Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Detecting Illegal Dumping on Roads of Bangladesh: A Machine Learning Approach with a Novel Dataset

Rohul Amin
University of Dhaka
Syed Naimul Islam
University of Dhaka
Sumaiya Tabassum
University of Dhaka
Md. Al Habib
University of Dhaka
Sakib Hossain
University of Dhaka
Mehedi Hasan Sakib
University of Dhaka
Muhammad Ibrahim
ibrahim313@du.ac.bd

University of Dhaka

Research Article

Keywords: illegal dumping detection, image classification, YOLOv8x-cls, Random Forest, SVM, ANN, data augmentation, critical negative images, urban waste monitoring

Posted Date: November 24th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11812457/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Detecting Illegal Dumping on Roads of Bangladesh: A Machine Learning Approach with a Novel Dataset

Syed Naimul Islam, Sumaiya Tabassum, Md. Rohul Amin,
Md. Al Habib, Sakib Hossain, Mehedi Hasan Sakib
Muhammad Ibrahim*
Department of Computer Science and Engineering
University of Dhaka, Dhaka, Bangladesh
Email: syednaimul-2021711213@cs.du.ac.bd, sumaiya-2021611205@cs.du.ac.bd
ruhul-2021811203@cs.du.ac.bd, al-2021011193@cs.du.ac.bd
sakib-2021311181@cs.du.ac.bd, mehedihasan-2021511189@cs.du.ac.bd
ibrahim313@du.ac.bd*
*Corresponding author

***Abstract*—Roadside dumping poses a municipal monitoring challenge, particularly when non-dumping scenes contain litter, shadows, or rough surfaces. This study investigates binary image classification as a screening approach to support human inspection, assigning images to Dump/Positive or Clean/Negative. A locally collected dataset contained 326 original Positive, 279 regular Negative, and 199 Critical Negative images. Critical Negatives represented visually challenging non-dumping scenes. Each group was partitioned using an approximately 70/20/10 split before five augmented variants were generated per original, with all variants retained within their source split. Merging regular and Critical Negative images yielded 4,824 images: 3,372 for training, 954 for validation, and 498 for testing. An ImageNet-pretrained YOLOv8x-cls model was compared with Random Forest, support vector machine (SVM), and an artificial neural network (ANN) implemented as a multilayer perceptron. YOLOv8x-cls achieved the highest test accuracy at 96.79%, compared with 87.35%, 81.93%, and 75.10% for Random Forest, SVM, and ANN, respectively. It produced four false-positive and twelve false-negative predictions. These results motivate further evaluation of the proposed approach for roadside-image screening. However, the test images derive from 83 originals, and differences in pretraining, resolution, augmentation, and accepted training images prevent attribution of the performance gap to architecture alone. Independent testing is needed before deployment.**

***Index Terms*—illegal dumping detection, image classification, YOLOv8x-cls, Random Forest, SVM, ANN, data augmentation, critical negative images, urban waste monitoring**

I. Introduction

Roadside dumping creates a monitoring challenge for municipal waste-management services. Previous research has examined manual identification of roadside dumps and the use of image-based monitoring to support waste-management services [2], [6]. A relevant computational task is to screen roadside images for possible dumping so that human inspectors can prioritize follow-up.

This study addresses this task through supervised binary image classification. Positive/Dump denotes scenes annotated as dumping, whereas Negative/Clean denotes scenes annotated as non-dumping, including some scenes with minor scattered litter. These labels describe visual categories rather than independently establishing legal status. Non-dumping scenes may contain leaves, shadows, rough surfaces, or construction materials that resemble dumping. The dataset therefore includes a Critical Negative group selected to represent challenging non-dumping examples.

The study investigates whether an image classifier can distinguish these scene categories under the reported experimental conditions. The approach combines locally collected images, augmentation within predefined splits, and a comparison of four classifier configurations using overall accuracy and class-specific errors. The intended application is to flag potentially relevant images for human inspection; an integrated municipal alerting workflow was not implemented or evaluated. The main contributions are:

- a description of the dataset acquisition, class definitions, partitioning, and augmentation procedures;
- an empirical comparison of an ImageNet-pretrained YOLOv8x-cls model with Random Forest, SVM, and ANN baselines under documented settings; and
- an analysis of class-specific errors and the limitations affecting model comparison and generalization.

