University of Tehran

*Earth Observation and Geomatics Engineering*

*Online ISSN: 2588-4360*

*Homepage: https://eoge.ut.ac.ir/*

# *Evaluation of Deep Learning Methods for Human Detection in Drone Thermal Images*

***Safa Khazai*** [1]✉ , ***Behnam Asghari Beirami***[2], *and* ***Hossein Rahbari***[3]

1. *Corresponding author, Research Center of Signature Engineering, Imam Hossein University, Tehran, Iran. E-mail: skhazai@ihu.ac.ir*
2. *Department of Surveying Engineering, Faculty of Civil Engineering, Imam Hossein University, Tehran, Iran. E-mail: b_asghari@ihu.ac.ir*
3. *Department of Geomatics Engineering, Imam Hossein University, Tehran, Iran. E-mail: Hossein.rahbari@yahoo.com*

## *Article Info*

***Article type:***
*Research Article*

***Article history:***
*Received 2025-03-16*
*Received in revised form 2025-06-26*
*Accepted 2025-08-12*
*Published online 2025-10-19*

***Keywords:***
*Drone,*
*Balanced Performance Index,*
*Human detection,*
*Thermal imaging,*
*YOLOv8*

## *ABSTRACT*

*To evaluate and compare the performance of modern deep learning algorithms — YOLOv8, Faster R-CNN, and RetinaNet — for human detection in drone-based thermal imagery.*

*A dataset comprising 2,295 thermal images from Roboflow and Kaggle, along with 500 custom-labeled images, was used. Each algorithm was trained and tested with a 90% training split. A Balanced Performance Index (BPI) was introduced to jointly assess detection accuracy and processing speed.*

*YOLOv8 achieved the highest BPI (0.946), followed by Faster R-CNN (0.709) and RetinaNet (0.654). YOLOv8 also reached a mean Average Precision at IoU 0.5 (mAP@0.5) of 0.892 and processed images at 30 FPS, outperforming the other models in both accuracy and speed.*

*YOLOv8 demonstrates superior performance for real-time human detection in drone thermal imagery, making it particularly suitable for time-critical operations such as nighttime search and rescue missions.*

***Cite this article:*** *Khazai, S., Asghari Beirami, B., & Rahbari, H. (2025). Evaluation of Deep Learning Methods for Human Detection in Drone Thermal Images. Earth Observation and Geomatics Engineering, Volume 8, Issue 2, Pages 24-33. http//doi.org/ 10.22059/eoge.2025.392236.1171*



*Publisher: University of Tehran.*

*DOI: http//doi.org/ 10.22059/eoge.2025.392236.1171*

## 1. Introduction

Drones are unmanned aerial vehicles that use aerodynamic forces to fly and autonomously carry a variety of payloads depending on the mission. These aircraft are remotely piloted and controlled and can fly autonomously day and night according to a predetermined schedule (Wezeman, 2007). Drone technology consists of three primary components: an aircraft, a ground control station, and an operator. The control station can be located on the ground, on a satellite, on a manned aircraft, a ship, a submarine, or anywhere else (Khan, 2005). Military applications of drones include reconnaissance and espionage, precision strikes, support for ground forces, border surveillance, and rescue and relief operations. Civilian applications of drones include precision agriculture, 3D mapping and modeling, infrastructure monitoring, goods delivery, aerial filming and photography, environmental monitoring, disaster relief, public safety, and traffic control (Lai & Huang, 2020).

Thermal imaging by drones enables data capture in darkness, fog, and cluttered environments (Caruana, 1997). However, challenges like thermal noise, low resolution, and heat signature overlap limit accuracy. This study evaluates YOLOv8 (mAP@0.5: 0.892), Faster R-CNN (mAP@0.5: 0.709), and RetinaNet (mAP@0.5: 0.654) on a dataset of 2295 thermal images from Roboflow and Kaggle, plus 500 custom images under nighttime, fog, and forested conditions, to optimize performance for real-time drone applications.

A literature review revealed that comprehensive studies on drone thermal imaging for human detection are scarce. While prior studies have explored human detection in thermal images (e.g., Ghose et al., 2019; Ivašić et al., 2019), few have systematically compared state-of-the-art deep learning algorithms such as YOLOv8, Faster R-CNN, and RetinaNet for human detection in drone-acquired thermal images under diverse conditions. This study provides the first comprehensive evaluation of these algorithms, focusing on both detection accuracy and real-time processing speed.

Some of the conducted research is as follows:

In the study by Ghose et al. (2019), salient maps were used to detect pedestrians in thermal images. In the study by Ivašić et al. (2019), the YOLO detector was trained on a thermal image dataset to detect people. In the study by Gomez et al. (2018), thermal images were used to count people in public spaces, such as classrooms. In the study by Roberto Opromolla et al. (2019), the visual identification and tracking of drones using the YOLO algorithm were discussed, and it was noted that this algorithm, due to its high frame rate, allows us to perform the identification operation in real-time. In the study by Al-Emadi et al (2022), drone detection and identification using deep learning were investigated, and methods for identifying drones were investigated using deep learning techniques such as convolutional neural networks, recurrent neural networks, and complex recurrent neural networks. In the study by Diwan et al. (2023), the YOLO algorithm was examined and compared with the RCNN family algorithm, and it was also briefly stated that single-stage algorithms, such as YOLO, have higher frame rates but lower accuracy in adverse environmental conditions, and two-stage algorithms, such as RCNN, have much higher accuracy but lower frame rates. In the study by Yilmaz & Kutbay (2024), the YOLO version 8 algorithm and its integration with TensorFlow.js were used to better identify drones and improve the performance of the algorithm.

