UrbanEye

Smart Surveillance for Cleaner Cityscapes

HUSSAIN ALSHABAAN, ABDULLAH M AL-AWLAQI, RAYAN ALSUBHI

ICS504 – Deep Learning Term Project, KFUPM 242

Abstract

The **UrbanEye** project explores the application of deep learning-based object detection techniques for the automatic classification of urban street issues, specifically targeting 11 distinct categories such as graffiti, potholes, construction roadblocks, and broken signage. Leveraging the YOLOv11 architecture, known for its high efficiency and real-time performance, the model processes urban scene images to accurately identify and categorize visual irregularities. A curated dataset consisting of 7,873 annotated images was utilized, ensuring a representative distribution across various types of urban defects. In addition to conventional training and inference pipelines, the project integrates hyperparameter optimization using an evolutionary algorithm. This technique involves iterative training cycles where model configurations are selectively blended and mutated based on performance evaluation, leading to progressive improvements in detection accuracy and robustness. By incorporating synthetic data generation, enhanced data augmentation strategies, and a customized loss function based on class weights, the UrbanEye framework significantly addresses dataset imbalance issues. The resulting robust and scalable model aims to enhance urban infrastructure monitoring and management, offering valuable tools to municipalities and smart city systems for improving public safety, environmental aesthetics, and quality of life.

Table of Contents

3. Introduction

Learning and computer vision technologies presents a promising avenue for automating detection and classification tasks in urban environments, enabling more efficient and scalable solutions.

This project aims to address these practical needs by developing a robust object detection model tailored for urban scene analysis. The task involves significant challenges such that variations in lighting conditions, image quality inconsistencies, object occlusion, and a diverse range of issues—from potholes to graffiti—demanding a sophisticated system capable of handling real-world complexities. Leveraging the YOLOv11 architecture, we investigate its ability to create a multi-class detection pipeline that accurately identifies and classifies various urban street issues.

Improving existing models is crucial, not just through architectural upgrades—such as transitioning to more advanced frameworks like YOLOv11—but also by refining their training and fine-tuning processes. Techniques like data augmentation, regularization, and sophisticated optimization methods (e.g., learning rate scheduling and hyperparameter tuning) can greatly enhance a model's generalization and mitigate overfitting in addition to contribute to making models more resilient, accurate, and suitable for real-time visual pollution detection in diverse environments.

3.1 Problem Statement

Cities today are struggling with increasing levels of visual pollution issues like graffiti, broken or faded signage, potholes, and poorly maintained infrastructure, all of which negatively impact the urban experience. Traditional monitoring methods rely heavily on manual inspections or citizen reporting, which are often time-consuming, inconsistent, and difficult to scale. While computer vision models have been applied in the past, many of these systems are limited to detecting single types of visual issues or struggle to operate reliably in diverse real-world conditions.

The core challenge lies in building a robust, automated solution that can accurately detect multiple forms of visual pollution across varied urban environments. This project addresses these limitations by adopting the latest YOLOv11 architecture to develop a more capable detection system. The aim is to improve classification accuracy, reduce false positives, and support large-scale implementation in smart city initiatives. The target issues include:

Category	Description
GRAFFITI	Vandalism on walls
FADED_SIGNAGE	Worn-out road signs
POTHOLES	Road surface damage
GARBAGE	Littering in public spaces

CONSTRUCTION_ROAD	Ongoing roadwork
BROKEN_SIGNAGE	Damaged traffic signs
BAD_STREETLIGHT	Faulty or broken streetlights
BAD_BILLBOARD	Damaged/unmaintained advertisements
SAND_ON_ROAD	Sand/debris obstructing roads
CLUTTER_SIDEWALK	Obstructions on pedestrian pathways
UNKEPT_FACADE	Poorly maintained building exteriors

Table 1

3.2 Objective

This project aims to develop a YOLOv11-based deep learning model for real-time detection and classification of 11 urban street anomalies that involve:

1. **Dataset Preparation:** Preprocessing and annotating urban street images
2. **Model Training & Optimization:** Training and fine-tuning the model, evaluating its performance using metrics like mean Average Precision (mAP).
3. **Generalization Testing:** Assessing the model's adaptability by testing it on unseen urban datasets with different city layouts, and camera angles.
4. **Comparative Analysis:** Benchmarking the model against vanilla YOLOv11 model with fine-tuned and modified model of YOLOv11 to demonstrate improvements in accuracy and efficiency.

3.3 Scope of Study

This study focuses on developing and evaluating a deep learning-based object detection system using YOLOv11 to automatically identify and classify 11 types of urban street issues, such as graffiti, potholes, damaged signage, litter, and neglected facades. The model is trained and validated on a dataset of 7,873 images, comprising 5,511 training images, 1,181 evaluation images, and 1,181 test images. All images are preprocessed and formatted for YOLOv11 compatibility. The project includes model optimization through hyperparameter tuning, augmentation, synthetic image generation and custom loss calculation function, with performance measured using accuracy, precision, recall, and mAP. The goal is to create

a scalable, automated detection system to support urban infrastructure monitoring and reduce dependence on manual inspection.

4. Literature Review

4.1 Overview of existing techniques for object detection

The table below shows some of the famous object detection models.

Model	Architecture Type	Real-time Capability	Accuracy	Small Object Detection	Parameter Efficiency	Deployment Complexity
YOLOv11	Advanced single-stage CNN	Very High	Very High	Very Good	Very High	Low
YOLOv9	Advanced single-stage CNN	High	Very High	Good	Very High	Low
RT-DETR	Hybrid CNN-Transformer	High	High	Good	High	Moderate
RTMDet	Optimized CNN-Transformer	High	High	Good	High	Low
EfficientDet	CNN with compound scaling	Moderate-High	High	Good	Very High	Low
DINOv2	Transformer-based	Low	Very High	Very Good	Moderate	High
Mask R-CNN	Two-stage detector	Low	Very High	Good	Low	Moderate
YOLO (original)	Single-stage CNN	High	Moderate	Limited	High	Low
ResNet-50	CNN with residual connections	Moderate	High	Moderate	Moderate	Low
DenseNet-121	CNN with dense connections	Moderate	High	Good	High	Low

Table 2 – Object Detection Models [1]

4.1.1 Review of deep learning models and their applications in object detection

A range of deep learning models have been employed in object detection tasks relevant to visual pollution detection, each offering trade-offs between speed, accuracy, and deployment feasibility. Traditional CNN-based backbones such as ResNet-50 and DenseNet-121 deliver strong accuracy with moderate real-time performance, making them suitable for static image analysis but less optimal for real-time deployment scenarios. Single-stage detectors like the original YOLO architecture prioritize speed and efficiency; however, they have historically struggled with small object detection, which is critical when identifying fine-grained urban issues such as potholes or broken signage.

Recent models such as YOLOv9 and RT-DETR achieve a better balance between accuracy and real-time performance, improving small object detection while maintaining deployment efficiency. Building on these advancements, YOLOv11 introduces significant improvements through enhanced augmentation strategies, synthetic data integration, and adaptive loss functions [2]. These enhancements address class imbalance challenges and make YOLOv11 particularly well-suited for complex urban environments, where the detection of rare or subtle types of visual pollution is crucial for effective monitoring and management.

Transformer-based models, including DINOv2 and Mask R-CNN, further advance detection accuracy and feature extraction capabilities but often at the expense of speed and computational efficiency, limiting their suitability for real-time, resource-constrained deployments. Models such as EfficientDet and RTMDet distinguish themselves by offering high accuracy combined with parameter efficiency, making them strong candidates for scalable, edge-based urban monitoring systems. Overall, the evolution from conventional CNNs to hybrid and transformer-based architectures reflects a steady shift towards more accurate, versatile, and deployment-friendly detection systems, with YOLO variants and CNN-transformer hybrids emerging as leading solutions for visual pollution detection applications.

4.1.2 Gaps in the literature and the need for improvement

While deep learning models have improved urban issue detection, several limitations still affect their real-world effectiveness. Challenges range from detection accuracy to operational scalability in diverse urban settings. Addressing these gaps is essential for developing reliable, high-performance visual pollution detection systems.

- **Detection Limitations**: Difficulty recognizing small or occluded objects and inconsistent performance under changing lighting or environmental conditions.

- **Dataset Challenges**: Incomplete annotations, class imbalance, and poor generalization to unfamiliar urban features.

- **Performance Trade-offs**: Balancing speed and accuracy remain difficult, especially with resource-heavy models and high-resolution inputs.

- **Deployment Barriers**: Limited edge-device compatibility, model drift over time, and lack of integration with other sensing technologies.

- **Opportunities for Improvement**: Hybrid architecture, few-shot learning, standardized datasets, and multi-scale feature optimization offer promising solutions.

4.2 Related Work

Visual pollution detection has attracted significant attention due to its environmental and aesthetic impacts. Early approaches relied on manual surveys or citizen reporting, which are time-consuming and subjective. With the advancement of computer vision, several studies have employed deep learning models to automate the detection of visual pollutants. For instance, Titu et al. [3] proposed a real-time detection system using models such as **YOLOv5** and **EfficientDet**, deployed on edge devices for urban and textile environments. Elbaz et al. [4] introduced a 3D-CNN approach enhanced with an attention mechanism for real-time air quality forecasting based on sky images. These studies demonstrate the potential of deep learning frameworks in environmental monitoring tasks.

Recent works have also explored active learning to improve model training efficiency by reducing the annotation burden. Alzu'bi et al. [2] proposed a deep active learning framework for visual pollution detection using public road images, significantly minimizing labeling efforts. Among the related studies, the work by AlElaiwi et al. [5] is the most relevant to this research. Their Visual Pollution Prediction (VPP) framework applies a deep active learning approach to public road images collected in Saudi Arabia, achieving high accuracy with reduced annotation costs. The regional focus and methodological design of their study make it particularly aligned with the objectives of this work. However, challenges remain in terms of generalization to diverse pollution categories and adaptability to different urban environments, motivating further improvements in visual pollution prediction frameworks.

4.3 Limitation in Existing Approaches

Current YOLO-based models, while highly efficient in real-time object detection [6], present notable limitations when applied to complex and imbalanced datasets such as visual pollution detection. Standard YOLO implementations primarily rely on static augmentation strategies [7] and conventional loss functions [8] that do not adequately address class imbalance. As a result, models tend to perform well on dominant classes but struggle to accurately detect minority or underrepresented categories.

Moreover, traditional loss functions in existing approaches calculate error uniformly across all classes, regardless of the class source or significance [8]. This uniform treatment leads to biased learning where majority classes dominate the optimization process. Without dynamic loss adjustment based on input source or class importance, models exhibit reduced recall and precision for less frequent classes.

To overcome these limitations, this project proposes an enhanced framework that introduces dynamic augmentation [9], adapting transformations based on dataset distribution, and implements a customized loss function that considers class weights while calculating the loss during training [10]. This ensures that each class, especially minority ones, has an appropriate influence on the model updates, leading to more balanced and robust detection performance across all visual pollution categories.

5. Proposed Methodology

The UrbanEye project explores the application of deep learning-based object detection for the automatic classification of urban street issues, specifically targeting 11 distinct categories such as graffiti, potholes, and broken signage. Leveraging the YOLOv11 architecture [11], known for its superior performance and speed, the model processes urban scene images to identify and categorize visual irregularities.