II. Related Work

Prior work relevant to this study spans four areas: illegal dump detection with deep learning, YOLO-based waste detection, IoT-integrated monitoring systems, and scene-level classification for urban waste.

A. Illegal Dump Detection Using Deep Learning

Padubidri et al. [1] investigated illegal dump-site detection using high-resolution aerial imagery and deep learning. Their aerial detection setting differs from the street-level image classification considered here. Shahab and Anjum [2] proposed a multipath convolutional neural network (mp-CNN) for illegal dump detection and localization on roads and roadsides in India. Using weak supervision on a locally constructed dataset, their model achieved 98.33% classification accuracy and an F1-score of 98.34%.

B. YOLO-Based Waste Dump Detection

Wibowo et al. [3] conducted a comparative study of five YOLO architectures—YOLOv8, YOLOv9, YOLOv10, YOLOv11, and YOLOv12—for detecting unlawful waste dumps across diverse outdoor environments. These models perform object detection, whereas the present study evaluates image-level classification. TrashNet-based studies [4], [5] address classification of individual waste objects; the present task concerns whole roadside scenes containing multiple objects and background structures.

C. IoT and Street-Level Monitoring Systems

Kumar et al. [6] proposed an integrated IoT and computer vision system for monitoring garbage vulnerable points (GVPs) in Sangareddy, India, using a street-level camera combined with YOLO object detection. Among the evaluated models, YOLO11m achieved 92.39% accuracy and a mAP@50 of 0.91. Their integrated monitoring system provides a related example of street-level waste monitoring, but differs from the image-classification component evaluated in the present study.

D. Scene-Level Classification and UAV-Based Detection

Izquierdo-Horna et al. [7] assessed the feasibility of UAV imagery combined with YOLOv7 object detection to identify improperly discarded plastic bags in Metropolitan Lima, Peru, processing 370 high-resolution aerial images. Their reported mAP@50 of 21.3% and low recall illustrate the challenges of waste detection under the study's data and imaging conditions. The proposed approach adopts scene-level binary classification to identify whether an image contains dumped waste; this design choice is not a direct performance comparison with their object detector. TACO [8] provides annotations of individual litter objects in varied real-world scenes, supporting object-level analysis rather than whole-image dump/clean classification. The original YOLO and YOLOv4 papers [9], [10] describe general-purpose object detection architectures and provide methodological background.

The reviewed studies address related tasks across aerial, object-level, and street-level settings. Their reported results do not directly establish performance on the Dhaka dataset considered here. This study therefore evaluates scene-level binary classification on locally collected images, including challenging non-dumping examples. Results across these studies are not directly comparable because their datasets and evaluation tasks differ.

III. Dataset Construction

A. Source Classes and Motivation

The dataset comprised three source groups before binary merging. Positive images were annotated as dumping, and regular Negative images were annotated as non-dumping. Critical Negative images were selected for visual characteristics that may resemble dumping, including scattered litter, shadows, and rough surfaces. Their inclusion was intended to represent challenging non-dumping conditions; its effect on false-positive predictions was not isolated experimentally.

Original images were partitioned before offline augmentation. Every generated variant remained in the same split as its source image, preventing variants of that original from being assigned to different splits. This procedure does not assess duplicate originals or repeated locations across splits.

B. Image Acquisition and Geographical Scope

Images were collected primarily in Dhaka, Bangladesh, including Chankharpul, Lalbagh, Azimpur, Chawkbazar, Anandabazar, and Bokshibazar in Old Dhaka; roads surrounding the University of Dhaka; Shahbag; and Uttara. Images were captured using multiple smartphone cameras, including Poco and iQOO devices and an iPhone 16.

Images were collected under daylight and low-light conditions, including sunny, overcast, and wet-road scenes. Images received image-level labels without bounding-box annotations. Negative/Clean included Critical Negative scenes annotated as non-dumping despite potentially confusing content, including minor scattered litter. Labels assigned during dataset organization were reviewed by other team members.