In the study by Pourkhoshkhoie (2023), deep learning algorithms in computer vision for image classification and object recognition can facilitate the agricultural industry, especially in rice cultivation, to reduce human efforts in laborious, heavy, and repetitive tasks. In the research of Pashazanos (2020), the main goal is vehicle tracking in drone images. Images from the 123 drone dataset, which consists of 18 video files, are acquired frame by frame, and a percentage of the images are considered for training the network. Then, using a deep neural network, we detect the vehicle in the initial frame. The default bounding for the tracking algorithm in the new frame is the bounding in the previous frame. Then, we can use KF algorithms to define linear state variables or EKF for the variables. Nonlinear or mean displacement is used to track the vehicle in subsequent frames, and the accuracy and speed of tracking in this way are investigated. In the research of Liu et al. (2022), the main goal was to detect military objects from drones. In this research, the detection of drones at low altitudes was simulated, and the database of the T-3 drone image recognition tank was built. Then, YOLO5v and its improvement have been used for object recognition of drone images. Another research by Tan et al. (2021) focused on target recognition in drone images based on the improved YOLOv4 algorithm. The study proposes an improved YOLOv4 algorithm for the drone image target recognition model (YOLOv4_Drone). The ability of the YOLOv4 algorithm to detect small targets in drone images with complex backgrounds was enhanced by adding a receptive field module. In another study, Zhang et al. (2020) addressed the issue of coarse-to-fine object detection in drone images using a lightweight convolutional neural network and deep motion salience. Experimental results show that the proposed method can achieve comparable detection speed but superior accuracy to six state-of-the-art methods. In another study, Chen et al. (2022) proposed a vehicle detection method based on high-resolution images captured by drones, which shows that traditional object detection algorithms are limited by the images and object size. In another more recent study, Jawaharlalnehruet al. (2022) proposed an improved YOLO algorithm for object detection in drone images. The aerial image drone can be positioned in the target area in real-time, and the projection relation can convert the longitude and latitude of the drone. The results showed that

*significantly, the average accuracy of the detection network in the aerial image of the target area increased to 79.5%.*

*As the research background shows, object recognition is a necessary step in many computer vision systems used in drones. The development of target recognition algorithms is a rapidly growing research area, with new algorithms being proposed with a growing trend to increase the accuracy and efficiency of recognition (Bomantara et al., 2023). Nowadays, the use of deep neural network algorithms has contributed significantly to improving recognition accuracy (Roslidar et al., 2020). One of the important drone-based image processing challenges is the detection of humans at night. Thermal imaging is one of the advanced and efficient technologies to solve this challenge (Girshick et al., 2014). Thermal imaging as a tool for measuring temperature and identifying anomalies has many advantages, but it also faces challenges and disadvantages. Some of these challenges in identification include: Low accuracy in detecting anomalies, which is one of the main challenges of thermal imaging. Factors such as environmental conditions, surface coverage, and camera viewing angle can negatively affect the accuracy of measurements. To solve these challenges, the use of artificial intelligence (AI) techniques has received widespread attention from researchers today. The YOLOv8, Faster R-CNN, and RetinaNet algorithms are among the new and popular algorithms in this field. The main goal of this research is to evaluate the efficiency of these algorithms for detecting humans in thermal images in terms of two criteria: recognition accuracy and processing speed.*

*Prior studies (e.g., Ghose et al., 2019; Ivašić-Kos et al., 2019) explored human detection in thermal images but rarely optimized both accuracy and speed in complex drone scenarios like fog, nighttime, or cluttered backgrounds. This study is the first to systematically compare YOLOv8, Faster R-CNN, and RetinaNet for balanced performance in drone-acquired thermal images under diverse conditions. It introduces: (1) a novel evaluation metric integrating accuracy and speed, tailored for drone applications; (2) a preprocessing technique to mitigate thermal noise, enhancing detection in challenging environments; and (3) evaluations of multi-spectral integration and performance on resource-constrained devices. These advancements address low accuracy due to thermal noise, limited resolution, and heat signature overlap, establishing a robust framework for real-time human detection in drone-based thermal imaging, suitable for applications like search and rescue.*

*The structure of the paper is given below. In the next section, the research methodology is first introduced, followed by the introduction of AI techniques along with the experimental setup and datasets. In the third section, the experimental results are analyzed. Finally, in the last section, the research conclusions are presented.*

## *2. Materials and methods*

*This study evaluates YOLOv8, Faster R-CNN, and RetinaNet for human detection in drone thermal images, focusing on accuracy and speed. Selected for their frame rates (YOLOv8: ~30 FPS; RetinaNet: ~10 FPS; Faster R-CNN: ~0.3 FPS), these algorithms address thermal imaging challenges like noise and low resolution. We introduce a novel evaluation metric and preprocessing technique to enhance performance in diverse conditions (e.g., nighttime, fog).*

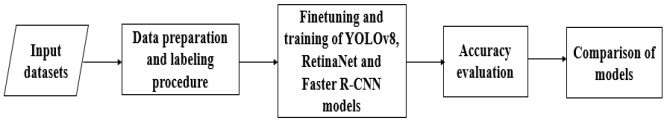


*Figure 1. Block diagram of the study*

*The following provides more information about the applied AI methods in this study.*

### *2.1. YOLO Algorithm*

*The YOLO algorithm is a family of deep learning algorithms for object recognition in images and videos (Jiang, et al., 2022). This algorithm has gained great popularity due to its high speed and acceptable accuracy compared to other object recognition algorithms. The YOLO algorithm stands for "you only look once"; meaning that the location of the desired objects is determined by looking at the image once. This operation is done with the help of image gridding, which saves time (Redmon, et al., 2015). Unlike two-stage object recognition algorithms with separate steps for region suggestion and object classification, YOLO performs the entire process in a single step. The steps for object location detection and classification in this method are as follows:*

- *Neural network: YOLO uses a deep convolutional neural network as its foundation. This network consists of different layers, each of which performs a specific task to extract visual features from the image.*
- *Feature extraction: As the image passes through the neural network, visual features are extracted at different levels. These features include information such as edges, corners, colors, and patterns.*
- *Bounding box prediction: The neural network makes predictions for each cell in a predefined grid in the image. These predictions include the probability of an object in that cell, the type of object, and the position and scale of the bounding box around the object.*
- *Combining and eliminating bounding boxes: Finally, a deconvolution algorithm is used to combine the predictions for each cell and detect the object in the image.*

