

# International Journal Research Publication Analysis

Page: 01-08

---

## SKYFUSION: A YOLOV8-BASED OBJECT DETECTION FRAMEWORK FOR IDENTIFYING AIRCRAFT, SHIPS AND VEHICLES IN SATELLITE IMAGERY

---

**\*Harini Vanapalli, G. N. R. Prasad**

*Department of MCA (Masters in Computer Applications) Chaitanya Bharathi Institute of  
Technology (CBIT), Gandipet, Hyderabad, Telangana, India.*

---

**Article Received: 17 September 2026**

**Article Revised: 07 October 2026**

**Published on: 27 October 2026**

**\*Corresponding Author: Harini Vanapalli**

Department of MCA (Masters in Computer Applications) Chaitanya Bharathi  
Institute of Technology (CBIT), Gandipet, Hyderabad, Telangana, India.

DOI: <https://doi-doi.org/101555/ijrpa.3404>

---

### ABSTRACT

Satellite imagery is generated at a scale that makes manual inspection slow, error-prone and impractical. This paper presents SKYFUSION, an end-to-end system that detects and counts aircraft, ships and vehicles in satellite images using the YOLOv8 deep learning detector. A YOLOv8m model is fine-tuned on a labelled satellite dataset of 2,992 images (2,094 training, 449 validation, 449 test) for 30 epochs using the Adam optimizer with a learning rate of 0.001 and a batch size of 32. The trained weights are integrated into a Django web application that lets a user log in, upload an image and instantly receive an annotated image with bounding boxes, confidence scores and per-class object counts. On the 449-image test set the system reports an overall accuracy of about 98%. Qualitative tests show that it can count 232 vehicles and one aircraft in a dense scene and locate a ship near a waterbody. The framework gives analysts a fast, accessible and extensible tool for surveillance, traffic analysis, urban planning and disaster response.

**INDEX TERMS:** satellite imagery, object detection, YOLOv8, deep learning, remote sensing, aircraft, ship and vehicle detection, Django web application

### 1. INTRODUCTION

Earth-observation satellites now produce petabytes of imagery every year, and the demand for turning this data into timely decisions has grown with it. Applications such as maritime surveillance, airport monitoring, traffic analysis, urban planning and post-disaster assessment all depend on knowing *what* is present in an image and *where*. Inspecting such images by

hand does not scale, and early automated approaches based on edge detection, template matching and handcrafted features such as HOG and SIFT struggled with changes in scale, illumination and cluttered backgrounds [1].

Convolutional neural networks changed this picture. Two-stage detectors such as Faster R-CNN [2] and single-stage detectors such as SSD [3] and YOLO [4] learn features directly from data and detect several object classes in one pass. Satellite images remain harder than natural photographs, however: objects are small, densely packed, rotated and seen from above, and the images themselves are very large. A practical system must therefore balance accuracy, inference speed and ease of use.

This paper presents **SKYFUSION**, an AI-based system for detecting aircraft, ships and vehicles in satellite images. The main contributions are: (i) a YOLOv8m detector fine-tuned on a three-class satellite dataset; (ii) an inference pipeline that draws bounding boxes and produces per-class object counts in a single pass; and (iii) a Django-based web interface that makes the model usable by non-technical users, with an evaluation of the complete workflow with a web interface so that detection results are directly usable.

### III. PROBLEM STATEMENT AND OBJECTIVES

*How can a single, fast and user-friendly system detect and count multiple object types in satellite imagery with high accuracy across varied scenes?* Conventional pipelines based on older detectors suffer from bottlenecks on large, high-resolution images with varied object scales, offer limited visualisation, often need manual pre-processing, and rarely count objects in the same pass. The objectives are to (a) detect aircraft, ships and vehicles accurately, (b) keep inference fast enough for near-real-time use, (c) provide an upload-and-view web interface, and (d) keep the design modular so that new classes can be added with little retraining.