C. Original-Image Partitioning

The original Positive and regular Negative image pools contained 326 and 279 images, respectively. Images were enumerated in sorted order and shuffled separately within each source group using Python's random generator initialized with seed 42. Training and validation received the integer-truncated 70% and 20% counts, respectively, and the remaining images formed the test set. Table I summarizes these partitions.

TABLE I: Partitioning of original Positive and regular Negative images.

Class	Train	Val.	Test	Total
Positive/Dump	228	65	33	326
Negative/Clean	195	55	29	279
Total	423	120	62	605

D. Critical Negative Split and Characterization

The Critical Negative pool contained 199 original images annotated as non-dumping. The same partitioning procedure assigned 139 images to training, 39 to validation, and 21 to testing before augmentation (Table II).



(a) Positive/Dump Samples



(b) Clean/Negative Samples



(c) Critical Negative Samples

Fig. 1: Representative samples from the roadside dumping dataset: (a) four Positive/Dump examples, (b) three Clean/Negative examples, and (c) four Critical Negative examples.

Critical Negatives represent roadside scenes labeled as non-dumping that were selected as potentially challenging examples. In this study, Critical Negatives are characterized by visual features such as:

- broken asphalt, potholes, and rough gravel road textures;
- high-contrast shadows and abrupt illumination changes;
- minor scattered roadside litter in scenes labeled as non-dumping;
- wet street pavements reflecting buildings or trees; and
- accumulations of dry leaves and organic street debris.

TABLE II: Partitioning of original Critical Negative images.

Class	Train	Val.	Test	Total
Critical Negative	139	39	21	199

E. Offline Augmentation and Binary Class Merging

Five augmented variants were generated per original using Pillow in a separate preparation notebook. Each variant underwent horizontal flipping with probability 0.5, rotation with an angle sampled uniformly from $[-20°, 20°]$, brightness and contrast adjustments with factors sampled uniformly from $[0.75, 1.25]$, and a centered crop that removed approximately 4–12% from each edge before resizing to the original dimensions. The crop fraction was sampled uniformly from $[0.04, 0.12]$. Bilinear interpolation was used for rotation and resizing, and rotation retained the original canvas dimensions. Augmented images were saved as JPEG files with quality 90. The random generator initialized with seed 42 was used for both partitioning and augmentation.

YOLOv8x-cls additionally used online training augmentation, including random resized cropping, horizontal flipping, RandAugment, and random erasing. In Ultralytics 8.4.90, classification training enabled these transforms independently of the supplied `augment=False` argument. The baseline pipeline applied no additional online augmentation. The models therefore received different training transformations.

Retaining the original and five variants yielded six images per original in each split. Regular Negative and Critical Negative images were subsequently merged into the Negative class, while Positive images retained their class identity. Table III summarizes the resulting dataset.

TABLE III: Dataset counts after augmentation and merging of regular and Critical Negative images into the Negative class.

Final class	Train	Val.	Test	Total
Positive/Dump	1,368	390	198	1,956
Negative/Clean	2,004	564	300	2,868
Total	3,372	954	498	4,824

After merging, the Negative class contained 2,868 images and the Positive class contained 1,956 images. These unequal class frequencies were considered in model training and per-class evaluation.

IV. Model Configuration and Training Setup

Four classifier configurations were compared. YOLOv8x-cls [11] was initialized with ImageNet-pretrained weights and fine-tuned for binary classification. Random Forest, SVM, and an ANN implemented as a multilayer perceptron (MLP) were used as baselines [12]. The label ANN is retained throughout the results to identify this MLP configuration.

A. Baseline Feature Representation

The Random Forest, SVM, and ANN implementations used fixed-length feature vectors derived from resized RGB images. Let $I_k \in \mathbb{R}^{H\times W\times 3}$ be the k-th raw RGB image. For the baseline classifiers, each image is resized to $W_{base} \times H_{base} = 64 \times 64$ pixels, converted to floating-point values, divided by 255, and flattened to form a feature vector:

$$x_k = \frac{\text{flatten}(\text{resize}(I_k))}{255} \in [0,1]^D \tag{1}$$

where the input dimensionality is $D = 64 \times 64 \times 3 = 12,288$. Random Forest used these normalized pixel vectors directly. SVM and ANN additionally used z-score standardization fitted only on the training vectors:

$$z_k = \frac{x_k - \mu}{\sigma} \tag{2}$$

where μ and σ represent the feature-wise means and standard deviations computed from the training subset. Table IV summarizes the model configurations.