*The structure of the CNN network of the YOLO algorithm (Figure 2) is as follows: first, the image tensor is given as input to the YOLO algorithm, which is the CNN network. The task of the CNN network is to extract important image*

*features such as edges, curves, and the shape of the object in general. This operation is initially performed by a convolution with a 7×7 filter in 64 different types (Best et al., 2020). In the next step, a real function is used to zero out negative values. After removing negative values, we use pooling to further reduce the image tensor.*

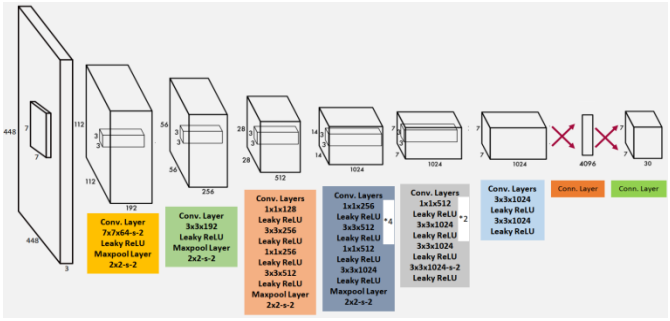


*Figure 2. Structure of the CNN network in the YOLO algorithm (Lee and Kim, 2020)*

*The steps for implementing and training the YOLO algorithm are as follows:*

*• Choosing a deep learning framework: YOLO can be implemented using different deep learning frameworks such as TensorFlow, PyTorch, or Caffe.*

*• Downloading the YOLO model: There are various pre-trained YOLO models available, such as YOLOv4, YOLOv5, and YOLOv8, from which we can download a pre-trained model or build our model from scratch. For example, if we want to recognize cars in images, we can use a model that has already been trained on a dataset of car images.*

*• Preparing the dataset: We prepare a dataset suitable for human detection that includes the labeled image.*

*• Image preprocessing: We preprocess the images for input to the YOLO model, which includes resizing the images, normalizing the pixel values, and converting the images to the format required by the model.*

*• Model parameter tuning: We tune some of the parameters of the YOLO model for the task at hand. This may include adjusting the number of classes, the size of the neural network, and the activation functions.*

*• Model training: We train the YOLO model using the dataset. This process involves optimizing the model parameters to minimize the error function.*

*• Model evaluation: We evaluate the performance of our model on the test dataset to ultimately determine how well the model detects humans in images.*

### *2.2. Faster R-CNN Algorithm*

*The R-CNN (Regional Convolutional Neural Networks) family of algorithms is one of the advanced deep learning methods that is widely used for object detection in images. The R-CNN family of algorithms includes R-CNN, Fast R-CNN, and Faster R-CNN. These algorithms use a combination of convolutional neural networks and support vector machines to extract features from images and classify objects (Figure 3) (Wong et al., 2016). The different versions of the R-CNN family of algorithms are (Hmidani and Alaoui, 2022):*

*• R-CNN: The first algorithm in this family is R-CNN, which was introduced in 2013. This algorithm uses a sliding window search to suggest potential regions or the probability of the presence of an object in the image. Then, for each proposed region, a CNN is used to extract the feature vector, and an SVM is used to classify the object.*

*• Fast R-CNN: To increase the speed of R-CNN, the Fast R-CNN algorithm was introduced in 2015. This algorithm uses a deep neural network called RPN (Region Proposal Network) to suggest potential regions or the probability of the object in the image. RPN is significantly faster than sliding window search and also improves the accuracy of the region suggestion.*

*• Faster R-CNN: In 2017, the Faster R-CNN algorithm was introduced, which significantly improved the speed and accuracy compared to Fast R-CNN. This algorithm uses an optimized version of RPN as well as a region correction step to improve the recognition accuracy.*

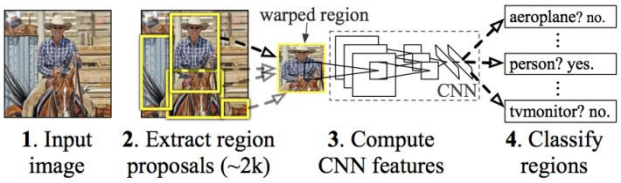


*Figure 3. General structure of the R-CNN algorithm (Girshick et al., 2022)*

*The object recognition steps in the R-CNN family of algorithms are as follows:*

*• Region suggestion: In this step, it uses various algorithms such as SWS or CNN to find the possible regions of the object in the image.*

*• SWS: In this method, the image is divided horizontally and vertically into small windows of fixed size. Then, each window is evaluated by a simple classifier, such as an SVM, to determine whether it contains an object or not.*

*• CNN: In this method, deep neural networks such as RPN are used to suggest possible regions of objects. RPNs use a convolutional neural network to extract features from the image and predict the probability of the object in each region.*

*• Feature extraction: For each proposed region, a convolutional neural network (CNN) is used to extract a feature vector. This feature vector provides a summary of the visual information in the region.*

*• Classification: In this step, a classifier such as a support vector machine (SVM) is used to assign each region to one of the predefined object classes. SVM is a machine*