### IV. PROPOSED METHODOLOGY

SKYFUSION has two phases (Fig. 1). In the **training phase**, the labelled dataset is pre-processed and augmented, and a pretrained YOLOv8m model is fine-tuned while loss values are monitored on the validation set; the best checkpoint is stored as *best.pt*. In the **inference phase**, a logged-in user uploads an image through the Django application, which loads *best.pt*, runs detection, filters and counts the detections, and returns the annotated image to the browser.

## II. RELATED WORK

Early satellite-image detectors adapted general-purpose networks. Chen *et al.* modified Faster R-CNN with multi-scale feature pyramids for post-disaster building detection and improved recall under occlusion, although processing time limited real-time use [5]. Jiang *et al.* used a GAN to enhance low-resolution satellite images before detection, which raised precision but made training unstable and sensitive to tuning [6]. For small targets, Zhan *et al.* combined Cascade R-CNN with contextual attention on urban aerial images [7], and Liu *et al.* proposed a scale-aware YOLO variant for maritime vessels [8]. Huang *et al.* showed that ensembling detectors helps on multi-class aerial imagery [9], while Zhang *et al.* reduced labelling effort with semi-supervised annotation [10].

These studies point to a recurring trade-off: higher accuracy is often bought with heavier models, longer training or more complex pipelines, and few of them are packaged for end users. SKYFUSION adopts YOLOv8 [11], whose anchor-free design and decoupled head give a good speed-accuracy balance, and pairs it

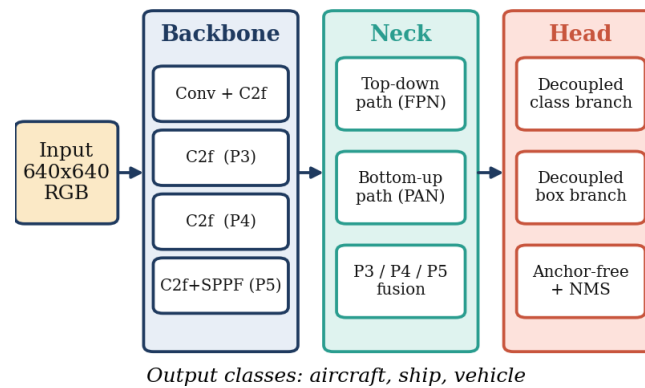


**Fig. 1. Overall architecture of SKYFUSION: training phase (left) and web-based inference phase (right).**

**A. Dataset.** The system uses the SkyFusion aerial-detection dataset from Kaggle, containing 2,992 images annotated for three classes: aircraft, ship and vehicle. The images cover varied terrain, lighting and scene types. The data are split into 2,094 training, 449 validation and 449 test images (Fig. 2, Table I).

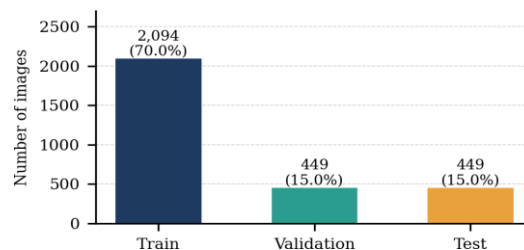


**Fig. 2. Distribution of images across the training, validation and test sets.**

**TABLE I. DATASET SUMMARY.**

| Subset     | Images | Share (%) | Purpose               |
|------------|--------|-----------|-----------------------|
| Training   | 2,094  | 70.0      | Weight optimisation   |
| Validation | 449    | 15.0      | Monitoring and tuning |
| Test       | 449    | 15.0      | Final evaluation      |
| Total      | 2,992  | 100       | 3 classes             |

**B. Pre-processing.** Images are resized to the input size expected by YOLOv8 so that every sample has consistent dimensions. Augmentation (flipping and random cropping) increases data variety and makes the detector more robust to orientation and scale changes, which matters for overhead imagery.

**C. YOLOv8 detector.** YOLOv8 [11] is a single-stage, anchor-free detector made of a CSP-style backbone with C2f blocks, a PAN-FPN neck that fuses features at three scales (P3, P4, P5), and a decoupled head with separate classification and box-regression branches (Fig. 3). Detecting at several scales helps with the very different object sizes found in satellite images, from small vehicles to large aircraft. The medium variant, YOLOv8m, was chosen as a compromise between accuracy and computational cost.