B. YOLOv8x-cls Architecture

YOLOv8x-cls uses convolutional layers and C2f blocks followed by a classification head [11]. C2f is a CSP bottleneck implementation with two convolutions. The ImageNet-pretrained classifier was adapted to two output classes and fine-tuned. The instantiated model contained 56,144,402 parameters before layer fusion, approximately 56.1 million.

C. Baseline Classifier Architectures

The three baseline classifiers were configured as follows:

- **Random Forest:** The model used 300 decision trees, with class weights set inversely proportional to training-class frequencies.
- **Support Vector Machine (SVM):** The model used a soft-margin classification objective with a radial basis function (RBF) kernel:

$$K(x, x') = \exp\left(-\gamma \|x - x'\|^2\right) \tag{3}$$

  where `gamma='scale'` sets $\gamma = 1/(D\,\text{Var}(X))$ for the standardized training matrix X. The regularization parameter was $C = 10$; larger values of C reduce regularization and increase the penalty for training errors.
- **Artificial Neural Network (ANN):** The model used three fully connected hidden layers with 512, 256, and 128 units, respectively. The activation function for hidden nodes is the Rectified Linear Unit (ReLU), defined as $f(z) = \max(0, z)$. The network used Adam with L2 regularization (`alpha=0.0001`). The data-fitting term was binary cross-entropy (BCE):

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right] \tag{4}$$

  where $y_i \in \{0, 1\}$ is the ground-truth label and $\hat{y}_i$ is the predicted probability.

TABLE IV: Model and training configuration.

Model	Configuration
YOLOv8x-cls	ImageNet-pretrained YOLOv8 classification model; 224×224 input; batch size 64; 30-epoch limit, 11 completed; patience 10; automatically selected AdamW; saved `best.pt` used for evaluation.
Random Forest	300 estimators; class-weight balanced; 64×64 RGB flattened features normalized to $[0, 1]$, without z-score standardization.
SVM	RBF kernel; $C = 10$; class-weight balanced; trained on standardized flattened features.
ANN	MLP with hidden layers 512, 256, and 128; ReLU activation; Adam optimizer; warm-start loop; 44 iterations completed; best weights restored from iteration 24.

Training used Ultralytics 8.4.90 on two NVIDIA Tesla T4 GPUs. Automatic optimizer selection chose AdamW with an initial learning rate of approximately 0.001667, overriding the requested value of 0.001. The saved configuration used `seed=0` and `deterministic=True`. Random Forest, SVM, and ANN used `random_state=42`.

The prepared counts in Table III differ from the effective counts accepted by the YOLO loader. Ultralytics excluded 17 training images and four validation images during image verification, leaving 3,355 training and 950 validation images. Its standalone test validation accepted 494 of 498 files. The recorded exceptions involved the read-only dataset filesystem. The baseline image loader processed all 3,372 training, 954 validation, and 498 test images. Consequently, the models used different effective training and validation sets.

D. Inference Workflow and Intended Use

During inference, an input image was processed by the Ultralytics classification pipeline, including resizing and center cropping to obtain a 224×224 input tensor. The saved YOLOv8x-cls checkpoint produced class probabilities, and the highest-probability class determined the Positive/Dump or Negative/Clean prediction. In a proposed application, Positive predictions would be flagged for human inspection. A Negative prediction would represent the model's classification rather than a guarantee that dumping is absent. Alert delivery and inspection management were outside the implemented experimental pipeline.

V. Training Behavior

A. YOLOv8x-cls Training

YOLOv8x-cls was configured for a maximum of 30 epochs with early-stopping patience of 10. Training stopped after 11 epochs, and final evaluation used the saved `best.pt` checkpoint selected by validation performance. Fig. 2 shows the recorded training loss, validation loss, and validation top-1 accuracy. Validation accuracy fluctuated, including a decline to 73.68% at epoch 3, and reached 98.32% at epochs 1 and 10.