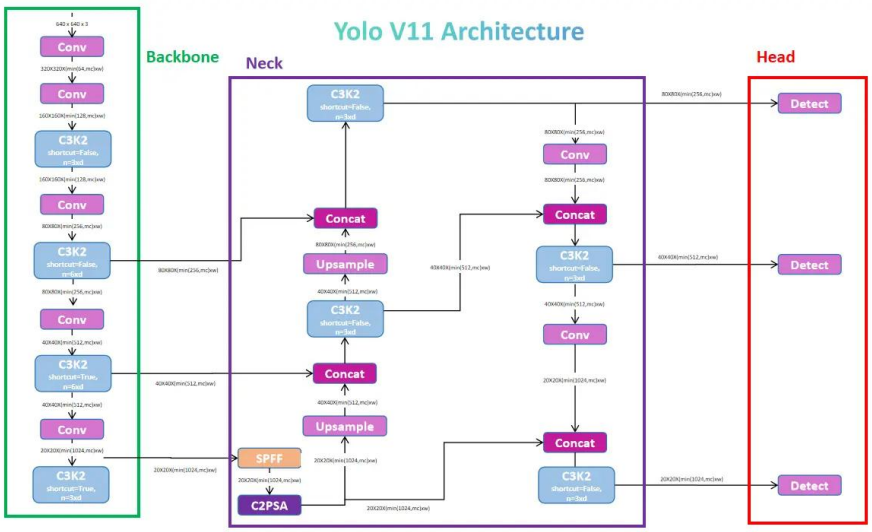

Figure 1 [12]

At its core, YOLOv11 consists of three fundamental components that contribute to its effectiveness in urban issue detection:

1. Backbone
a. Convolutional Block and Bottle Neck

Convolutional Block is named as Conv Block which process the given c,h,w passing through a 2D convolutional layer following with a 2D Batch Normalization layer at last with a SiLU Activation Function
Bottle Neck is a sequence of convolutional block with a shortcut parameter; this would decide if you want to get the residual part or not. It is similar to the ResNet Block, if shortcut is set to False, then no residual would be considered.

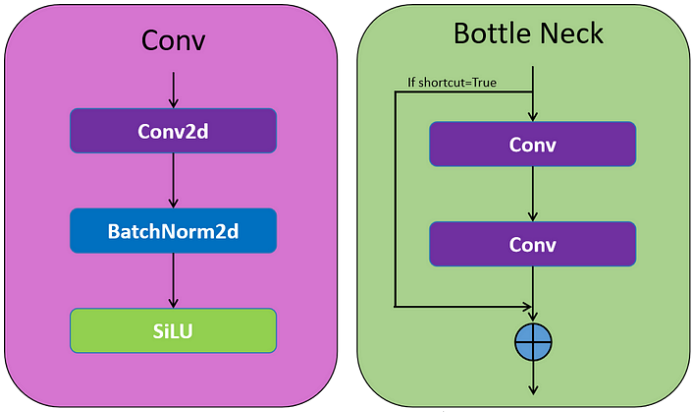

Figure 2 [12]

b. C3K2 Utilizing advanced C3K2 (Cross Convolution Block with 3×3 Kernels) modules for efficient feature extraction. This component incorporates SPPF (Spatial Pyramid Pooling Fast) to consolidate features from multiple receptive fields, enhancing detection across various object scales.
C3K2 is an Efficient feature extraction module derived from Cross Stage Partial (CSP) bottlenecks, optimized to reduce model complexity while maintaining strong feature representation across different image scales

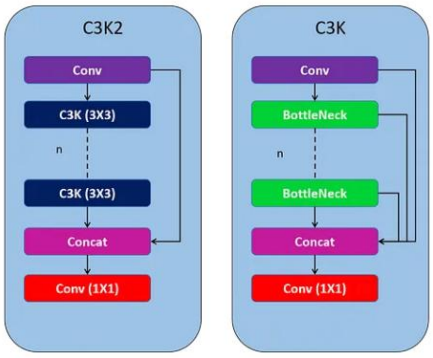

Figure 3 – C3K2 block [12]

Transition Layer (Multi-Scale Pooling (SPPF))
At the end of the backbone, before the neck begins, YOLOv11 introduces a Spatial Pyramid Pooling Fast where SPPF Aggregates **multi-scale features** by pooling different spatial regions, improving the model's ability to detect objects of **varying sizes** without significantly slowing down inference.
This module enriches the feature maps with contextual information critical for downstream refinement

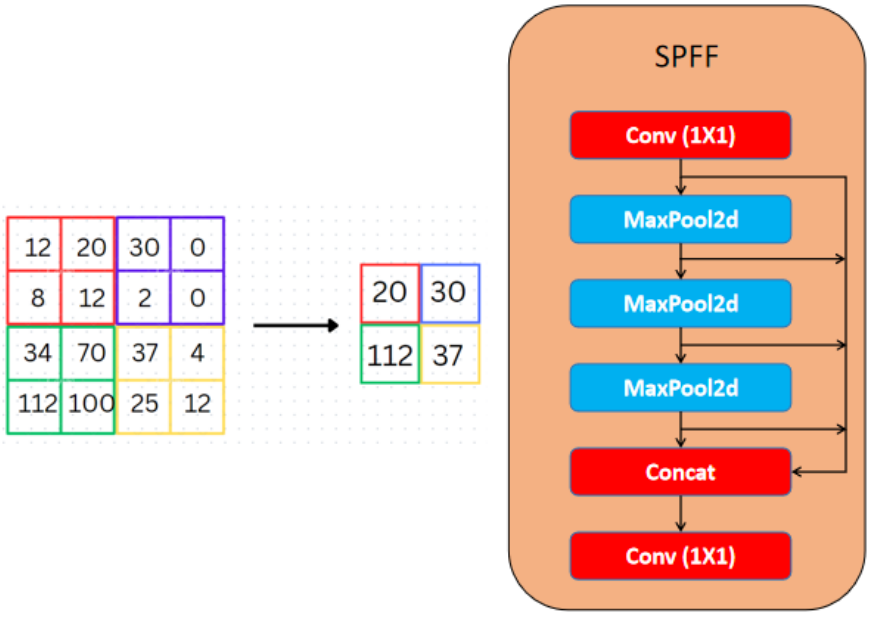

Figure 4 - Spatial Pyramid Pooling Fast block [12]

2- Neck (Feature Refinement and Attention)
The neck acts as a feature processing bridge between the backbone and the detection head. It consists of:
***C3K2 Blocks**:*
Continuing efficient processing and further strengthening feature connections between different scales.
C2PSA (Cross Stage Partial with Spatial Attention) blocks:
Enhance spatial attention mechanisms, allowing the model to focus more effectively on critical and more informative regions. This is particularly valuable when detecting partially hidden or small objects

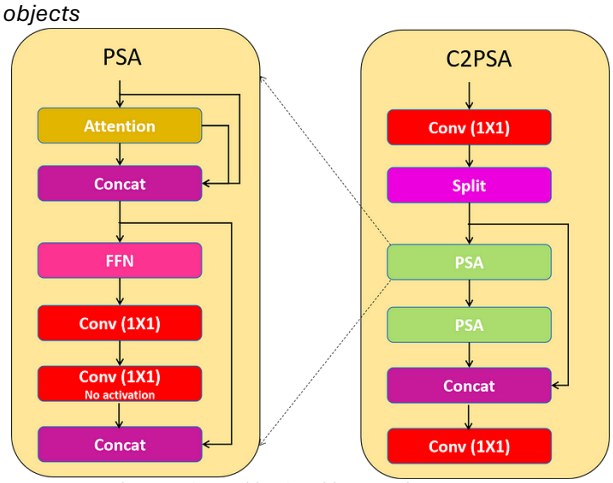

Figure 5 – C2-Position Sensitive Attention Block [12]

6. **Head (**Prediction and Output)

Employs multiple C3K2 blocks and **CBS (Convolution-BatchNorm-SiLU)** layers to efficiently process feature maps, generating precise bounding box coordinates and classification probabilities.

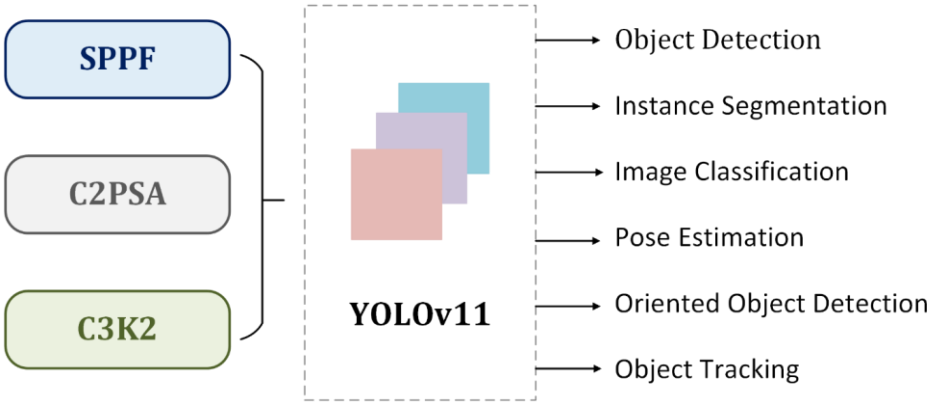

Figure 6 [11]

6.1 Existing Model and Challenges

The YOLOv11 architecture, an advanced single-stage object detection model, was selected due to its high efficiency and accuracy in real-time scenarios. While effective, YOLOv11 still faces several notable challenges, including:

- **Class Imbalance:** Uneven class representation in the dataset, which adversely affects detection performance, especially for minority classes.
- **Small Object Detection:** Difficulty in detecting small or partially occluded objects, such as potholes or faded signage, which are small relative to the overall image frame or partially obscured.
- **Real-world Generalization:** Sensitivity to variations in lighting, background clutter, and differing camera angles, impacting overall accuracy and generalization.
- **Limited Dataset Diversity:** Despite extensive annotation, the complexity and variability of real-world scenarios require a more diverse training set to optimize performance.
- **Label Quality:** Labels provided in the dataset were not as tight or precise as ideal for optimal training. Although this issue was identified, adjustments were reserved for future improvements.

6.2 Proposed Enhancement

Several enhancements were implemented to address the identified limitations:

- **Synthetic Image Generation:** Synthetic data generation was employed to balance the dataset by augmenting underrepresented classes up to the median representation level.

- **Weighted Loss Function:** The binary cross-entropy (BCE) loss function was modified to incorporate dynamically calculated class weights to address class imbalance.
- **Customized Data Augmentation:** Distinct augmentation strategies were applied separately to original and synthetic images, enhancing the model's generalization capabilities.

These enhancements collectively aim to improve the detection accuracy and robustness of the model, particularly in challenging urban scenarios.

6.3 Algorithms and implementation

Commented [RA1]: @ABDULLAH MABKHOT ABDULLAH M AL-AWLAQI epoch & early stopping

Commented [RA2R1]: Find in code, mention max epoch, early stopping and patience

The project utilized the YOLOv11 algorithm, incorporating specific enhancements:

- **YOLOv11l Model:** A CNN-based, single-stage object detection architecture optimized for real-time inference.

Performance

Detection (COCO) | Segmentation (COCO) | Classification (ImageNet) | Pose (COCO) | OBB (DOTAv1)

See Detection Docs for usage examples with these models trained on COCO, which include 80 pre-trained classes.

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed T4 TensorRT10 (ms)	params (M)	FLOPs (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

Figure 7 - Comparison between different YOLO11 models

- **Dynamic Synthetic Data Generation:** Synthetic images were generated per class dynamically based on dataset statistics.
- **Customized Loss Function:** A modified BCE loss incorporating class weight was integrated to ensure balanced learning across classes. This approach dynamically adjusts the loss based on the importance of each class, ensuring that minority classes contribute significantly to the gradient updates during training.
- **Early Stopping:** Early stopping was implemented to prevent overfitting by monitoring the validation loss during training. Training was halted when the validation performance plateaued, ensuring optimal model generalization and reducing the risk of overfitting to the training set.

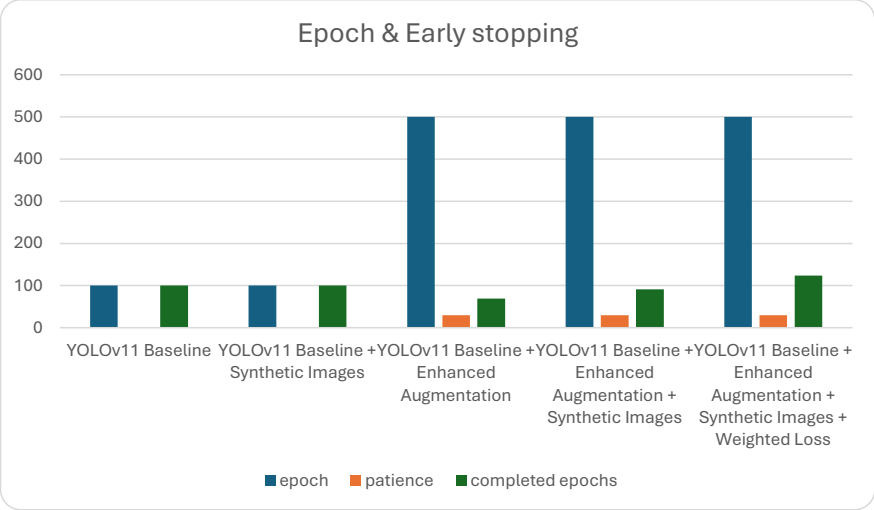

Figure 8 – Early stopping

6.4 Loss Function and Optimization

The training utilized a modified version of **YOLOv11**'s standard loss function to better address class imbalance and improve classification accuracy across underrepresented categories:

- **BCE Loss with Class Weighting**:
  The standard binary cross-entropy (BCE) loss used in YOLOv11 was enhanced by explicitly integrating **class weights** into the loss calculation. Instead of treating all classes equally, the modified approach assigns higher importance to minority classes based on their frequency distribution. During training, a **class weight tensor** is computed and applied dynamically to the BCE loss term. For each sample, the weight corresponding to the ground truth class is selected and multiplied with the loss value, effectively increasing the gradient contribution from underrepresented classes. This modification helps the model learn better decision boundaries for rare or subtle pollution categories without overwhelming the learning process.