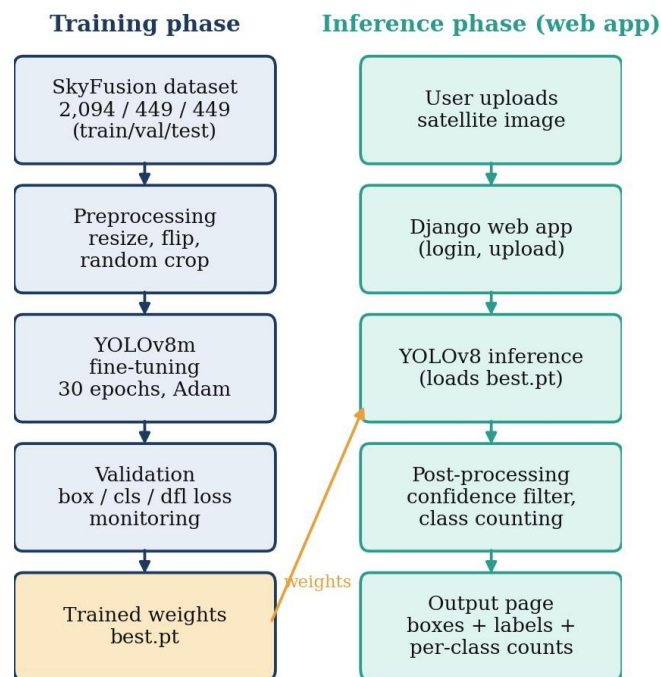


**Fig. 3. Simplified YOLOv8 architecture: backbone, neck and decoupled detection head.**

**D. Training.** The pretrained *yolov8m.pt* weights are fine-tuned on the dataset with the settings in Table II. Training minimises a weighted sum of box, classification and distribution-focal losses:

$$L = \square L + \square L + \square L \quad (1)$$

*box boxcls cls dfl dfl*

These three losses on the training and validation sets are logged for every epoch and used to check convergence and over-fitting. The weights with the best validation performance are saved as *best.pt*.

**TABLE II. TRAINING CONFIGURATION.**

| Parameter             | Value                   |
|-----------------------|-------------------------|
| Base model            | YOLOv8m (pretrained)    |
| Epochs                | 30                      |
| Batch size            | 32                      |
| Optimiser             | Adam                    |
| Initial learning rate | 0.001                   |
| Random seed           | 6                       |
| Classes               | aircraft, ship, vehicle |

**E. Inference and counting.** For an uploaded image, the model returns boxes, class labels and confidence scores. The system draws the boxes on the image and counts the detections of each class  $c$ :

$$N_c = \sum_i I[class_i = c], \quad c \in \{aircraft, ship, vehicle\} \quad (2)$$

So that a user obtains the location and the quantity of each object type from one run.

## V. SYSTEM IMPLEMENTATION

The system is implemented in Python 3.10 with Visual Studio Code as the development environment. Model training uses the Ultralytics YOLOv8 framework, with NumPy, Pandas, Matplotlib, OpenCV, Pillow and Supervision for data handling, visualisation and image processing. The web layer is built with Django, HTML, CSS and JavaScript, with MySQL as the specified database.

The application has four modules (Table III). The *Data Management* module organises and pre-processes the images; the *Model Training* module configures and trains YOLOv8; the *Detection and Inference* module loads *best.pt*, draws boxes and counts objects; and the *Web Interface* module provides login, a dashboard, an upload portal and a result page (Fig. 4). The model is loaded from disk once and reused, and the upload-to-result path runs in a single request.

