Optimal Deep Learning-Based Vehicle Detection and Classification Using Chaotic Equilibrium Optimization Algorithm in Remote Sensing Imagery

1. Introduction The exponential growth of urban populations and corresponding increases in traffic have necessitated smarter, more efficient vehicle monitoring systems. Remote sensing imagery (RSI) offers a robust solution by capturing wide-scale aerial and satellite images for vehicle detection and classification. This report explores a novel framework named VDTC-CEOADL that integrates deep learning with a chaotic equilibrium optimization algorithm to enhance vehicle detection and classification in RSIs.

2. Problem Statement Detecting vehicles from RSIs poses several challenges, including:

- Small object sizes.
- Complex backgrounds.
- Varying object orientations.
- Class similarities and occlusions.

Traditional approaches like classical machine learning or basic CNN models fall short in accuracy and computational efficiency. Hence, there's a need for an optimized, deep learning-based solution.

3. Proposed Solution: VDTC-CEOADL Framework The VDTC-CEOADL (Vehicle Detection and Type Classification using Chaotic Equilibrium Optimization Algorithm with Deep Learning) framework combining multiple cutting-edge techniques:

3.1 YOLO-HR for Object Detection

- A high-resolution variant of YOLO tailored for RSIs.
- Incorporates ResNet as the backbone for improved feature extraction.
- Uses PANet and SPPF for multi-scale feature fusion.

3.2 Chaotic Equilibrium Optimization Algorithm (CEOA)

- A physics-inspired metaheuristic optimization algorithm.
- Enhances YOLO-HR by optimizing hyperparameters.
- Integrates chaotic maps (logistic map) to improve search diversity and avoid local minima.

3.3 Attention-Based LSTM (ALSTM) for Classification

- Leverages long-term dependencies and attention mechanisms.
- Uses Luong's multiplicative attention to emphasize relevant features.

4. Dataset and Preprocessing Two public datasets were used:

- **VEDAI:** 3687 vehicle samples, 9 classes, 12.5 cm/pixel resolution.
- **ISPRS Potsdam:** 2244 samples, 4 classes, 5 cm/pixel resolution.

Preprocessing steps included:

- Resizing to 512x512 pixels.
- Normalization and data augmentation (rotation, flipping, brightness).
- Patch-based segmentation for large images.

5. Evaluation Metrics The following metrics were used for performance evaluation:

- Accuracy
- Sensitivity
- Specificity
- F1-Score
- AUC (Area Under Curve)

6. Results VEDAI Dataset:

- Accuracy: 99.11%
- F1-Score: 88.08%
- AUC: 92.01%

ISPRS Potsdam Dataset:

- Accuracy: 99.11%
- F1-Score: 79.58%
- AUC: 85.20%

Comparative Analysis:

- Outperformed VGG-19, LeNet, TFC, and AMS-DAT in both accuracy and computational time.
- Computational time: VDTC-CEOADL ~3.93s vs. VGG-19 ~8.73s

7. Discussion The VDTC-CEOADL framework demonstrated superior performance across multiple datasets and metrics. It efficiently handled small object sizes and class variances through optimized detection and classification modules. However, the high computational load and dependency on annotated data remain limitations.

8. Future Work

- Explore semi-supervised and weakly supervised learning methods.
- Develop lightweight models for real-time deployment.
- Automate data labeling to reduce manual annotation efforts.

9. Conclusion The VDTC-CEOADL model successfully integrates deep learning and physics-inspired optimization to deliver high-performance vehicle detection and classification in remote sensing imagery. Its modular design and robust results position it as a valuable tool for smart city surveillance and urban planning applications.

Acknowledgements This research was funded by Umm Al-Qura University, Saudi Arabia under

Authors

- Youseef Alotaibi
- Krishnaraj Nagappan
- Tamilvizhi Thanarajan
- Surendran Rajendran

Object Detection Using YOLOv8

1. Introduction Object detection is a key application in computer vision, supporting a range of use cases such as surveillance, autonomous driving, and face recognition. Traditional methods often lack the speed and accuracy required for real-time applications. YOLO (You Only Look Once) algorithms, especially the latest YOLOv8, provide fast and accurate solutions for detecting and classifying objects in static and dynamic visual data.

2. Objective This project aims to implement an object detection system using YOLOv8 that can process images, videos, and live webcam feeds to identify and localize multiple objects with high accuracy and speed.

3. Existing System Previous object detection methods heavily relied on manual intervention and lacked real-time capabilities. Systems were prone to errors and delays, making them unsuitable for critical applications like surveillance or self-driving vehicles.

4. Proposed System The proposed system uses YOLOv8 to process images, videos, and live feeds. Key features include:

- Real-time processing at 150 FPS.
- Recognition of 85 distinct object classes.
- Utilization of pre-trained and custom models.
- A simple and user-friendly interface.

5. Literature Review

- Early work by Lowe introduced SIFT for robust local feature detection.
- Papageorgiou et al. used wavelets and SVMs for object recognition.
- Girshick's Fast R-CNN improved speed and accuracy.
- YOLO evolved through versions 1 to 8, improving architecture, feature fusion, anchor-free detection, and training methods.
- YOLOv8 outperforms alternatives like EfficientDet and DETR in speed and real-time performance.

6. Methodology A. Input Formats:

- Image, video, or live webcam stream.

B. YOLOv8 Algorithm:

- Uses CNN with CSPDarknet architecture to extract features.
- Feature Pyramid Network (FPN) enables multi-scale object detection.
- Outputs bounding boxes, confidence scores, and class probabilities.

C. Tools & Requirements:

- Language: Python
- Minimum Hardware: 4GB RAM, SSD, Windows 11

7. Dataset & Preprocessing

- Custom dataset includes ID cards, tags, and facial features collected manually and via Google Images.
- Annotation done using Roboflow in YOLO format.
- Preprocessing includes auto-orientation, resizing (650x650), and noise injection.

8. Training

- Labeled images are processed through CNN.
- Feature maps passed through FPN.
- YOLO heads predict bounding boxes and class probabilities.
- Custom loss function used for training.
- Pre-trained YOLOv8n model with 80 classes was used to expedite training.

9. Results

- **Precision:** 0.85
- **Recall:** 0.80
- **mAP:** 0.82

Additional Metrics:

- Confusion matrix shows class-wise performance.
- F1-Confidence curve and Precision-Recall curve demonstrate strong model accuracy and robustness.

10. Comparison with Other Algorithms

Algorithm	Precision	Recall	mAP
YOLOv8	0.85	0.80	0.82
YOLOv7	0.83	0.78	0.80
Faster R-CNN	0.86	0.83	0.84
SSD	0.80	0.75	0.78
EfficientDet	0.85	0.81	0.83

11. Output Examples

- Object detection was tested on:
    - Downloaded images

  - Recorded video
  - Live webcam

- Successfully detected objects like eyes, hands, ID cards, mouths, and tags.

12. Conclusion and Future Enhancements This project successfully demonstrates real-time object detection using YOLOv8. The system is accurate, fast, and flexible across various visual data types.

Future improvements:

- Enhance accuracy and speed further.
- Add more advanced detection features.
- Improve UI and scalability for broader deployment.

Authors:

- Karthika B
- Dharssinee M
- Reshma V
- Venkatesan R
- Dr. Sujarani R