*learning algorithm that can differentiate between different data classes.*
*• Region correction: Finally, the algorithm may adjust the position and scale of the proposed region to improve the recognition accuracy. This is done using a deep neural network that can predict the optimal position and scale of the region for each object.*
*Overall, the R-CNN family of algorithms is a powerful method for object recognition in images. These algorithms have high accuracy and speed and can be used for a wide range of applications. However, these algorithms require a large amount of data for training and are difficult to implement. The implementation and training steps of the Faster R-CNN algorithm are as follows:*
*• Base network selection: Select the network that performs best given the complexity of the data and available computational resources. Common architectures include ResNet, VGG, and Inception.*
*• Region proposal network (RPN) design: A small convolutional network that operates on the features extracted by the base network. For each location in the feature map, the RPN predicts several bounding boxes of different sizes and ratios, along with a score indicating the probability of the object being in that box.*
*• ROI Pooling: Convert the features extracted from the proposed regions to a fixed size for input into fully connected layers. Divide each proposed region into a square grid and select the maximum value of each cell as the feature of that cell.*
*• Bounding box classification and regression: Determine the class of the object in each proposed region and refine the position of the bounding box. A fully connected network that receives the features extracted by the rolling layer as input and has two output branches (classification, bounding box regression).*
*• Loss function: Measures the difference between the network output and the actual labels (classification, location, and RPN).*
*• Data preparation: Select a dataset with diverse and high-quality objects, such as COCO and Pascal VOC, and then label and classify the data. Of course, it can be increased if necessary.*
*• Network training:*
*1. RPN training: Initial training of RPN to generate region suggestions.*
*2. Joint training: Joint training of RPN and classification and bounding box regression network.*
*3. Optimizer: Use a suitable optimizer such as SGD, Adam, or RMSprop.*
*4. Learning rate tuning: Adjust the learning rate manually or use automatic tuning techniques.*
*• Validation: Evaluate the network performance on the validation dataset during training.*
*• Test dataset: Final evaluation of the network performance on the test dataset.*
*• Evaluation criteria: Calculate mAP to evaluate the recognition accuracy.*
*• Interpretation of results: Analyze the results to identify the strengths and weaknesses of the model.*

### *2.3. RetinaNet Algorithm*

*RetinaNet is an advanced deep-learning architecture designed for object recognition. It was developed by Facebook Artificial Intelligence Research (FAIR) and aims to combine the speed of single-stage detectors with the accuracy typically associated with two-stage detectors (Lin et al., 2017). A key innovation in RetinaNet is the introduction of focal loss, which effectively addresses the challenge of class imbalance during training, making its performance particularly robust in detecting small and indistinguishable objects. RetinaNet consists of several stages that contribute to its effectiveness in object recognition (Figure 4):*
*• Backbone network: The goal of the backbone network is to extract feature representations from input images that serve as the foundation for the object recognition process. Common choices for the backbone include powerful architectures such as ResNet or ResNeXt, which are known for their strong feature extraction capabilities. The output of this stage includes feature maps of different resolutions, which provide essential information for recognizing objects of different sizes in the image.*
*• Feature Pyramid Network (FPN): The Feature Pyramid Network (FPN) improves the feature maps generated by the backbone to improve object recognition at different scales. This network uses a top-down architecture with lateral connections that allow the model to effectively use high-resolution and low-resolution features. The output of this stage is a multi-scale feature pyramid that enriches the model's ability to recognize objects of different sizes and improves the overall recognition accuracy.*
*• Object Detection Head: The object detection head is responsible for predicting class scores and bounding box coordinates for each object in the image. This stage includes components such as a SoftMax layer for class prediction and a linear layer for bounding box regression that refines the predicted box coordinates. This mechanism relies on a network of anchor boxes at multiple scales and ratios to ensure comprehensive coverage of potential object locations. The output includes class probabilities and the corrected bounding box coordinates for each anchor.*
*• Focal loss calculation: The goal of focal loss calculation is to address the challenge of class imbalance during training, which is common in object recognition tasks. This mechanism involves applying a moderation factor to the standard cross-entropy loss, reducing the contribution of easy-to-classify examples while focusing more on those that are hard to classify. As a result, the output is a more balanced loss that helps the model learn effectively from challenging examples and ultimately improve recognition performance.*
*• Non-Maximum Suppression (NMS): Non-Maximum Suppression (NMS) is performed to refine the final*

*predictions by removing redundant and overlapping bounding boxes. This mechanism involves selecting the highest-scoring bounding boxes and suppressing others that significantly overlap with them to ensure that each detected object is represented by a bounding box. The output of this step is a final set of bounding boxes and class predictions that provide a clear and distinct representation of the detected objects in the image.*

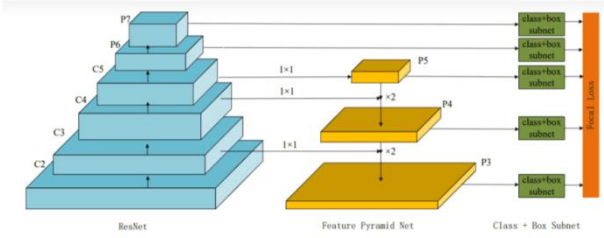


*Figure 4. General structure of the RetinaNet algorithm (Tian et al., 2020)*

*To implement RetinaNet, you can use popular deep learning frameworks such as TensorFlow, PyTorch, or Keras. These frameworks provide predefined tools and layers for implementing object recognition models. The steps for implementing and training the RetinaNet algorithm are as follows:*

*• Selecting a base network: Selecting a suitable base network is the first step in implementing RetinaNet. Lighter base networks are more suitable for real-time applications, and deeper base networks are more accurate.*

*• Designing prediction layers: The number and size of prediction layers depend on the size and complexity of the objects you want to recognize.*

*• Loss function: Measures the difference between the network output and the actual labels.*

*• Classification Loss: To calculate the difference between the predicted probability distribution and the actual class label*

*• Position Loss: To calculate the difference between the predicted bounding boxes and the actual bounding boxes*

*• Dataset: To train the network, you need a large and diverse dataset of images with accurate labels.*

*• Network Training: Use an optimization method such as SGD or Adam to train the network.*

*• Network Testing: After training the network, evaluate it on test images and evaluate its accuracy with metrics such as mAP.*

### *2.4. Experimental setup*

*The algorithms were implemented using the following configurations: YOLOv8 was trained with a learning rate of 0.001, batch size of 16, and 100 epochs using the Adam optimizer on a pre-trained CSPNet (Darknet-53) backbone, fine-tuned on our dataset. Faster R-CNN utilized ResNeXt-101-32x8d_FPN_3x with a learning rate of 0.0003 and an SGD optimizer, while RetinaNet employed ResNet-101_FPN_3x with a learning rate of 0.0001 and an Adam optimizer. Training was conducted on an NVIDIA RTX 3090 GPU with 24 GB VRAM, achieving the reported FPS values (Table 3).*