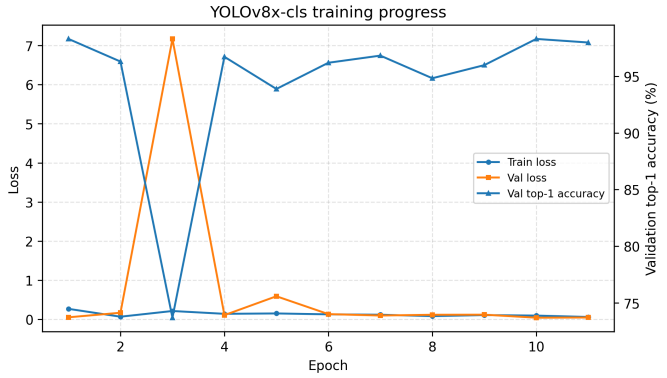

Fig. 2: YOLOv8x-cls training and validation loss and validation top-1 accuracy across 11 epochs.

B. ANN Training

The ANN was trained using a warm-start loop with one optimization epoch per iteration, a maximum of 100 iterations, and early stopping after 20 consecutive iterations without improved validation accuracy. It completed 44 iterations and restored the weights from iteration 24, which achieved the highest validation accuracy of 78.20%. Table VI reports training, validation, and test accuracies for those restored weights, whereas Fig. 3 shows the entire 44-iteration history. The training–validation gap indicates overfitting in this baseline configuration.

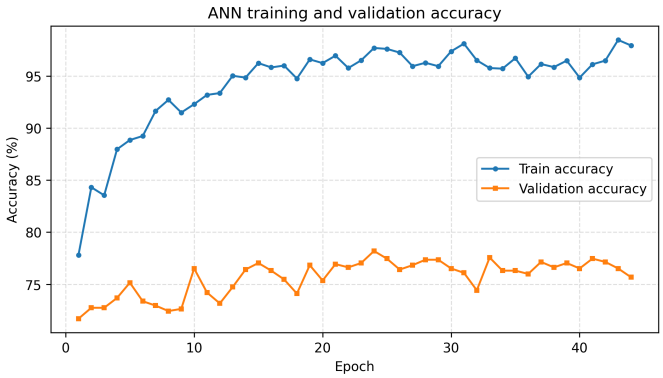

Fig. 3: ANN training and validation accuracy curve.

VI. Experimental Results

A. Performance Metrics

To evaluate model performance, standard classification metrics were computed based on the counts of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN):

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

B. Accuracy and Model Complexity Comparison

Table VI reports the train, validation, and test accuracy of the four models. Fig. 4 visualizes the accuracy comparisons.

Table V summarizes input resolution, model size descriptors, checkpoint storage, and test accuracy. Parameter counts, tree counts, and support-vector counts describe different model structures and are not directly equivalent complexity measures. Checkpoint sizes use decimal megabytes ($1\ \text{MB} = 10^6$ bytes) and do not by themselves establish deployment feasibility.

TABLE V: Comparison of input resolution, model parameters/support count, file storage footprint, and test accuracy.

Model	RGB input	Size descriptor	Disk	Test
YOLOv8x-cls	224×224	56.14M params	112.5 MB	**96.79%**
Random Forest	64×64	300 trees	10.2 MB	87.35%
SVM (RBF)	64×64	1,818 SVs	178.8 MB	81.93%
ANN (MLP)	64×64	6.46M params	103.3 MB	75.10%

Table V shows that SVM has a larger checkpoint than YOLOv8x-cls, while ANN has a similar storage requirement. Random Forest has the smallest checkpoint at 10.2 MB, approximately one-eleventh the size of the YOLOv8x-cls checkpoint. File size depends on numerical precision, serialization format, and retained model state as well as model structure.