TABLE III. SKYFUSION MODULES.

| Module                  | Responsibility                                                     |
|-------------------------|--------------------------------------------------------------------|
| Data Management         | Dataset organisation, resizing, augmentation, train/val/test split |
| Model Training          | YOLOv8m configuration, fine-tuning, loss monitoring                |
| Detection and Inference | Load best.pt, predict, draw boxes, count classes                   |
| Web Interface           | Login, dashboard, image upload, result display                     |

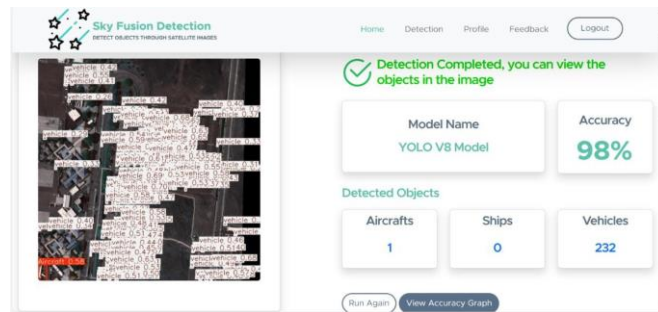


Fig. 4. Result page of the SKYFUSION web application showing the annotated image, model name, accuracy and per-class counts.

## VI. RESULTS AND DISCUSSION

**A. Quantitative result.** After training, the final model was evaluated on the 449-image test set. The system reports an overall accuracy of approximately 98%, which is also displayed on the result page of the web application. This indicates that the detector can reliably identify aircraft, ships and vehicles across the varied scenes in the test data.

**B. Qualitative results.** Fig. 5 shows two outputs of the deployed system and Fig. 6 summarises the counts that it returned. In a dense urban scene, SKYFUSION marked 232 vehicles and 1 aircraft and no ships (Fig. 5a). In a coastal scene with a waterbody, it found 1 ship and 3 vehicles and no aircraft (Fig. 5b). The correct absence of aircraft in the second scene and of ships in the first shows that the classes are being separated and not simply over-predicted a more complete picture and should be reported alongside it. Very dense scenes can produce overlapping detections, and performance on classes or sensors outside the training data has not been tested. The system processes one uploaded image at a time and is not yet connected to a live satellite feed.

## VII. APPLICATIONS AND ADVANTAGES

SKYFUSION supports maritime surveillance (vessel monitoring), airport and airfield monitoring, traffic and parking analysis from vehicle counts, urban planning, and rapid damage assessment after disasters. Its main advantages are accurate multi-class detection, fast single-pass inference, automatic object counting, a browser-based interface that needs no technical expertise, and a modular design to which new classes can be added with minimal retraining.