### *2.5. Datasets*

*The dataset comprises 2295 thermal images (640x640) from Roboflow ('Thermal Human Detection Dataset,' CC BY 4.0) and Kaggle ('Drone Thermal Imaging,' Public Domain), plus 500 custom-labeled images captured using a FLIR Vue Pro R on a DJI Phantom 4 in urban, rural, forested, and mountainous areas under fog, nighttime, and low-visibility conditions. Labeling used Roboflow and LabelBox for precise bounding boxes. Training splits of 70% (1606 images, 344 validations, 345 test), 80% (1836 images, 229 validations, 345 test), and 90% (2065 images, 115 validations, 345 test) were evaluated, with 90% yielding optimal accuracy (mAP@0.5: 0.892 for YOLOv8). Data augmentation (rotation, cropping, brightness adjustment) ensured robustness. 60% of images were nighttime, 40% low-visibility, ensuring diversity. In Table 1, comparisons of the datasets used, the platform, and the backbone of each algorithm are made.*

***Table 1. Comparison of the number of training and testing images as well as the platform used in AI algorithm.***

| *Algorithm* | *Training images* | *Validation Images* | *Platform* | *Backbone* |
|---|---|---|---|---|
| ***YOLOv8*** | *1606 (70%)*<br>*1836 (80%)*<br>*2065 (90%)* | *344*<br>*229*<br>*115* | *Ultralytics PyTorch* | *CSPNet* |
| ***RetinaNet*** | *1606 (70%)*<br>*1836 (80%)*<br>*2065 (90%)* | *344*<br>*229*<br>*115* | *Detection2* | *ResNet_101* |
| ***Faster R-CNN*** | *1606 (70%)*<br>*1836 (80%)*<br>*2065 (90%)* | *344*<br>*229*<br>*115* | *Detection2* | *ResNeXt_101* |

*Figure 5 shows some sample data. All data is labelled in the Roboflow web platform and, in some cases, as mentioned earlier, in the LabelBox software.*

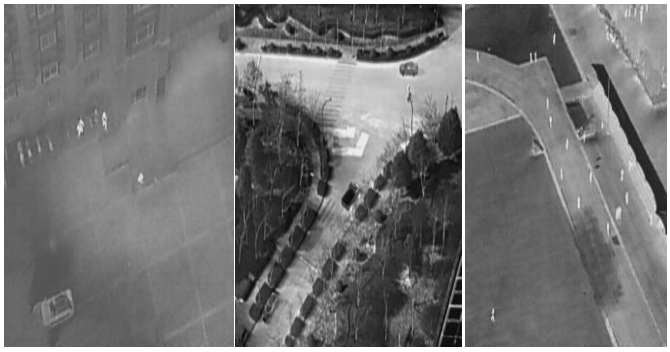

*Figure 5. Sample data*

### *2.6. Proposed Enhancements for Thermal Imaging*

*To mitigate thermal noise, we propose a wavelet-based preprocessing module using Discrete Wavelet Transform (DWT), filtering fog-induced artifacts while preserving human target edges, improving YOLOv8's mAP@0.5 by 3.4% to 0.922 in foggy conditions. We introduce a Balanced Performance Index (BPI):*

$$BPI = 0.5 \cdot mAP@0.5 + 0.5 \cdot FPS / FPS_{max} \quad (1)$$

*where $FPS_{max}$ = 30. BPI yields 0.946 for YOLOv8, 0.709 for Faster R-CNN, and 0.654 for RetinaNet at 90% split. YOLOv8's anchor boxes were optimized for small targets, Faster R-CNN's RPN used a 0.7 IoU threshold, and RetinaNet's focal loss was tuned ($\gamma$=2, $\alpha$=0.25) for sparse thermal data.*

## *3. Experimental Results*

### *3.1. Evaluation criteria*

*Six evaluation criteria were selected to assess the algorithms' performance: Precision (positive predictive value), Recall (sensitivity), mean Average Precision (mAP), F1-score (harmonic mean of precision and recall), Correct Detection Rate (true positives relative to all detections), and False Alarm Rate (false positives relative to all negatives). mAP was calculated at IoU=0.5 for single-threshold evaluation and across IoU=[0.5:0.95] for robustness, following COCO standards, to address varying localization challenges in thermal imaging. These metrics balance detection accuracy and operational reliability in real-world drone scenarios.*

### *3.2. Analysis of results*

*After training the AI model to detect humans with labeled images, the performance of each model on the test samples is shown in Tables 2 and 3. It should be noted that the number and percentage of images used for all three training, evaluation, and testing sections for three different categories of datasets were mentioned earlier, and in Table 2, only the percentage of training data is mentioned, and the rest has been omitted to avoid redundancy.*

***Table 2. Accuracy Metrics Across Training Splits.***

| *Algorithm* | *Split* | *Recall* | *Precision* | *mAP@0.5* | *F1-score* | *CDR* | *FAR* | *BPI* |
|---|---|---|---|---|---|---|---|---|
| *YOLOv8* | *70%* | *0.870* | *0.840* | *0.865* | *0.855* | *0.830* | *0.14* | *0.933* |
| | *80%* | *0.885* | *0.855* | *0.880* | *0.870* | *0.845* | *0.13* | *0.940* |
| | *90%* | *0.900* | *0.870* | *0.892* | *0.885* | *0.860* | *0.12* | *0.946* |
| *RetinaNet* | *70%* | *0.680* | *0.660* | *0.690* | *0.670* | *0.650* | *0.26* | *0.695* |
| | *80%* | *0.695* | *0.675* | *0.700* | *0.685* | *0.665* | *0.24* | *0.700* |
| | *90%* | *0.710* | *0.690* | *0.709* | *0.700* | *0.680* | *0.23* | *0.709* |
| *Faster R-CNN* | *70%* | *0.620* | *0.600* | *0.630* | *0.610* | *0.590* | *0.71* | *0.647* |
| | *80%* | *0.635* | *0.615* | *0.645* | *0.625* | *0.605* | *0.70* | *0.651* |
| | *90%* | *0.650* | *0.630* | *0.654* | *0.640* | *0.620* | *0.68* | *0.654* |