Original BCE Loss **[13]**

$$\text{BCE Loss} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{\text{BCE}}(\hat{y}_i, y_i)$$

Weighted BCE Loss

$$\text{Weighted BCE Loss} = \frac{1}{S}\sum_{i=1}^{N} w_{c_i} \cdot \mathcal{L}_{\text{BCE}}(\hat{y}_i, y_i)$$

where:

- W_{ci} is the class weight for the true class c_i of sample i,

- S is the sum of positive samples (i.e., $S = \sum_{i=1}^{N} yi$) to normalize the loss properly.

- **Distribution Focal Loss (DFL)**:
  For bounding box regression, **Distribution Focal Loss** was used, which models the bounding box coordinates as discrete distributions rather than fixed points, improving localization accuracy, especially for small and hard-to-detect objects.
- **Optimization Strategy**:
  An **adaptive optimization strategy** was employed based on the scale of training. For larger training runs (over 10,000 iterations), **Stochastic Gradient Descent (SGD)** with a learning rate of 0.01 and momentum of 0.9 was selected to ensure stable convergence and better generalization. For smaller training runs (fewer than 10,000 iterations), **AdamW** was automatically chosen, with the learning rate dynamically adjusted according to the number of classes [14]. This approach balances fast convergence during smaller runs with the robust optimization characteristics needed for larger, more complex training sessions.

Overall, the introduced class-weighted loss mechanism combined with advanced optimization strategies significantly improved the model's robustness to class imbalance and contributed to higher detection performance on underrepresented visual pollution classes.

7. Experimental Design and Evaluation

A structured experimental approach was adopted, comprising:

- **Baseline Model:** Original YOLOv11 without modifications.
- **Baseline Model with synthetic data:** Original YOLOv11 without modifications and with synthetic data balancing.
- **Enhanced Augmentation Model:** YOLOv11 with enhanced augmentation
- **Enhanced Augmentation with synthetic data Model:** YOLOv11 with enhanced augmentation
- **Final Enhanced Model:** Combined approach utilizing synthetic data and class-weighted loss.

7.1 Datasets and Preprocessing

The utilized dataset includes:

- **Total Images:** 7,873 (Training: 5,511; Validation: 1,181; Testing: 1,181).
- **Categories:** Initially 11 distinct classes representing various urban street issues; the class "BAD_STREETLIGHT" was excluded due to insufficient data (only one sample provided).
- **Original Image Resolution:** The dataset images originally had a resolution of 1920x1080 pixels. These images were resized to 640x640 pixels to align with YOLOv11's input requirements, significantly reducing computational load and facilitating efficient real-time inference.
- **Label Format Conversion:** Original labels were provided in diverse annotation formats, necessitating conversion to the YOLO-compatible format to ensure compatibility and effective training.
- **Crop-based Synthetic Image Generation:** This method was specifically chosen to avoid unintentionally increasing instances of majority classes, as images containing minority classes could also include majority class objects. Additionally, strong augmentation methods (Mosaic, MixUp, Copy-Paste) were disabled for synthetic images to maintain their effectiveness.
- **Geography:** All images were taken in Riyadh, Saudi Arabia. It is also worth mentioning that all images were taken in broad daylight and with clear skies.

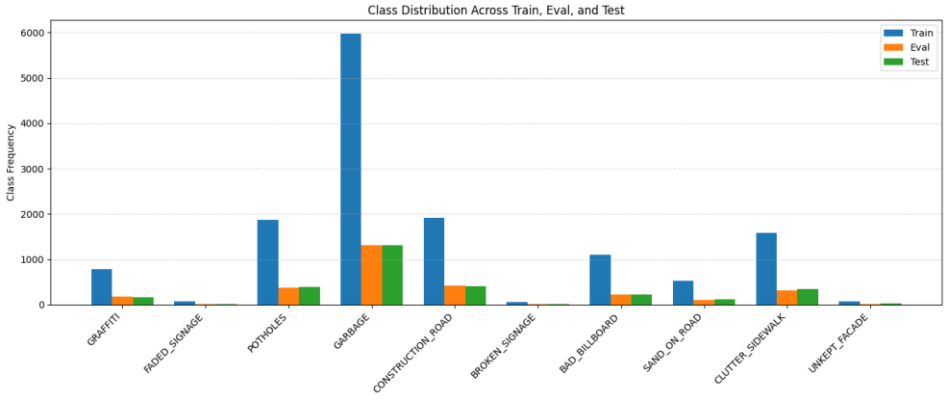

Figure 9

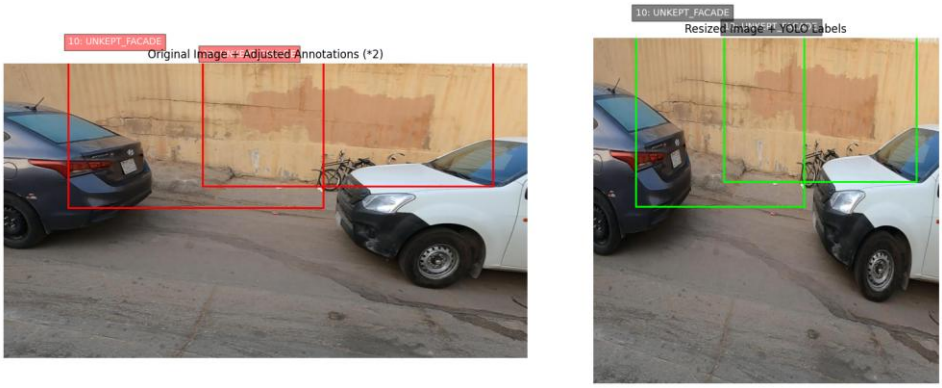

Figure 10 - Image Resizing for YOLOv11

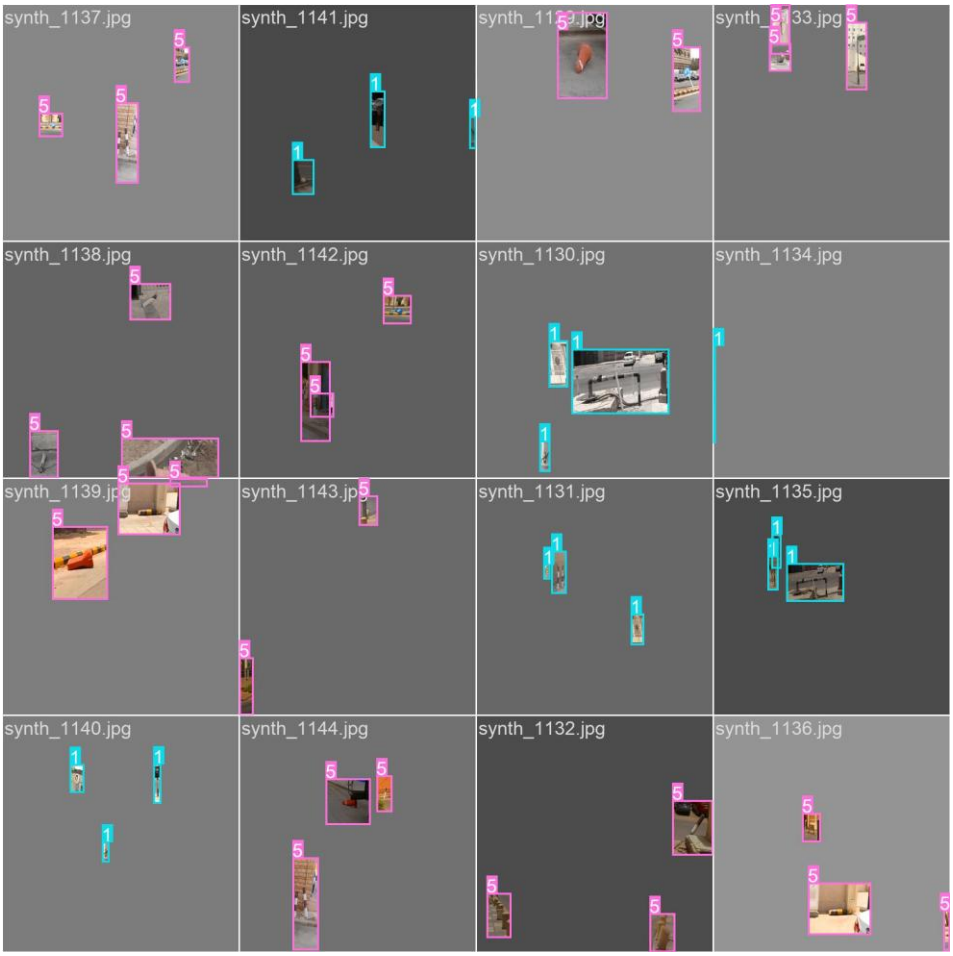

Figure 11 - Sample of synthetic images

7.1.1 Data Augmentation Techniques

Data augmentation techniques were systematically applied to enhance the robustness and generalization capabilities of the model. Below is the detailed configuration:

Augmentation	Value	Description
Mosaic	1.0	Combines four images into a single image to simulate complex backgrounds and varied object contexts.

MixUp	0.1	Blends two images to create intermediate representations, improving generalization.
HSV Adjustment	H:0.02, S:0.7, V:0.4	Adjusts hue, saturation, and brightness to simulate diverse lighting scenarios.
Rotation	±5°	Slight rotations to mimic camera angle variations.
Translation	±0.15	Horizontal and vertical shifts to simulate varied object positions.
Scale	±0.5	Varies object sizes to emulate different object distances.
Horizontal Flip	0.5	Flips images horizontally to diversify object orientation.
Perspective	0.0	Perspective distortion was disabled to maintain structural realism in urban scenes.

Table 3

These augmentations, particularly strong ones like Mosaic and MixUp, were selectively disabled for synthetic images to ensure that synthetic data maintained realistic integrity and avoided artificially inflating majority classes.

Figure 12 – Sample of augmented images

7.2 Performance Metrics

To effectively evaluate the performance of the proposed YOLOv11-based visual pollution detection framework [15], four key metrics were selected: mAP50, mAP50–95, Precision, and Recall. These metrics collectively capture the model's ability to accurately detect, localize, and classify objects under varying levels of strictness. While mAP metrics provide a holistic view of detection and localization performance, Precision and Recall offer insight into the balance between false positives and false negatives, both of which are critical for real-world environmental monitoring tasks.

Metric	Definition	Significance
mAP50	Mean Average Precision at 50% IoU threshold.	Measures detection accuracy with moderate localization strictness.
mAP50–95	Mean Average Precision averaged across IoU thresholds from 0.50 to 0.95.	Provides a comprehensive evaluation of both coarse and fine localization.
Precision	Ratio of true positive detections to all predicted positives.	Reflects how accurate the detections are (low false positives).
Recall	Ratio of true positive detections to all actual positives in the dataset.	Reflects how well the model captures all true objects (low false negatives).

Table 4

7.3 Experiments Setup

Describe the setup used and experimental configurations.
The experiments involved:

- **Hardware Utilization:** Training was conducted on a rented GPU — an NVIDIA H100 80GB — accessed through Vast.ai, a cloud computing marketplace known for offering affordable, flexible, and scalable GPU rental solutions. Vast.ai's flexible environment allowed quick setup, resource-efficient scaling, and cost-effective training compared to traditional cloud providers. The investment for GPU rental was approximately **$1.604 per hour**. The total investment was $65 in which we spent around 40 hours training and experimenting with different models.