The SVM checkpoint occupies 178.8 MB, approximately 58.9% more than the YOLO checkpoint when calculated from the exact file sizes. It stores 1,818 support vectors, comprising 1,002 Negative and 816 Positive vectors, each with 12,288 features. These account for 53.9% of the 3,372 baseline training images. The ANN checkpoint occupies 103.3 MB. Its input-to-first-hidden-layer connection alone contains $12,288 \times 512 = 6,291,456$ weights, contributing substantially to its storage requirement.

Checkpoint size alone does not establish deployment suitability. Although the execution logs contain GPU timing measurements, latency, memory consumption, and resource requirements must be evaluated on the intended hardware.

TABLE VI: Train, validation, and test accuracy of the four models.

Model	Train	Val.	Test
YOLOv8x-cls	99.97%	98.32%	96.79%
Random Forest	100.00%	84.91%	87.35%
SVM	99.97%	83.54%	81.93%
ANN	97.69%	78.20%	75.10%

YOLO train/validation accuracies use 3,355/950 accepted images; the baselines use 3,372/954 images. All test accuracies shown here use 498 images. YOLO's full-test value is taken from the separate prediction loop, rather than the 494-image `val()` result.

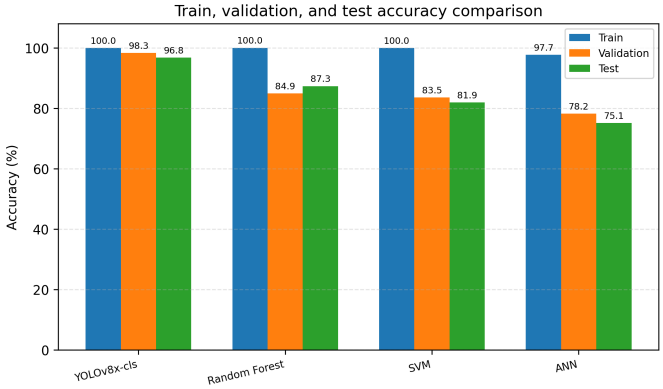

Fig. 4: Train, validation, and test accuracy comparison. YOLO train/validation values use its accepted subsets; all test values use 498 images. Bar labels are rounded to one decimal place.

C. Confusion Matrix Analysis

Fig. 5 shows test predictions with actual classes in rows and predicted classes in columns. The matrices include all 498 test images: 300 Negative and 198 Positive. YOLOv8x-cls produced 482 correct predictions, corresponding to 96.79% accuracy.

The Ultralytics validation routine excluded four Positive images during verification and reported 478 correct predictions among 494 accepted images, or 96.76%. The main comparative results in Tables V, VI, and VII and Fig. 5 use the separate prediction pass over all 498 images.

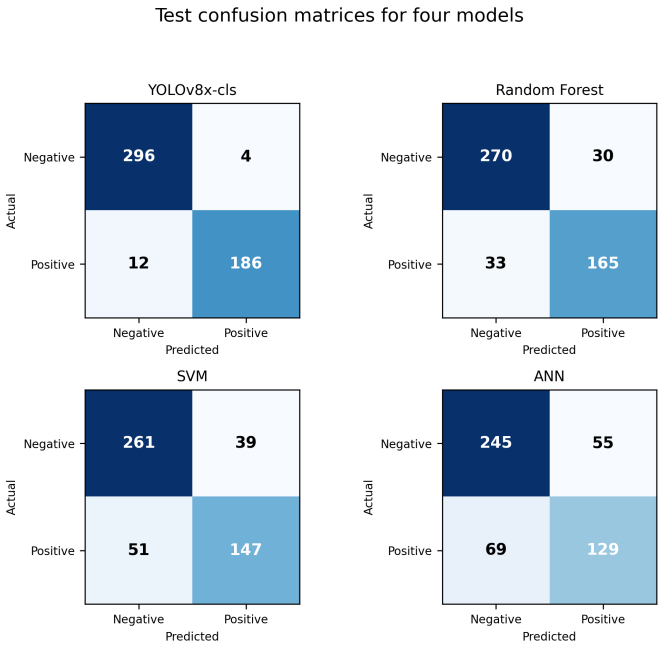

Fig. 5: Test confusion matrices for YOLOv8x-cls, Random Forest, SVM, and ANN. Rows indicate actual classes and columns indicate predicted classes.