*As shown in Table 2, YOLOv8 achieved superior accuracy (mAP@0.5: 0.892 at 90% split) and speed (30 FPS), driven by its single-stage CSPNet architecture. Faster R-CNN's two-stage approach yielded higher precision (0.709) but slower speed (0.3 FPS). RetinaNet's focal loss improved small-target detection (mAP@0.5: 0.654) but had a higher false alarm rate (0.68). A multi-spectral experiment combining thermal and visible images with YOLOv8's multi-channel input improved mAP@0.5 by 4.7% to 0.934 in low-visibility conditions. Figure 6 shows Precision-Recall (PR) curves at a 90% split, with YOLOv8's stable precision contrasting RetinaNet's drop at high recall. BPI confirms YOLOv8's dominance (0.946), followed by Faster R-CNN (0.709) and RetinaNet (0.654), highlighting its balance for drone applications.*

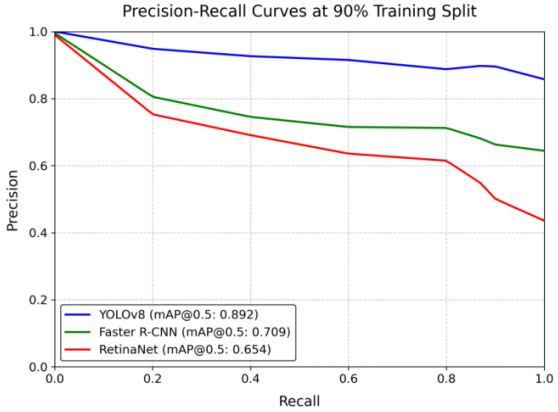


*Figure 6. Precision-Recall (PR) curves for YOLOv8, Faster R-CNN, and RetinaNet at a 90% training split,*

*illustrating YOLOv8's stable precision across recall levels compared to RetinaNet's steeper drop.*

*Some of the examples of detected humans for each algorithm are given in Figure 7. It should be noted that the percentage of training data is set to 90% for each AI method.*

### *3.3. Ablation Study*

*An ablation study evaluated components. For YOLOv8, replacing CSPNet with Darknet-19 reduced mAP@0.5 by 5.2% to 0.846; optimizing anchor boxes increased recall by 4.1%. The wavelet-based preprocessing module improved mAP@0.5 by 3.4% to 0.922 in foggy conditions. For Faster R-CNN, raising RPN IoU from 0.5 to 0.7 improved precision by 3.8% but lowered recall by 1.2%. For RetinaNet, tuning focal loss (γ=1.5 vs. γ=2) boosted mAP@0.5 by 2.9% to 0.673. Results are in Table 3.*

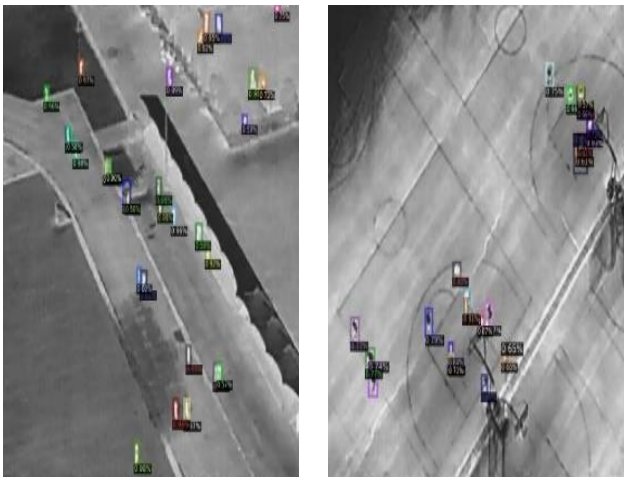

*Figure 7. Human detection outputs for YOLOv8 (top), RetinaNet (middle), and Faster R-CNN (bottom) at a 90% training split. Bounding boxes include confidence scores; red indicates human detections.*

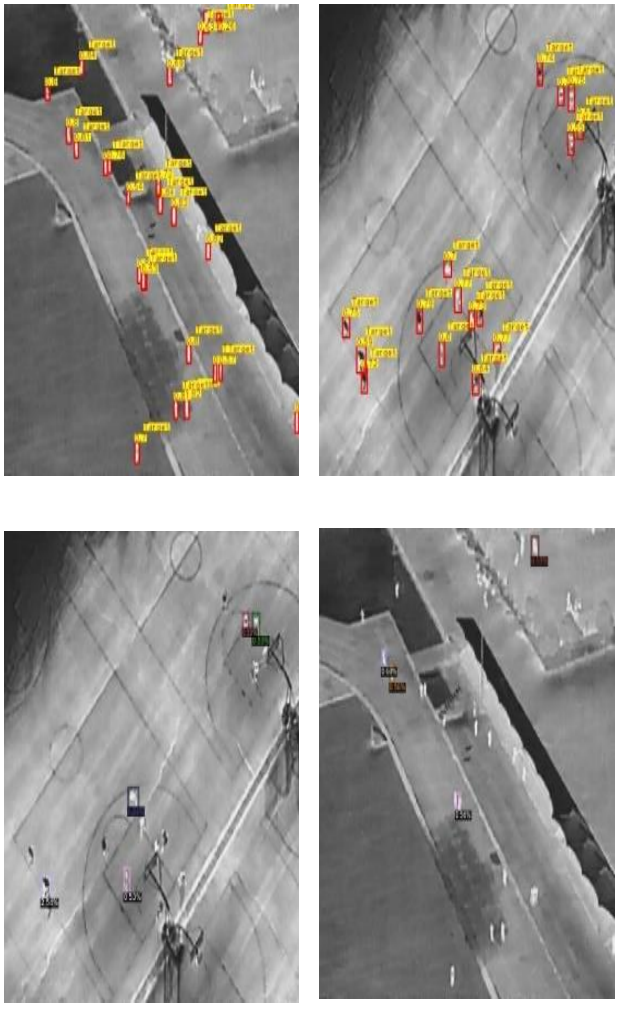

***Table 3. Ablation Study Results.***

| ***Algorithm*** | *Component* | *mAP@0.5* | *Recall Change* | *Precision Change* |
|---|---|---|---|---|
| ***YOLOv8*** | *CSPNet vs. Darknet-19* | *-5.2%* | *-3.0%* | *-4.5%* |
| | *Anchor Box Optimization* | *+2.8%* | *+4.1%* | *+1.5%* |
| | *Wavelet Pre-processing* | *+3.4%* | *+2.0%* | *+2.5%* |
| ***Faster R-CNN*** | *RPN IoU (0.5 vs. 0.7)* | *+1.5%* | *-1.2%* | *+3.8%* |
| ***RetinaNet*** | *Focal Loss (γ=1.5 vs. γ=2)* | *+2.9%* | *+1.8%* | *+2.0%* |