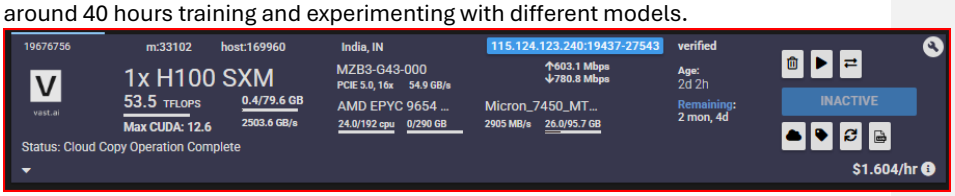

Figure 13

The specifications of the rented instance are detailed below:

Component	Specification
GPU	1x NVIDIA H100 SXM, 80GB VRAM

GPU Compute	53.5 TFLOPS, Max CUDA: 12.6
CPU	AMD EPYC 9654, 24 cores / 192 threads
CPU Memory	290 GB
Storage	Micron 7450 SSD, 5.7 TB
Network Speed	~603 Mbps download, ~780 Mbps upload
PCIe Bandwidth	5.4-9 GB/s

Table 5

Leveraging the immense memory capacity and powerful computational abilities of the H100 significantly accelerated training times and enabled the handling of larger batch sizes without memory bottlenecks. For instance, the best model took around 1.35 hours to run on the rented hardware while it took around 13 hours to train on our RTX3070 NVIDIA GPU.

- **Software Framework:** Implementation in PyTorch using YOLOv11 (Ultralytics framework).
- **Training Parameters:** Batch size of 100, training for 500 epochs with an early stopping patience of 30 epochs.
- **Hyperparameter Tuning:** Conducted using evolutionary algorithms for optimal configuration identification.

7.4 Result Comparative Analysis

This section presents a comprehensive comparative analysis between the proposed models and the baseline YOLOv11 implementation. Both subjective and objective evaluations are conducted to assess performance improvements. Subjective analysis is performed through qualitative visual comparisons, highlighting the detection accuracy and robustness of each model across a variety of test images. Objective analysis is carried out through quantitative comparisons based on key performance metrics, including mAP50, mAP50–95, Precision, and Recall. To ensure clarity and thoroughness, multiple figures and tables are provided to illustrate the results from different perspectives.

7.4.1 YOLOv11 Baseline (Vanilla)

The YOLOv11 Baseline model corresponds to the vanilla implementation of YOLOv11, trained without the application of regularization techniques or data balancing strategies. As a result, its performance is relatively lower compared to enhanced variants, particularly in handling class imbalance and complex visual pollution scenarios.

Model	*mAP50*	*mAP50–95*	*Precision*	*Recall*
YOLOv11 Baseline	0.318	0.160	0.471	0.188

Table 6

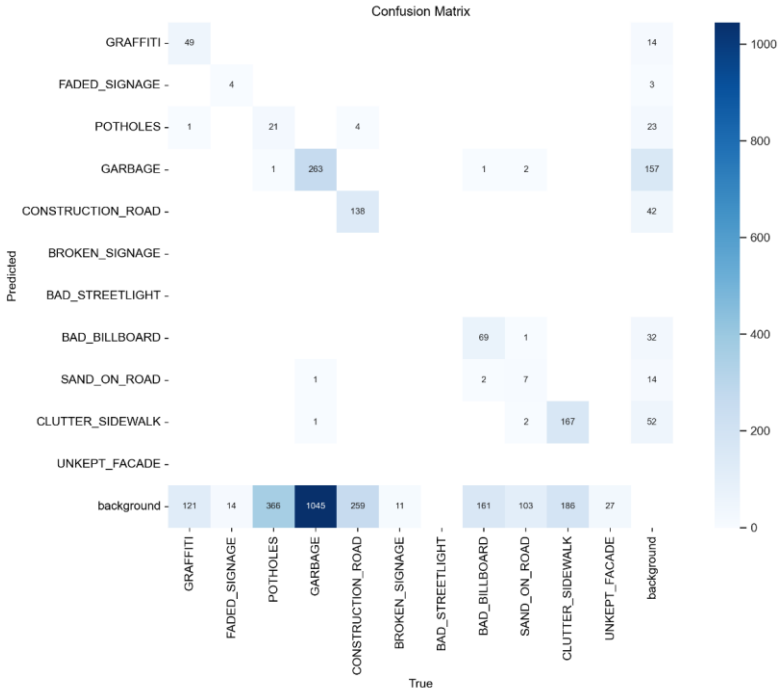
Confusion Matrix
Predicted
True
GRAFFITI
FADED_SIGNAGE
POTHOLES
GARBAGE
CONSTRUCTION_ROAD
BROKEN_SIGNAGE
BAD_STREETLIGHT
BAD_BILLBOARD
SAND_ON_ROAD
CLUTTER_SIDEWALK
UNKEPT_FACADE
background
1000
800
600
400
200
0

Figure 14

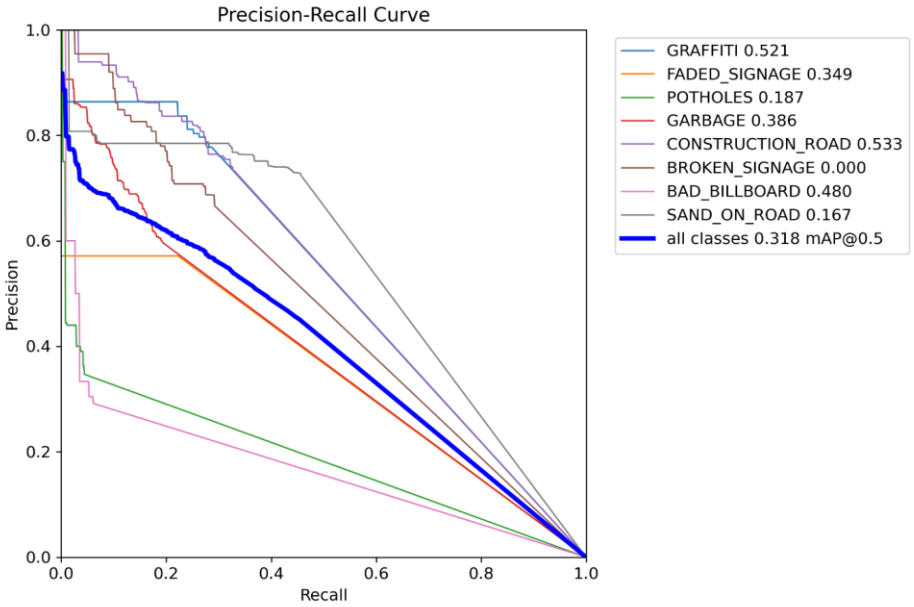
Precision-Recall Curve
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI 0.521
FADED_SIGNAGE 0.349
POTHOLES 0.187
GARBAGE 0.386
CONSTRUCTION_ROAD 0.533
BROKEN_SIGNAGE 0.000
BAD_BILLBOARD 0.480
SAND_ON_ROAD 0.167
all classes 0.318 mAP@0.5

Figure 15

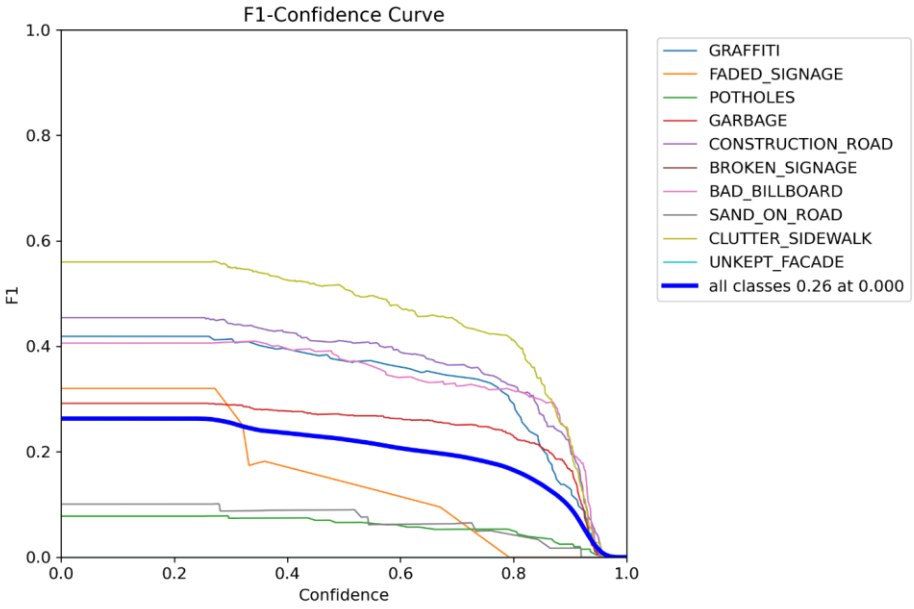
F1-Confidence Curve
F1
Confidence
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI
FADED_SIGNAGE
POTHOLES
GARBAGE
CONSTRUCTION_ROAD
BROKEN_SIGNAGE
BAD_BILLBOARD
SAND_ON_ROAD
CLUTTER_SIDEWALK
UNKEPT_FACADE
all classes 0.26 at 0.000

Figure 16

Figure 17 - Actual Labels

Figure 18 - Predicted Labels

7.4.2 YOLOv11 Baseline with Synthetic images

This model builds upon the YOLOv11 baseline by incorporating synthetic images to balance the dataset. While the underlying architecture remains unchanged, the inclusion of synthetic data addresses class imbalance issues, aiming to improve the model's ability to detect underrepresented visual pollution categories. As a result, slight performance gains are observed compared to the vanilla baseline.

The model demonstrates a positive impact of dataset balancing through synthetic images. Specifically, the model achieves an **mAP50 of 0.344**, compared to **0.318** for the baseline, reflecting an improvement in object detection accuracy. Similarly, improvements are

observed in mAP50–95 (**0.164** vs. **0.160**), Precision (**0.484** vs. **0.471**), and Recall (**0.221** vs. **0.188**). These results indicate that balancing the dataset with synthetic images enhances the model's ability to detect a broader range of visual pollution categories, even without modifying the model architecture or training strategy.