YOLOv8x-cls correctly classified 296 of 300 Negative images and 186 of 198 Positive images, producing 16 errors. Random Forest produced 63 errors, SVM 90, and ANN 124. Among the evaluated configurations, YOLOv8x-cls achieved the highest recall for both classes: 98.67% for Negative and 93.94% for Positive images. Its four false-positive and twelve false-negative predictions corresponded to a false-positive rate of 1.33% and a false-negative rate of 6.06%, respectively.

TABLE VII: Per-class test-set classification performance computed from the confusion matrices.

Model	Class	Prec.	Recall	F1	Supp.
YOLOv8x	Neg.	96.10%	98.67%	97.37%	300
YOLOv8x	Pos.	97.89%	93.94%	95.88%	198
Random F.	Neg.	89.11%	90.00%	89.55%	300
Random F.	Pos.	84.62%	83.33%	83.97%	198
SVM	Neg.	83.65%	87.00%	85.29%	300
SVM	Pos.	79.03%	74.24%	76.56%	198
ANN	Neg.	78.03%	81.67%	79.80%	300
ANN	Pos.	70.11%	65.15%	67.54%	198

D. Model-by-Model Interpretation

The ImageNet-pretrained YOLOv8x-cls configuration achieved the highest accuracy on the prepared test set. This comparison evaluates complete training configurations and does not isolate the effects of convolutional architecture, pretraining, input resolution, or augmentation.

Random Forest achieved the highest test accuracy among the three baselines, at 87.35%. Its 100.00% training accuracy and lower validation and test accuracies indicate overfitting in this configuration. SVM achieved 99.97% training accuracy and 81.93% test accuracy, with lower recall for Positive

than Negative images. ANN had the largest training-to-test accuracy gap among the evaluated configurations. These observations concern the specific implementations and do not establish general limitations of the model families.

TABLE VIII: Materials supporting experimental reproducibility.

Material	Contents and purpose
YOLO checkpoint	Best and last YOLO classification checkpoints, including the checkpoint selected by validation performance.
YOLO logs and figures	Training-result CSV, training-progress figure, validation batch images, and YOLO confusion-matrix outputs.
Accuracy summaries	CSV and PNG summaries; the original YOLO test summary uses 494 images, while the confusion-matrix summary uses all 498.
Confusion matrices	Test confusion-matrix CSV and PNG files for YOLOv8x-cls, Random Forest, SVM, and ANN.
Baseline models	Serialized Random Forest, SVM, and ANN model files for reproducibility.
Preprocessing	Saved StandardScaler object used before SVM and ANN inference.
ANN history	Epoch-wise ANN train and validation accuracy used for the ANN curve.

VII. Discussion

The pretrained YOLOv8x-cls configuration outperformed the three evaluated baselines on the full prepared test set. The configurations differed in pretraining, image resolution, augmentation, model capacity, and the effective training and validation samples accepted by their loaders. Consequently, the observed gap cannot be attributed to convolutional architecture alone. Convolutional layers explicitly exploit local spatial relationships and shared filters. Flattening preserves pixels in a fixed order, but the baseline models lack the same convolutional structure; resizing to 64×64 also removes visual detail.

The preparation procedure retained each original image and its variants within one split but did not assess image-content duplicates or repeated locations across splits. The 498 test images comprise 83 original images and 415 augmented variants and should not be interpreted as 498 independent scenes. Evaluation on independently collected, unaugmented images is needed to assess generalization beyond this prepared test set.

Critical Negative examples were included to represent challenging non-dumping scenes, but their contribution was not isolated through a controlled comparison with training that excluded them. The observed error counts therefore cannot establish their effect on false-positive predictions. YOLOv8x-cls produced twelve missed-dump predictions and four false alarms; missed dumping images accounted for most observed errors. A separate evaluation of challenging Negative images and explicit annotation criteria for distinguishing minor litter from dumping are needed to assess false-positive risk and labeling consistency.