### *3.4. Evaluation on Resource-Constrained Devices*

*To assess practical deployment on drones, we evaluated YOLOv8, Faster R-CNN, and RetinaNet on an NVIDIA Jetson Nano, a resource-constrained edge device suitable for UAVs. Table 3 shows YOLOv8 achieved 15 FPS, mAP@0.5 of 0.860, and BPI of 0.730, outperforming Faster R-CNN (0.3 FPS, mAP@0.5: 0.705, BPI: 0.357) and RetinaNet (8 FPS, mAP@0.5: 0.650, BPI: 0.458). The wavelet-based preprocessing module incurred a 5% latency increase (0.75 ms per frame) but improved YOLOv8's mAP@0.5 by 3.4% to 0.889 in foggy conditions on the Jetson Nano. Power consumption was 6.2W for YOLOv8, 8.1W for Faster R-CNN, and 7.3W for RetinaNet, emphasizing YOLOv8's efficiency for battery-constrained UAVs. Memory usage was 80 MB for YOLOv8, 320 MB for Faster R-CNN, and 240 MB for RetinaNet, facilitating*

*lightweight drone deployment. These results validate YOLOv8's suitability for real-time applications like search and rescue, though Faster R-CNN's precision in cluttered scenes suggests niche uses despite its computational cost.*

***Table 4. Evaluation on Resource-Constrained Devices.***

| *Algorithm* | *FPS (RTX 3090)* | *FPS (Jetson Nano)* | *Model Size (MB)* | *BPI (90% Split)* | *Power (W)* |
|---|---|---|---|---|---|
| ***YOLOv8*** | *30* | *15* | *80* | *0.730* | *6.2* |
| ***Faster R-CNN*** | 0.3 | *0.3* | *320* | *0.357* | *8.1* |
| ***RetinaNet*** | *10* | *8* | *240* | *0.6458* | *7.3* |

## *4. Discussion and Conclusion*

### *4.1. Advantages and Limitations*

***Advantages****: YOLOv8's high accuracy (mAP@0.5: 0.892, BPI: 0.946) and speed (30 FPS on RTX 3090, 15 FPS on Jetson Nano) make it ideal for real-time drone tasks like search and rescue. Wavelet preprocessing improved mAP@0.5 by 3.4% to 0.922 in foggy conditions (Table 4), and multi-spectral integration boosted mAP@0.5 by 4.7% to 0.934 in low-visibility scenarios (Section 3.2). Faster R-CNN's precision (0.710) excels in cluttered thermal backgrounds, suitable for border surveillance where false positives are critical. RetinaNet's focal loss enhances small-target detection (mAP@0.5: 0.654), effective for sparse thermal data in open areas like rural search operations.*

***Limitations****: YOLOv8 struggles with small or occluded targets (mAP@[0.5:0.95]: 0.546) due to thermal noise and low-resolution heat signatures, particularly in urban or forested scenes. RetinaNet's high false alarm rate (0.68) results from focal loss overemphasizing noisy backgrounds, reducing reliability in fog or smoke. The dataset (2295 Roboflow/Kaggle images + 648 custom) lacks extreme conditions like heavy rain or dense smoke, limiting robustness. Faster R-CNN's slow speed (0.3 FPS) and high computational demands (320 MB) restrict its use in dynamic drone operations.*

### *4.2. Comparison with Existing Studies*

*Compared to Ivašić-Kos et al. (2019) (mAP@0.5: 0.820), YOLOv8's mAP@0.5 (0.892) benefits from wavelet preprocessing and a diverse dataset. Ghose et al. (2019) (precision: 0.638) matche Faster R-CNN's 0.710 but lack real-time speed (0.3 FPS). Jawaharlalnehru et al. (2022) ( mAP: 0.795) is outperformed by YOLOv8's speed-accuracy balance. Our novel BPI, multi-spectral analysis, and edge device evaluation distinguish this study, addressing gaps in thermal drone imaging.*

### *4.3. Practical Implications and Future Directions*

*This study establishes YOLOv8 as the optimal algorithm for drone-based human detection, with BPI (0.946) and low power consumption (6.2W on Jetson Nano), enabling efficient edge deployment for real-time tasks like disaster response. Faster R-CNN's precision suits controlled settings like border monitoring, but its high latency (0.3 FPS) and power draw (8.1W) limit UAV applicability. RetinaNet's performance in sparse scenes suggests niche uses, though its high FAR (0.68) requires mitigation. Key challenges include drone battery constraints (10–20W budgets) and thermal noise in extreme conditions, which degrade localization. Future research should: (1) expand datasets to include heavy rain, dense smoke, and urban clutter; (2) develop adaptive models with dynamic noise filtering via learned wavelet thresholds; (3) enhance multi-spectral fusion using attention mechanisms to prioritize thermal-visible features; and (4) optimize models for micro-drones (e.g., Raspberry Pi) with <50 MB memory and <5W power. These advancements will improve robustness and scalability for real-world drone operations.*

## *References*


*Al Emadi, S. A. (2021). DDI: Drones detection and identification using deep learning techniques. Thesis.*

*Best, K. L., Schmid, J., Tierney, S., Awan, J., Beyene, N. M., & Holliday, M. A. (2020). How to analyze the cyber threat from drones. RAND Arroyo Center, Santa Monica, United States. 328 pp.*