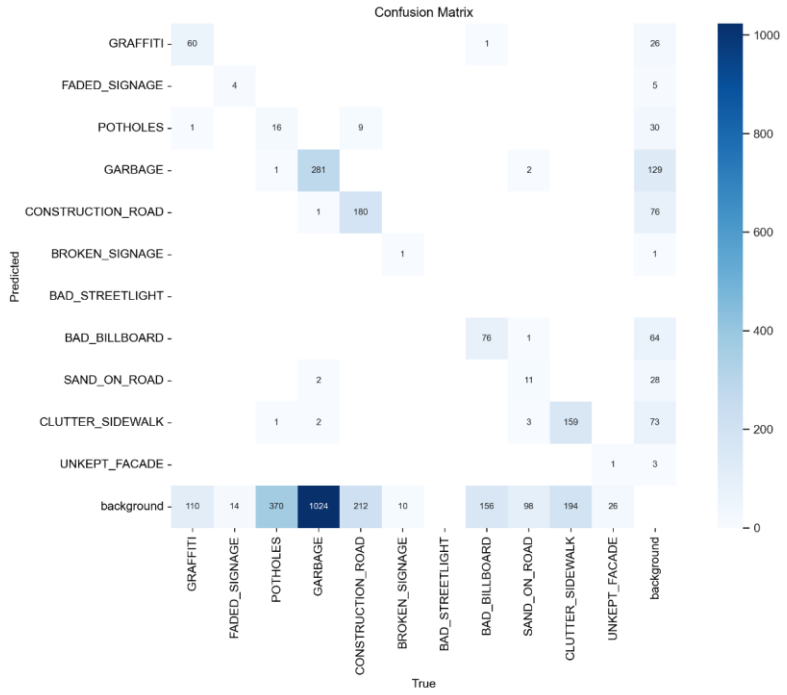

Figure 19

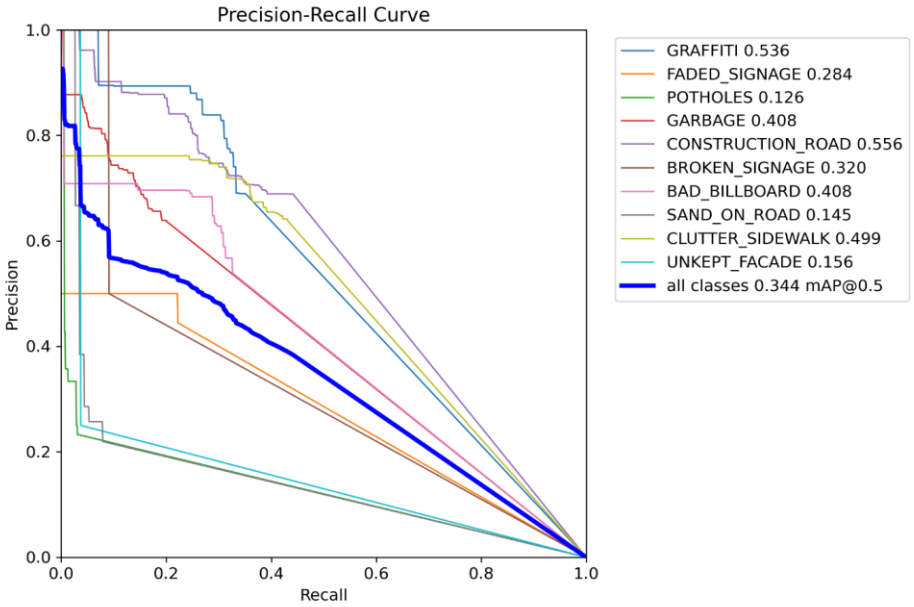
Precision-Recall Curve
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI 0.536
FADED_SIGNAGE 0.284
POTHOLES 0.126
GARBAGE 0.408
CONSTRUCTION_ROAD 0.556
BROKEN_SIGNAGE 0.320
BAD_BILLBOARD 0.408
SAND_ON_ROAD 0.145
CLUTTER_SIDEWALK 0.499
UNKEPT_FACADE 0.156
all classes 0.344 mAP@0.5

Figure 20

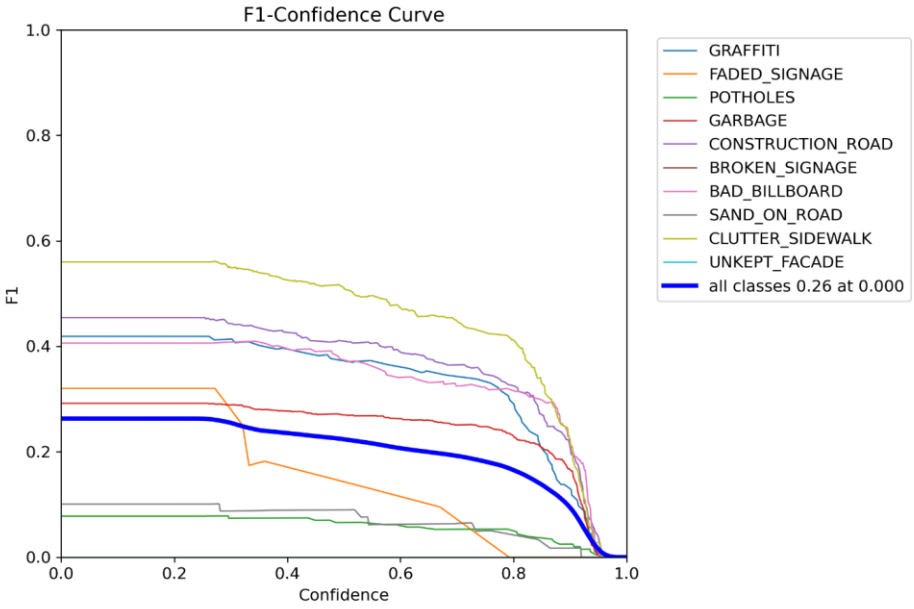
F1-Confidence Curve
F1
Confidence
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI
FADED_SIGNAGE
POTHOLES
GARBAGE
CONSTRUCTION_ROAD
BROKEN_SIGNAGE
BAD_BILLBOARD
SAND_ON_ROAD
CLUTTER_SIDEWALK
UNKEPT_FACADE
all classes 0.26 at 0.000

Figure 21

Figure 22 – Actual Labels

Figure 23 - Predicted Lables

7.4.3 YOLOv11 Baseline with Enhanced Augmentation Model

This model is based on the YOLOv11 baseline architecture but integrates an enhanced data augmentation pipeline during training. Unlike the vanilla model, this version applies advanced augmentation techniques to improve the model's robustness and generalization to diverse visual pollution scenarios. No changes were made to the model architecture; improvements are attributed solely to the augmentation strategy, resulting in noticeable gains across key performance metrics compared to the baseline.

A comparison between the model and the baseline highlights the benefits of enhanced data augmentation. The model achieves an **mAP50 of 0.432**, significantly higher than the

baseline value of **0.318**, indicating improved object detection accuracy. Additionally, the model records an increase in mAP50–95 (**0.240** vs. **0.160**), Precision (**0.553** vs. **0.471**), and Recall (**0.358** vs. **0.188**). These improvements demonstrate that stronger data augmentation strategies can effectively enhance the model's ability to generalize across a wider range of visual pollution instances, even without modifying the model's architecture.

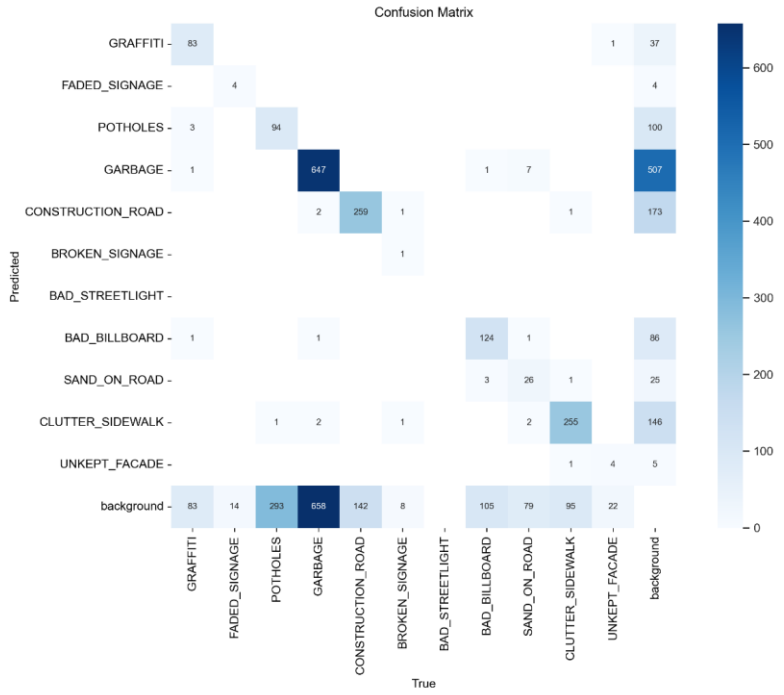

Figure 24

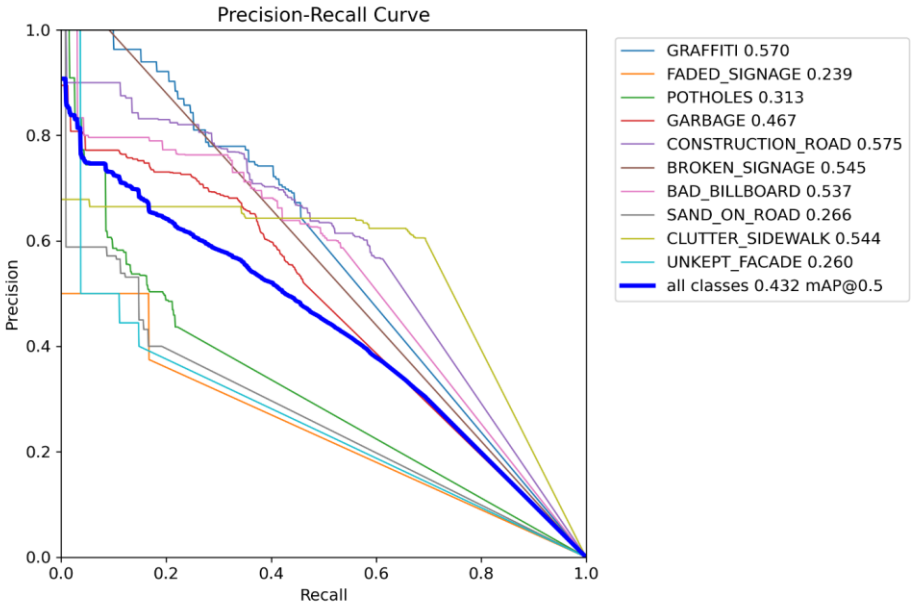
Precision-Recall Curve
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI 0.570
FADED_SIGNAGE 0.239
POTHOLES 0.313
GARBAGE 0.467
CONSTRUCTION_ROAD 0.575
BROKEN_SIGNAGE 0.545
BAD_BILLBOARD 0.537
SAND_ON_ROAD 0.266
CLUTTER_SIDEWALK 0.544
UNKEPT_FACADE 0.260
all classes 0.432 mAP@0.5

Figure 25

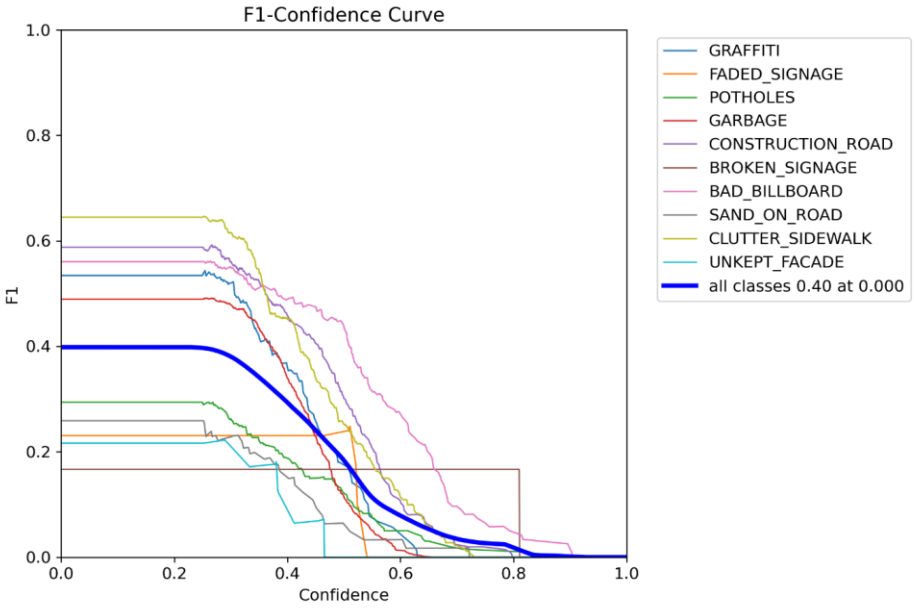
F1-Confidence Curve
F1
Confidence
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI
FADED_SIGNAGE
POTHOLES
GARBAGE
CONSTRUCTION_ROAD
BROKEN_SIGNAGE
BAD_BILLBOARD
SAND_ON_ROAD
CLUTTER_SIDEWALK
UNKEPT_FACADE
all classes 0.40 at 0.000

Figure 26

Figure 27 - Actual Labels

Figure 28 - Predicted Labels

7.4.4 YOLOv11 Baseline with Enhanced Augmentation Model + Synthetic Images

This model builds upon the previous version by additionally incorporating synthetic images to balance the dataset. While it maintains the enhanced data augmentation pipeline, the integration of synthetic data further addresses class imbalance, aiming to improve detection performance for underrepresented visual pollution categories. This combination of strategies enhances both the generalization capability and robustness of the model compared to the original improved variant.

A comparison between this model and base highlights the benefits of combining enhanced data augmentation with synthetic data balancing. The model achieves an **mAP50 of 0.443**, significantly higher than the baseline value of **0.318**, indicating improved object detection accuracy. Additionally, the model records an increase in **mAP50–95 (0.211 vs. 0.160)**, **Precision (0.600 vs. 0.471)**, and **Recall (0.293 vs. 0.188)**. These improvements demonstrate that the integration of synthetic images with advanced augmentation strategies can effectively enhance the model's ability to generalize across a broader range of visual pollution instances, even without modifying the model's underlying architecture.

