", *Symmetry*, Vol. 15, No. 6, p. 1194, 2023.
[19] X. Gao, Z. Wang, X. Wang, S. Zhang, S. Zhuang, and H. Wang, "DetTrack: An Algorithm for Multiple Object Tracking by Improving Occlusion Object Detection", *Electronics*, Vol. 13, No. 1, p. 91, 2023.
[20] Y. Wang and V.Y. Mariano, "A Multi Object Tracking Framework Based on YOLOv8s and Bytetrack Algorithm", *IEEE Access*, vol. 12, pp. 120711-120719, 2024.
[21] Dataset link for MOT17: https://motchallenge.net/data/MOT17/
[22] Dataset link for open image bus truck: https://www.kaggle.com/datasets/sixhky/open-images-bus-trucks
[23] Dataset link for MOT15: https://motchallenge.net/data/MOT15/
[24] Dataset link for MOT16: https://motchallenge.net/data/MOT16/
[25] Dataset link for MOT20: https://motchallenge.net/data/MOT20/
[26] A. Bar, X. Wang, V. Kantorov, C.J. Reed, R. Herzig, G. Chechik, A. Rohrbach, T. Darrell, and A. Globerson, "Detreg: Unsupervised pretraining with region priors for object detection", In: *Proc. of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14605-14615, 2022.
[27] G.S. Nijaguna, J.A. Babu, B.D. Parameshachari, R.P. de Prado, and J. Frnda, "Quantum Fruit Fly algorithm and ResNet50-VGG16 for medical diagnosis", *Applied Soft Computing*, Vol. 136, p. 110055, 2023.
[28] H.Q.T. Ngo, "Design of automated system for online inspection using the convolutional neural network (CNN) technique in the image processing approach", *Results in Engineering*, Vol. 19, p. 101346, 2023.