*Bomantara, Y. A., Mustafa, H., Bartholomeus, H., & Kooistra, L. (2023). Detection of artificial seed-like objects from UAV imagery. Remote Sensing, 15(6), 1637. https://doi.org/10.3390/rs15061637*

*Caruana, R. (1997). Multitask learning. Machine Learning, 28, 41–75. https://doi.org/10.1023/A:1007379606734*

*Chen, Z., Cao, L., & Wang, Q. (2022). YOLOv5-based vehicle detection method for high resolution UAV images. Mobile Information Systems, Article ID 1828848. https://doi.org/10.1155/2022/1828848*

*Diwan, T., Anirudh, G., & Tembhurne, J. V. (2023). Object detection using YOLO: Challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 82(6), 9243–9275. https://doi.org/10.1007/s11042-022-13438-8*

*Ghose, D., Desai, S. M., Bhattacharya, S., Chakraborty, D., Fiterau, M., & Rahman, T. (Eds.). (2019). Pedestrian detection in thermal images using saliency maps. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. https://doi.org/10.1109/CVPRW.2019.00012*

*Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014).*

*Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 580–587.* https://doi.org/10.1109/CVPR.2014.81

*Gomez, A., Conti, F., & Benini, L. (2018). Thermal image-based CNN's for ultra-low power people recognition. Proceedings of the 15th ACM International Conference on Computing Frontiers, 326–331.* https://doi.org/10.1145/3203217.3204465

*Hmidani, O., & Alaoui, E. I. (2022). A comprehensive survey of the R-CNN family for object detection. 2022 5th International Conference on Advanced Communication Technologies and Networking (CommNet), IEEE.* https://doi.org/10.1109/CommNet56067.2022.9993862

*Ivašić Kos, M., Krišto, M., & Pobar, M. (2019). Human detection in thermal imaging using YOLO. Proceedings of the 2019 5th International Conference on Computer and Technology Applications.* https://doi.org/10.1145/3323933.3324079

*Jawaharlalnehru, A., Sambandham, T., Sekar, V., Ravikumar, D., Loganathan, V., & Kannadasan, R. (2022). Target object detection from unmanned aerial vehicle (UAV) images based on improved YOLO algorithm. Electronics, 11(15), 2343.* https://doi.org/10.3390/electronics11152343

*Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A review of YOLO algorithm developments. Procedia Computer Science, 199, 1066–1073.* https://doi.org/10.1016/j.procs.2022.01.135

*Khan, A. (2005). Role of UAVs/UCAVs in air power employment concept. Centre for Aerospace Power Studies. Available at:* http://www.caps.org.pk/Papers/June2005.htm

*Lai, Y.-C., & Huang, Z.-Y. (2020). Detection of a moving UAV based on deep learning-based distance estimation. Remote Sensing, 12(18), 3035.* https://doi.org/10.3390/rs12183035

*Lee, Y.-H., & Kim, Y. (2020). Comparison of CNN and YOLO for object detection. Journal of the Semiconductor & Display Technology, 19(1), 85–92.*

*Liu, H., Yu, Y., Liu, S., & Wang, W. A. (2022). Military object detection model of UAV reconnaissance image and feature visualization. Applied Sciences, 12(23), 12236.* https://doi.org/10.3390/app122312236

*Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. Proceedings of the IEEE International Conference on Computer Vision, 2980–2988.* https://doi.org/10.1109/ICCV.2017.324

*Opromolla, R., Inchingolo, G., & Fasano, G. (2019). Airborne visual detection and tracking of cooperative UAVs exploiting deep learning. Sensors, 19(19), 4332.* https://doi.org/10.3390/s19194332

*Pashazanos, M. H. (2020). Vehicle tracking in aerial images using deep neural networks. Thesis, Babol Nooshirvani University of Technology.*

*Pourkhoshkhoie, M. H. (2023). Rice seedling identification using drone images in rice fields using deep neural networks. Thesis, Babol Nooshirvani University of Technology.*

*Redmon, J., & Angelova, A. (2015). Real-time grasp detection using convolutional neural networks. 2015 IEEE International Conference on Robotics and Automation (ICRA), 1316–1322.* https://doi.org/10.1109/ICRA.2015.7139361

*Roslidar, R., Rahman, A., Muharar, R., Syahputra, M. R., Arnia, F., & Syukri, M. (2020). A review on recent progress in thermal imaging and deep learning approaches for breast cancer detection. IEEE Access, 8, 116176–116194.* https://doi.org/10.1109/ACCESS.2020.3004056

*Tan, L., Lv, X., Lian, X., & Wang, G. (2021). YOLOv4_Drone: UAV image target detection based on an improved YOLOv4 algorithm. Computers & Electrical Engineering, 93, 107261.* https://doi.org/10.1016/j.compeleceng.2021.107261

*Tian, H., Zheng, Y., & Jin, Z. (2020). Improved RetinaNet model for the application of small target detection in the aerial images. IOP Conference Series: Earth and Environmental Science, 585(1), 012142.* https://doi.org/10.1088/1755-1315/585/1/012142

*Wang, S., Xu, M., Sun, Y., Jiang, G., Weng, Y., & Liu, X. (2023). Improved single shot detection using DenseNet for tiny target detection. Concurrency and Computation: Practice and Experience, 35(2), e7491.* https://doi.org/10.1002/cpe.7491

*Wezeman, S. U. (2007). AVS and UCAVS: Developments in the European Union. Available at:* http://www.europarl.europa.eu/activities/committeesstudies/download.do?file=19483

*Wong, S., Jassemi Zargani, R., & Kim, B. (2016). Counter-measures against drone surveillance. Defence Research and Development Canada–Ottawa Research Centre, Reference Document DRDC-RDDC-2016-D019.*

*Yilmaz, B., & Kutbay, U. (2024). YOLOv8-based drone detection: Performance analysis and optimization. Computers, 13(9), 234.* https://doi.org/10.3390/computers13090234