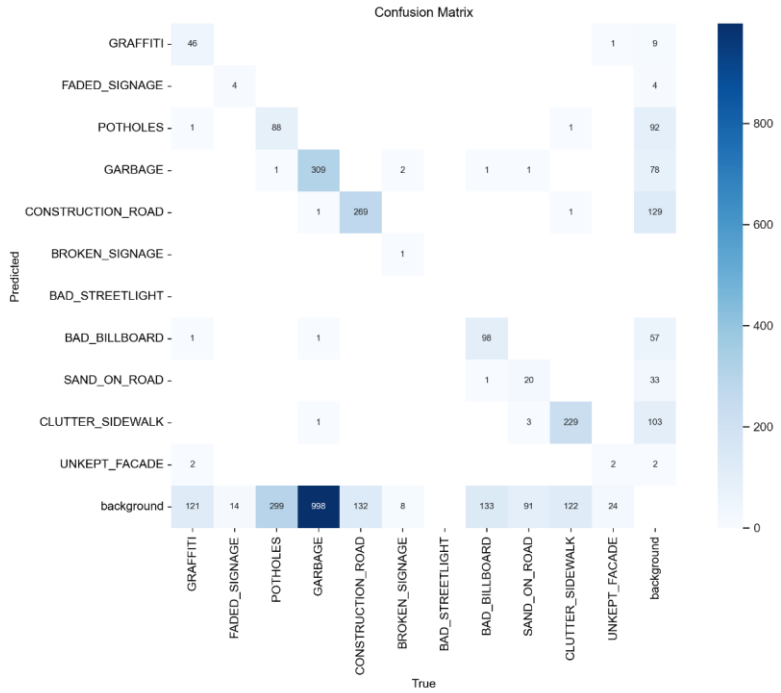

Figure 29

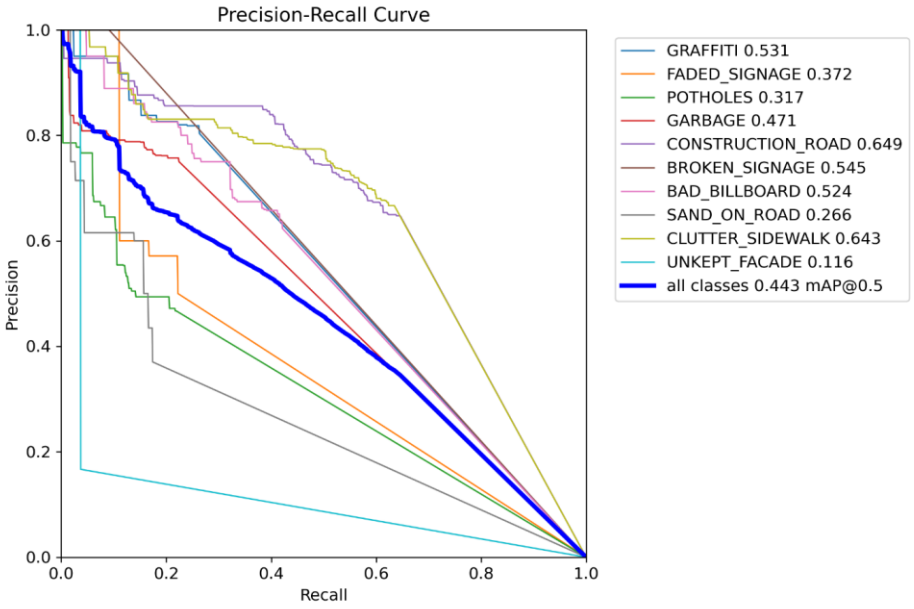
Precision-Recall Curve
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI 0.531
FADED_SIGNAGE 0.372
POTHOLES 0.317
GARBAGE 0.471
CONSTRUCTION_ROAD 0.649
BROKEN_SIGNAGE 0.545
BAD_BILLBOARD 0.524
SAND_ON_ROAD 0.266
CLUTTER_SIDEWALK 0.643
UNKEPT_FACADE 0.116
all classes 0.443 mAP@0.5

Figure 30

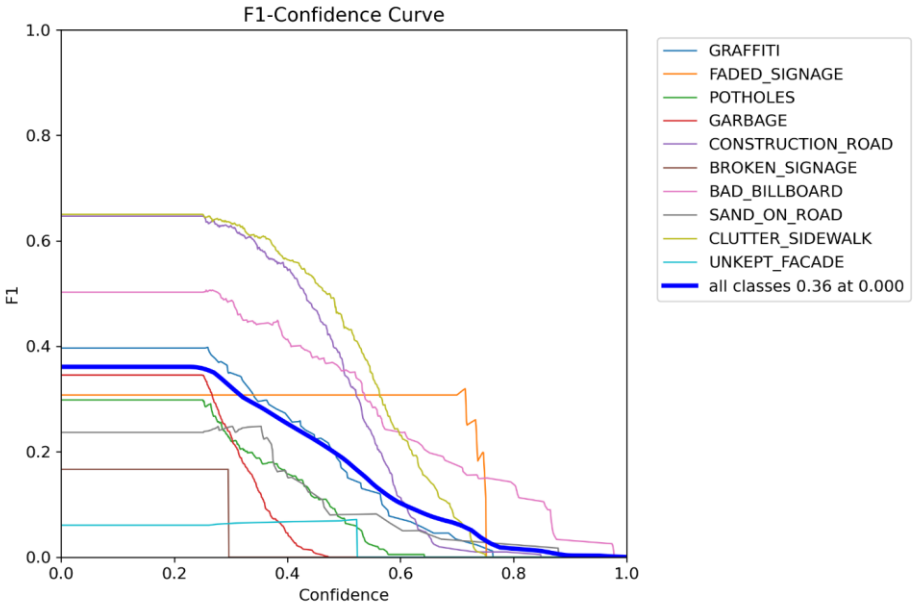
F1-Confidence Curve
F1
Confidence
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI
FADED_SIGNAGE
POTHOLES
GARBAGE
CONSTRUCTION_ROAD
BROKEN_SIGNAGE
BAD_BILLBOARD
SAND_ON_ROAD
CLUTTER_SIDEWALK
UNKEPT_FACADE
all classes 0.36 at 0.000

Figure 31

Figure 32 - Actual Labels

Figure 33 - Predicted Labels

7.4.5 YOLOv11 Baseline with Enhanced Augmentation Model + Synthetic Images + Weighted Loss

The model builds upon the previous variant by additionally incorporating a weighted loss function based on class weights. While maintaining the enhanced data augmentation and synthetic data balancing strategies, the introduction of class-weighted loss further addresses class imbalance by penalizing misclassifications of minority classes more heavily during training. This adjustment aims to improve the model's detection accuracy

across underrepresented visual pollution categories, resulting in enhanced overall performance.

A comparison between the model and the baseline highlights the benefits of combining enhanced data augmentation, synthetic data balancing, and a class-weighted loss function. The model achieves an **mAP50 of 0.513**, significantly higher than the baseline value of **0.318**, indicating improved object detection accuracy. Additionally, the model records an increase in **mAP50–95 (0.260 vs. 0.160)**, **Precision (0.650 vs. 0.471)**, and **Recall (0.397 vs. 0.188)**. These improvements demonstrate that integrating class weighting alongside data-centric strategies can further enhance the model's ability to generalize and accurately detect diverse visual pollution instances.

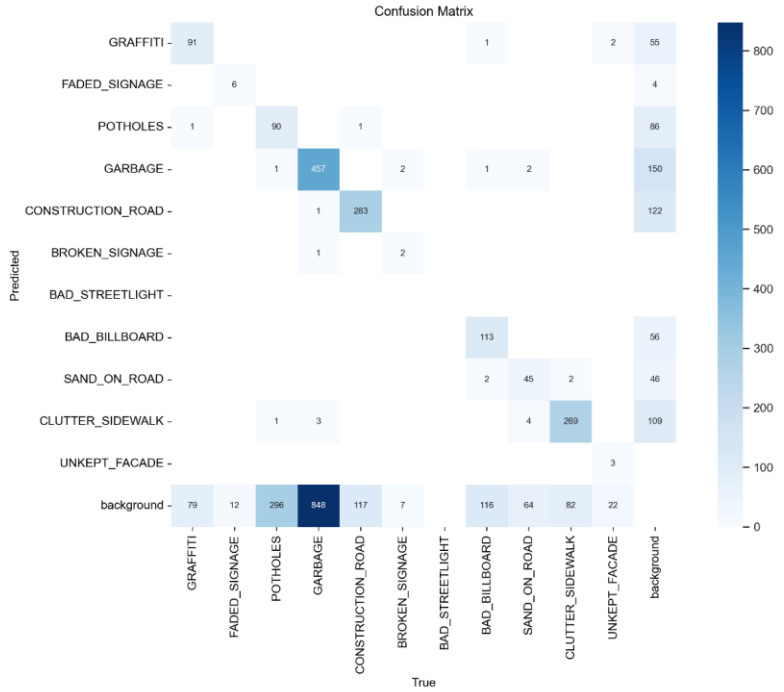

Figure 34

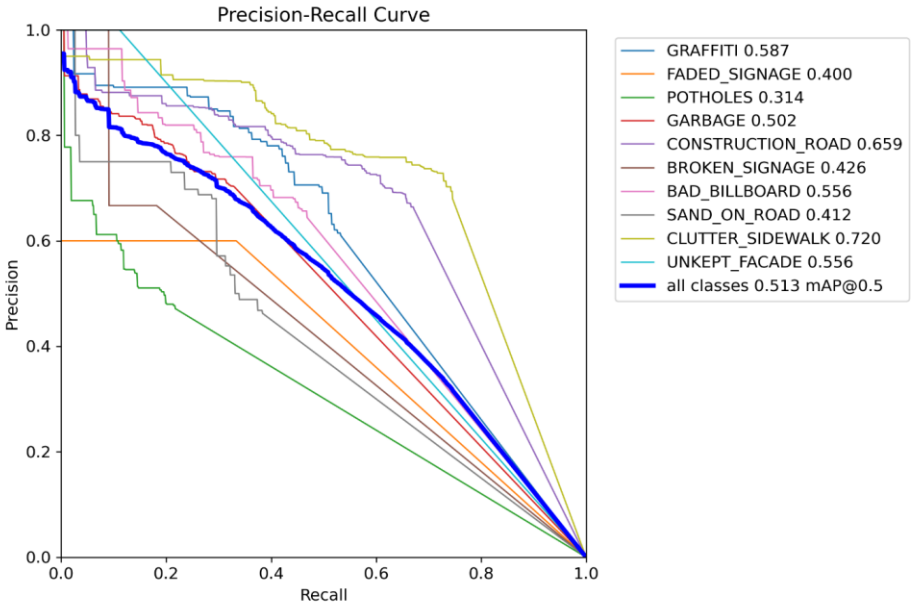
Precision-Recall Curve
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI 0.587
FADED_SIGNAGE 0.400
POTHOLES 0.314
GARBAGE 0.502
CONSTRUCTION_ROAD 0.659
BROKEN_SIGNAGE 0.426
BAD_BILLBOARD 0.556
SAND_ON_ROAD 0.412
CLUTTER_SIDEWALK 0.720
UNKEPT_FACADE 0.556
all classes 0.513 mAP@0.5