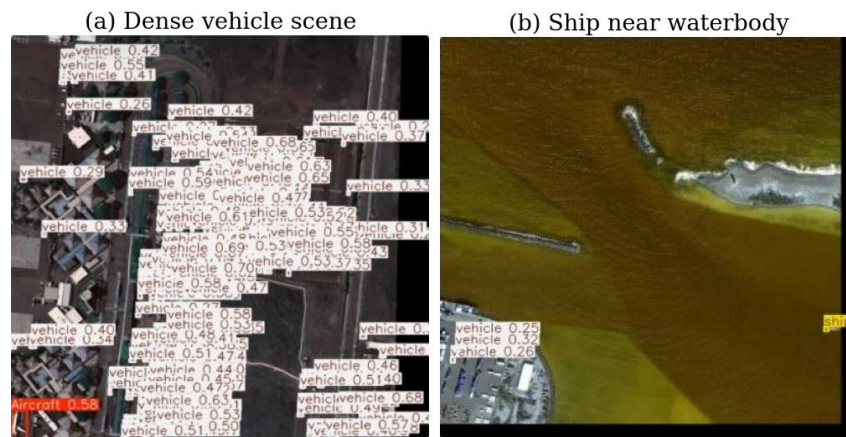


**Fig. 5. Detection outputs of SKYFUSION with bounding boxes and confidence scores.**

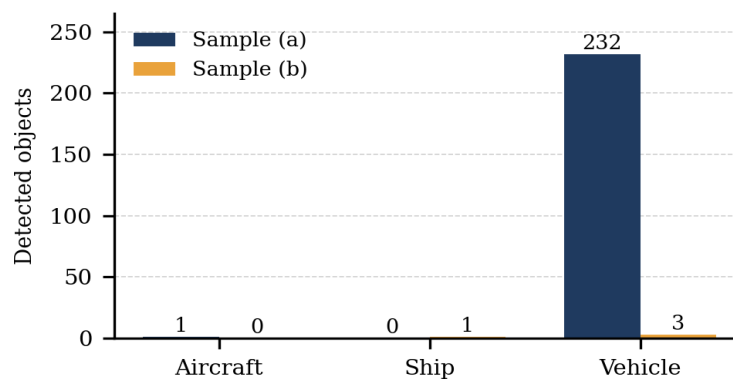


**Fig. 6. Per-class object counts returned for the two sample images.**

## VIII. CONCLUSION AND FUTURE SCOPE

This paper presented SKYFUSION, a YOLOv8-based system that detects and counts aircraft, ships and vehicles in satellite imagery and exposes the model through a Django web application. Fine-tuned on 2,094 training images, the model reached about 98% overall accuracy on a 449-image test set and gave sensible detections in both dense and sparse scenes. Future work includes adding classes such as buildings and roads, using multispectral and hyperspectral bands to improve detection in poor visibility, reporting full mAP and per-class metrics, comparing against other detectors, tiling very large images, and deploying the model on edge or cloud platforms linked to real-time satellite streams and GIS tools.

**c. DISCUSSION.** The first sample is a demanding case: hundreds of small, closely packed vehicles produce overlapping labels, and confidence scores range from about 0.26 to 0.70, so low-confidence vehicles appear alongside confident ones. Even so, the model keeps up with the density of the scene and returns a usable count in one pass, which would be tedious to

obtain manually. The web interface adds practical value: a non-expert can upload an image and read the result without any code. Testing with satellite and drone-captured images also guided refinements to the interface layout, speed and stability.

#### **D. LIMITATIONS**

The accuracy figure is a single overall number; class-wise precision, recall and mAP, and a confusion matrix, give

#### **REFERENCES**

- 1 A. Krizhevsky, I. Sutskever and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. NIPS*, 2012.
- 2 S. Ren, K. He, R. Girshick and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Proc. NIPS*, 2015.
- 3 W. Liu *et al.*, “SSD: Single shot multibox detector,” in *Proc. ECCV*, 2016, pp. 21–37.
- 4 J. Redmon, S. Divvala, R. Girshick and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. IEEE CVPR*, 2016, pp. 779–788.
- 5 X. Chen, J. Li and Y. Fan, “A modified Faster R-CNN for post-disaster building detection in satellite imagery,” *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 5, pp. 3552–3567, 2019.
- 6 L. Jiang, T. Xue and W. Yang, “GAN-based low-resolution satellite image enhancement for object detection,” *ISPRS J. Photogramm. Remote Sens.*, vol. 149, pp. 1–12, 2019.
- 7 R. Zhan, X. Guo and Y. Li, “Small object detection in urban aerial images via Cascade R-CNN with contextual attention,” *Remote Sens.*, vol. 13, no. 20, p. 4088, 2021.
- 8 F. Liu, R. Yang and M. Zhu, “Scale-aware YOLO for detecting maritime vessels in satellite imagery,” *IEEE Access*, vol. 10, pp. 51234–51247, 2022.
- 9 W. Huang, G. Chen and P. Zhang, “Ensemble strategy for multi-class aerial imagery detection,” *Remote Sens.*, vol. 15, no. 2, p. 321, 2023.
- 10 J. Zhang, S. Liu and W. Zhang, “Semi-supervised labeling for satellite imagery: Accelerating object detection data generation,” *ISPRS J. Photogramm. Remote Sens.*, vol. 194, pp. 160–172, 2023.
- 11 G. Jocher, A. Chaurasia and J. Qiu, “Ultralytics YOLOv8,” 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>