VIII. Future Work

Future work should evaluate the classifier on independently collected, unaugmented images and use consistent image-verification procedures across model configurations. A controlled comparison with and without Critical Negative training examples would assess their effect on false-positive predictions. Additional work should examine sensitivity to hyperparameter choices, assess uncertainty while accounting for variants of the same original image, and evaluate latency, memory consumption, and classification performance on the intended deployment hardware.

IX. Conclusion

This study compared four classifier configurations for roadside dumping classification using a locally collected Dhaka dataset. YOLOv8x-cls achieved the highest accuracy on the prepared test set, at 96.79%, compared with 87.35%, 81.93%, and 75.10% for Random Forest, SVM, and ANN. Its predictions included four false positives and twelve false negatives. These findings motivate further investigation of image classification as a screening component for human inspection. They do not establish an isolated architectural advantage because pretraining, resolution, augmentation, and accepted training images differed. The test set derived from 83 original images and their augmented variants. Independent testing, a controlled assessment of Critical Negative training examples, and evaluation on target hardware are needed before operational deployment.

Data and Code Availability

The roadside dumping dataset is available on Mendeley Data at https://doi.org/10.17632/rkh7nz2rcp.1. The source code is available at https://github.com/ruhul1845/DumpAlert.

AI Usage Declaration

The authors developed the core research idea, studied the related literature, collected the dataset, trained the models, and conducted the experiments. During manuscript preparation, ChatGPT was used to draft and refine text based on the authors' own notes, writing plans, methodological descriptions, and experimental results. AI assistance supported wording and organization; the reported data and experimental results came from the authors' work and were not generated by AI. The authors take responsibility for the accuracy and integrity of the final manuscript.

References

[1] C. Padubidri, A. Kamilaris, and S. Karatsiolis, "Accurate detection of illegal dumping sites using high resolution aerial photography and deep learning," in *Proc. IEEE PerCom Workshops*, 2022, pp. 451–456, doi: 10.1109/PerComWorkshops53856.2022.9767451.

[2] S. Shahab and M. Anjum, "Solid waste management scenario in India and illegal dump detection using deep learning: An AI approach towards the sustainable waste management," *Sustainability*, vol. 14, no. 23, Art. no. 15896, 2022, doi: 10.3390/su142315896.

[3] P. Wibowo, A. Z. Wajdi, T. H. Muliawati, A. Fariza, and Y. Kiyoki, "Illegal waste dump detection using a deep learning approach: Performance evaluation of YOLO architectures," in *Proc. 2025 International Electronics Symposium (IES)*, 2025, pp. 802–807, doi: 10.1109/IES67184.2025.11161008.

[4] M. Yang and G. Thung, "Classification of trash for recyclability status," CS229 Project Report, Stanford University, 2016.
[5] R. A. Aral, S. R. Keskin, M. Kaya, and M. Haciomeroglu, "Classification of TrashNet dataset based on deep learning models," in *Proc. 2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 2058–2062, doi: 10.1109/BigData.2018.8622212.
[6] R. Kumar, A. Lall, S. Chaudhari, M. Kale, and A. Vattem, "Garbage vulnerable point monitoring using IoT and computer vision," in *Proc. 2025 IEEE 11th World Forum on Internet of Things (WF-IoT)*, 2025, doi: 10.1109/WF-IoT64238.2025.11270560.
[7] L. Izquierdo-Horna, J. Zevallos, and M. Angulo, "Strategic monitoring of improperly disposed urban waste using UAV imagery and object detection," *Resources, Conservation & Recycling Advances*, vol. 29, Art. no. 200306, 2026, doi: 10.1016/j.rcradv.2025.200306.
[8] P. F. Proença and P. Simões, "TACO: Trash annotations in context for litter detection," arXiv preprint arXiv:2003.06975, 2020.
[9] J. Redmon et al., "You only look once: Unified, real-time object detection," in *Proc. IEEE CVPR*, 2016, pp. 779–788.
[10] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv:2004.10934, 2020.
[11] Ultralytics, "Explore Ultralytics YOLOv8," Ultralytics Docs. Accessed: Oct. 6, 2026. [Online]. Available: https://docs.ultralytics.com/models/yolov8/
[12] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.