Figure 35

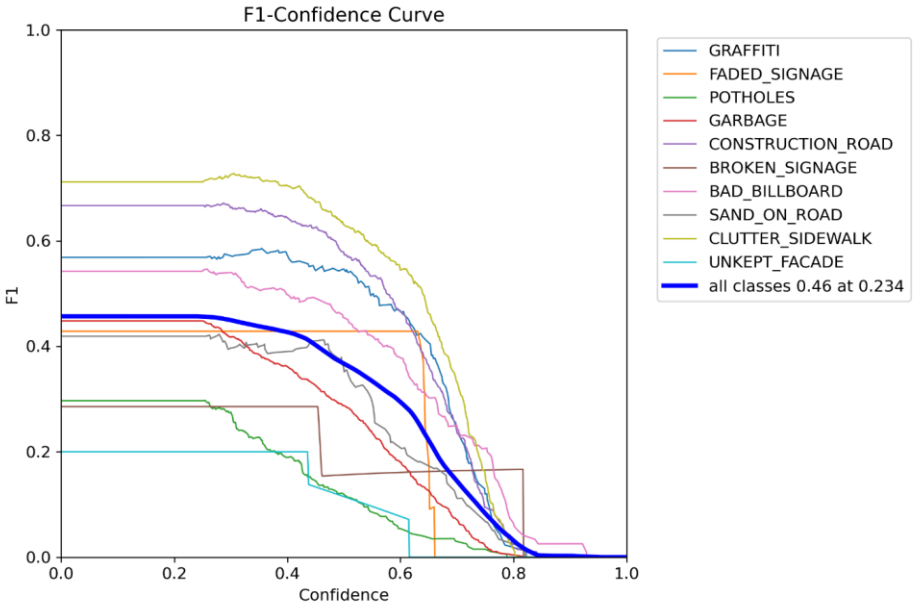
F1-Confidence Curve
F1
Confidence
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
GRAFFITI
FADED_SIGNAGE
POTHOLES
GARBAGE
CONSTRUCTION_ROAD
BROKEN_SIGNAGE
BAD_BILLBOARD
SAND_ON_ROAD
CLUTTER_SIDEWALK
UNKEPT_FACADE
all classes 0.46 at 0.234

Figure 36

Figure 37 - Actual Labels

Figure 38 - Predicted Labels

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	1181	3027	0.471	0.188	0.318	0.160
GRAFFITI	97	171	0.778	0.287	0.521	0.279
FADED_SIGNAGE	12	18	0.571	0.222	0.349	0.198
POTHOLES	169	388	0.347	0.0438	0.187	0.0785
GARBAGE	581	1310	0.597	0.193	0.386	0.177
CONSTRUCTION_ROAD	157	401	0.733	0.329	0.533	0.236
BROKEN_SIGNAGE	9	11	0.000	0.000	0.000	0.000
BAD_BILLBOARD	165	233	0.667	0.292	0.480	0.275
SAND_ON_ROAD	74	115	0.292	0.0609	0.167	0.0654

CLUTTER_SIDEWALK	172	353	0.725	0.456	0.555	0.291
UNKEPT_FACADE	14	27	0.000	0.000	0.000	0.000

Table 7 - YOLOv11 Baseline (Vanilla) test performance

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	1181	3027	0.484	0.221	0.344	0.164
GRAFFITI	97	171	0.690	0.351	0.536	0.259
FADED_SIGNAGE	12	18	0.444	0.222	0.284	0.167
POTHOLES	169	388	0.232	0.0335	0.126	0.0619
GARBAGE	581	1310	0.637	0.201	0.408	0.178
CONSTRUCTION_ROAD	157	401	0.689	0.441	0.556	0.212
BROKEN_SIGNAGE	9	11	0.500	0.0909	0.320	0.160
BAD_BILLBOARD	165	233	0.539	0.326	0.408	0.218
SAND_ON_ROAD	74	115	0.220	0.0783	0.145	0.0681
CLUTTER_SIDEWALK	172	353	0.639	0.431	0.499	0.240
UNKEPT_FACADE	14	27	0.250	0.037	0.156	0.0779

Table 8-YOLOv11 Baseline with Synthetic images Test Performance

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	1181	3027	0.553	0.358	0.432	0.240
GRAFFITI	97	171	0.645	0.456	0.570	0.300
FADED_SIGNAGE	12	18	0.375	0.167	0.239	0.168
POTHOLES	169	388	0.437	0.222	0.313	0.142
GARBAGE	581	1310	0.520	0.462	0.467	0.198
CONSTRUCTION_ROAD	157	401	0.564	0.613	0.575	0.247
BROKEN_SIGNAGE	9	11	1.000	0.0909	0.545	0.545
BAD_BILLBOARD	165	233	0.587	0.536	0.537	0.272
SAND_ON_ROAD	74	115	0.400	0.191	0.266	0.101
CLUTTER_SIDEWALK	172	353	0.602	0.694	0.544	0.270
UNKEPT_FACADE	14	27	0.400	0.148	0.260	0.160

Table 9-YOLOv11 Baseline with Enhanced Augmentation Model Test Performance

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	1181	3027	0.600	0.293	0.443	0.211
GRAFFITI	97	171	0.804	0.263	0.531	0.268
FADED_SIGNAGE	12	18	0.500	0.222	0.372	0.198
POTHOLES	169	388	0.467	0.219	0.317	0.126
GARBAGE	581	1310	0.750	0.224	0.471	0.210
CONSTRUCTION_ROAD	157	401	0.647	0.646	0.649	0.266
BROKEN_SIGNAGE	9	11	1.000	0.0909	0.545	0.218
BAD_BILLBOARD	165	233	0.624	0.421	0.524	0.293
SAND_ON_ROAD	74	115	0.370	0.174	0.266	0.124

CLUTTER_SIDEWALK	172	353	0.667	0.635	0.643	0.333
UNKEPT_FACADE	14	27	0.167	0.037	0.116	0.0694

Table 10 - 7.4.4 YOLOv11 Baseline with Enhanced Augmentation Model + Synthetic Images Test Performance

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	1181	3027	0.650	0.397	0.513	0.260
GRAFFITI	97	171	0.611	0.532	0.587	0.314
FADED_SIGNAGE	12	18	0.600	0.333	0.400	0.209
POTHOLES	169	388	0.472	0.216	0.314	0.133
GARBAGE	581	1310	0.703	0.329	0.502	0.219
CONSTRUCTION_ROAD	157	401	0.663	0.671	0.659	0.277
BROKEN_SIGNAGE	9	11	0.667	0.182	0.426	0.298
BAD_BILLBOARD	165	233	0.645	0.468	0.556	0.311
SAND_ON_ROAD	74	115	0.463	0.383	0.412	0.186
CLUTTER_SIDEWALK	172	353	0.681	0.745	0.720	0.356
UNKEPT_FACADE	14	27	1.000	0.111	0.556	0.299

Table 11 - YOLOv11 Baseline with Enhanced Augmentation Model + Synthetic Images + Weighted Loss Test Performance

7.5 Ablation Study

The ablation study demonstrates a clear progression in performance as incremental improvements are applied to the baseline model. Incorporating synthetic data balancing led to moderate gains across all metrics compared to the YOLOv11 Baseline. Introducing enhanced data augmentation further improved detection accuracy and generalization. Combining both synthetic images and augmentation resulted in additional performance boosts. Finally, applying a class-weighted loss function achieved the highest scores across all evaluation metrics, highlighting the cumulative benefits of addressing class imbalance through both data-level and loss-level strategies. These results validate the effectiveness of each proposed enhancement in progressively improving the model's capability to detect diverse visual pollution categories.

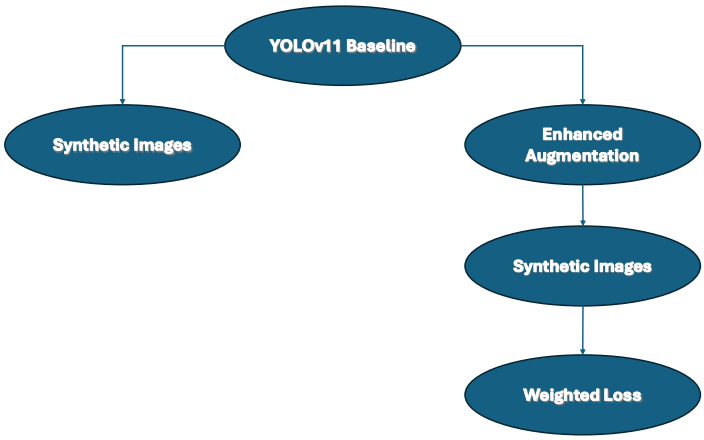

Figure 39 - Model design hierarchy

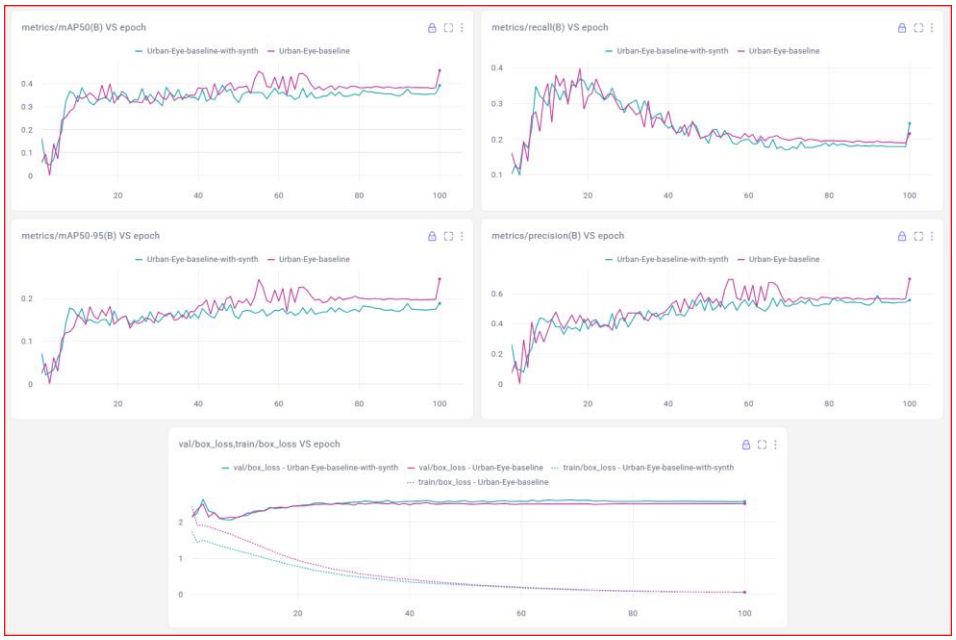

Figure 40

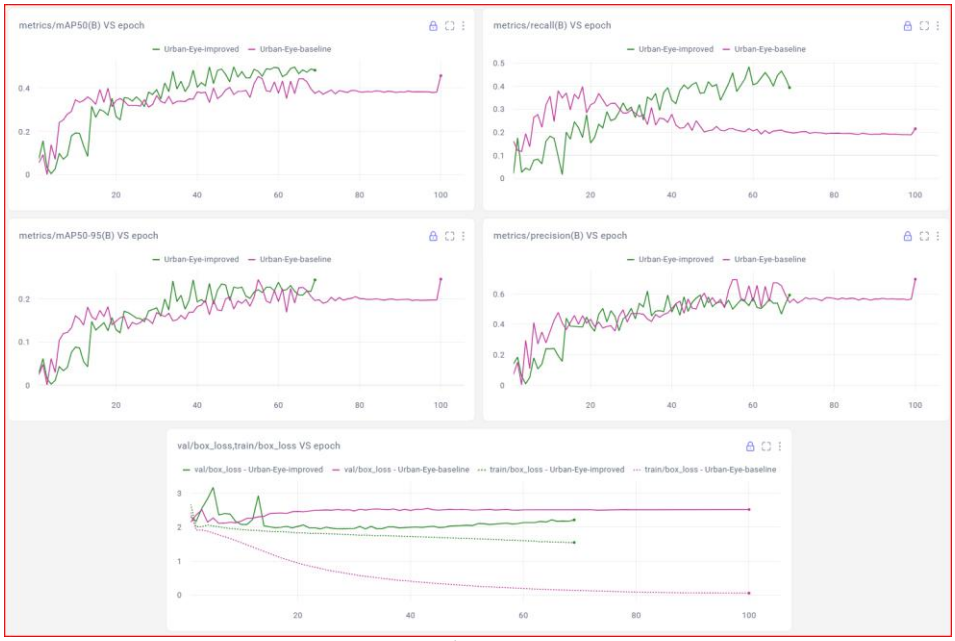
metrics/mAP50(B) VS epoch
Urban-Eye-improved
Urban-Eye-baseline
metrics/recall(B) VS epoch
Urban-Eye-improved
Urban-Eye-baseline
metrics/mAP50-95(B) VS epoch
Urban-Eye-improved
Urban-Eye-baseline
metrics/precision(B) VS epoch
Urban-Eye-improved
Urban-Eye-baseline
val/box_loss,train/box_loss VS epoch
val/box_loss - Urban-Eye-improved
val/box_loss - Urban-Eye-baseline
train/box_loss - Urban-Eye-improved
train/box_loss - Urban-Eye-baseline

Figure 41

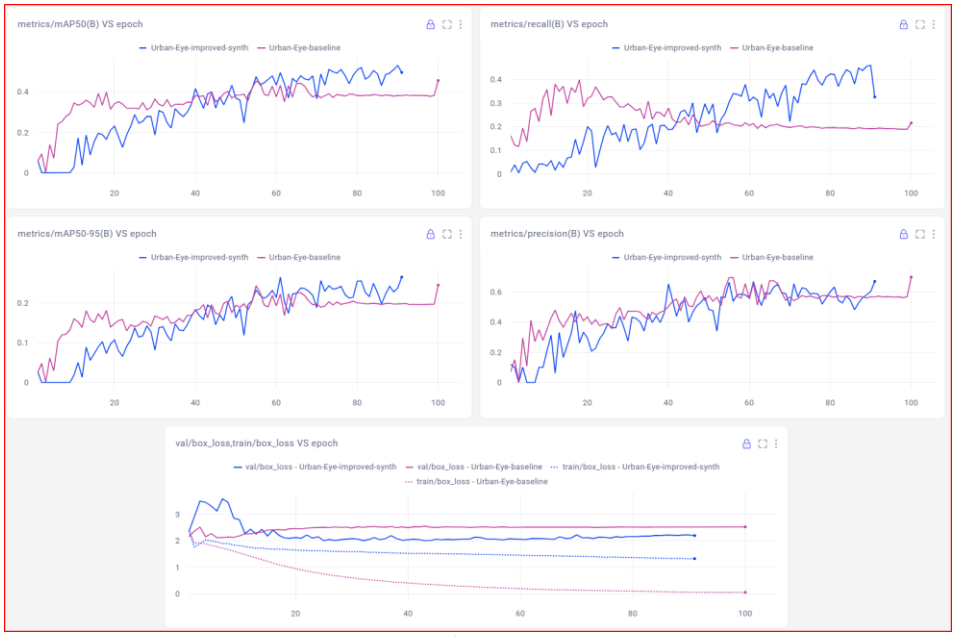
metrics/mAP50(B) VS epoch
Urban-Eye-improved-synth
Urban-Eye-baseline
metrics/recall(B) VS epoch
Urban-Eye-improved-synth
Urban-Eye-baseline
metrics/mAP50-95(B) VS epoch
Urban-Eye-improved-synth
Urban-Eye-baseline
metrics/precision(B) VS epoch
Urban-Eye-improved-synth
Urban-Eye-baseline
val/box_loss,train/box_loss VS epoch
val/box_loss - Urban-Eye-improved-synth
val/box_loss - Urban-Eye-baseline
train/box_loss - Urban-Eye-improved-synth
train/box_loss - Urban-Eye-baseline

Figure 42

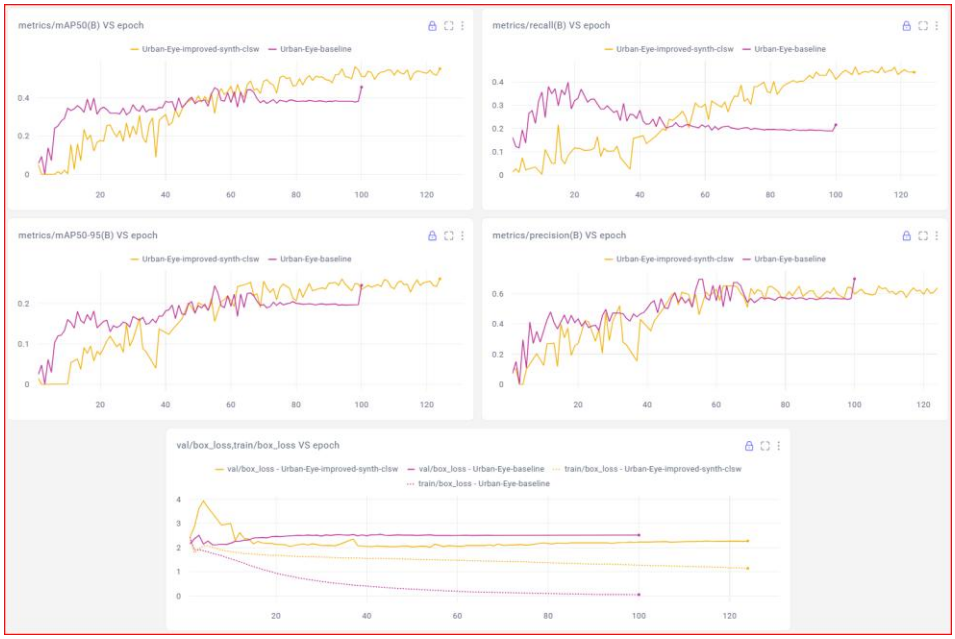

Figure 43

Model	**mAP50**	**mAP50–95**	**Precision**	**Recall**
YOLOv11 Baseline + Enhanced Augmentation + Synthetic Images + Weighted Loss	0.513	0.260	0.650	0.397
YOLOv11 Baseline + Enhanced Augmentation + Synthetic Images	0.443	0.211	0.600	0.293
YOLOv11 Baseline + Enhanced Augmentation	0.432	0.240	0.553	0.358
YOLOv11 Baseline + Synthetic Images	0.344	0.164	0.484	0.221
YOLOv11 Baseline	0.318	0.160	0.471	0.188

Table 12 – Models evaluation

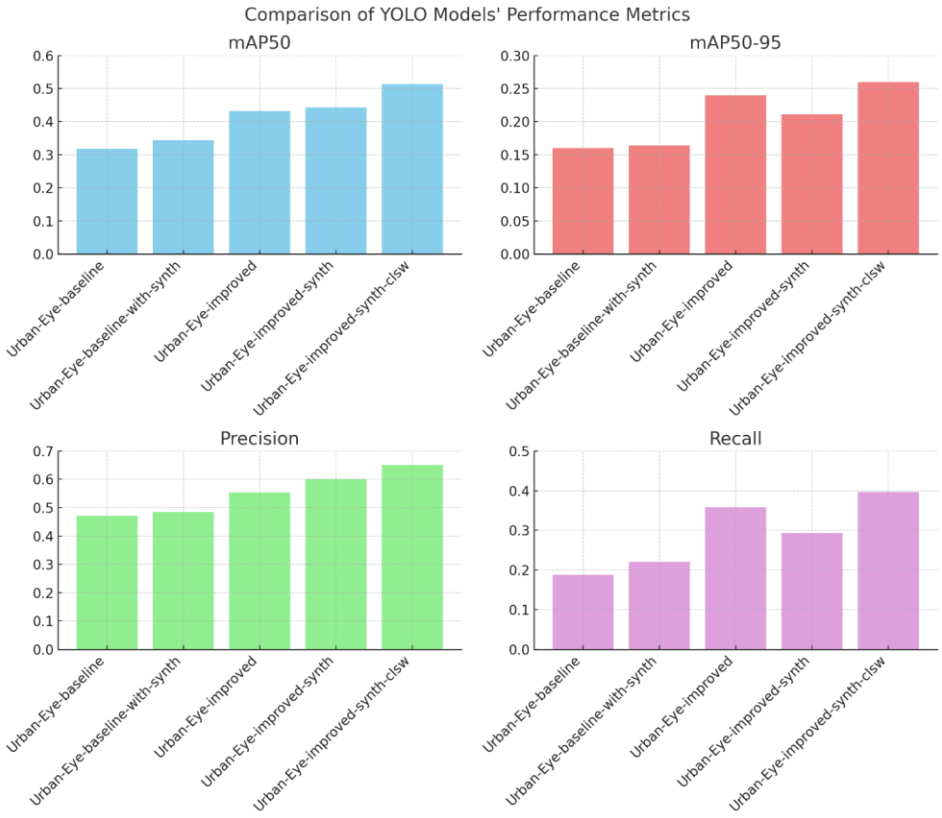

Figure 44

8. Extended Contribution

This project introduces an enhanced YOLOv11-based framework for visual pollution detection, integrating synthetic data generation, class-weighted loss functions, and advanced data augmentation techniques. These data-centric enhancements effectively address class imbalance and improve model robustness without modifying the base architecture, leading to notable gains in detection performance across a wide range of visual pollution categories.

The proposed framework offers a scalable and practical solution for urban monitoring applications. By demonstrating the importance of dataset engineering and training strategies, this work establishes a foundation for future advancements in fighting visual pollution in cities.

9. Conclusion and Future Work

This project presented an enhanced YOLOv11-based framework for visual pollution detection, incorporating synthetic data generation, advanced data augmentation, and a class-weighted loss function. The proposed improvements effectively addressed dataset imbalance, resulting in significant performance gains across all major evaluation metrics compared to the baseline model. Through extensive comparative and ablation studies, the contributions of each enhancement were systematically validated, demonstrating the framework's potential for real-world urban monitoring applications.

For future work, expanding the dataset by collecting additional images from different cities and regions across Saudi Arabia is recommended to improve the model's generalization to diverse urban environments. Furthermore, enriching the dataset with samples captured under varying conditions, such as different times of day (day and night) and weather scenarios (rain, fog, dust), would help develop a more inclusive and resilient detection system. Ensuring labeling the classes as tights as possible to eliminate false positive backgrounds. Exploring advanced domain adaptation techniques and multi-modal data integration could further strengthen the framework's robustness and applicability in broader smart city and environmental monitoring contexts.

10. References

Commented [RA3]: Add references (min 15)

Commented [RA4R3]: Add YOLO paper and website

Commented [RA5R3]: Also update Github

[1] DFRobot, "Top 6 Most Favored Object Detection Models in 2024 | YOLOv10, EfficientDet, DETR, etc," [Online]. Available: https://www.dfrobot.com/blog-13914.html. [Accessed 18 04 2025].

[2] A. Alzu'bi, F. M. Alkhateeb and A. Kayed, "A deep active learning approach for visual pollution detection using public road images," *Mathematics,* vol. 11, no. 1, p. 186, 2023.

[3] M. F. S. Titu, A. A. M. Chowdhury, S. M. R. Haque and R. Khan, "Deep-Learning-Based Real-Time Visual Pollution Detection in Urban and Textile Environments," *Sci,* vol. 6, p. 5, 2024.

[4] K. Elbaz, W. M. Shaban, A. Zhou and S. Shen, "Real time image-based air quality forecasts using a 3D-CNN approach with an attention mechanism," *Chemosphere,* vol. 333, p. 138867, 2023.

[5] M. AlElaiwi, M. A. Al-antari, H. F. Ahmad, A. Azhar, B. Almarri and J. Hussain, "VPP: Visual Pollution Prediction Framework Based on a Deep Active Learning Approach Using Public Road Images," *Mathematics,* vol. 11, no. 1, p. 186, 2023.

[6] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv, 2018.

[7] A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv, 2020.

[8] C.-Y. Wang, A. Bochkovskiy and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," 2022.

[9] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data,* vol. 1, p. 60, 2019.

[10] Y. Cui, M. Jia, T.-Y. Lin, Y. Song and S. Belongie, "Class-Balanced Loss Based on Effective Number of Samples for Deep Long-Tailed Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[11] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements".

[12] S. N. Rao, "YOLOv11 Architecture Explained: Next-Level Object Detection with Enhanced Speed and Accuracy," 22 October 2024. [Online]. Available: https://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71. [Accessed 18 04 2025].

[13] "PyTorch," [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html. [Accessed 18 April 2025].

[14] G. Jocher, "optimizer='auto'," [Online]. Available: https://github.com/ultralytics/ultralytics/issues/9182#issuecomment-2012548155. [Accessed 18 April 2025].

[15] "Performance Metrics Deep Dive," [Online]. Available: https://docs.ultralytics.com/guides/yolo-performance-metrics/#speed-metrics. [Accessed 18 April 2025].

[16] M. Yasin, G. Jocherm, M. R. Munawar, P. Derrenger and F. Mattioli, "Ultralytics YOLO11 Overview," Ultralytics, [Online]. Available: https://docs.ultralytics.com/models/yolo11/. [Accessed 18 April 2025].