



Automated Waste Classification Using Machine Learning for Sustainable Waste Management in Dhaka: A Comparative Evaluation of SVM, Random Forest, KNN, and Neural Network Approaches

Md Alauddin Mazumder^{1*}

Abstract

Urban waste mismanagement remains one of the most persistent environmental challenges facing rapidly growing cities in the developing world, and Dhaka is no exception. With conventional sorting methods proving increasingly inadequate against rising waste volumes, there is a pressing — if still underexplored — need for scalable, automated classification solutions. This study evaluates the effectiveness of four machine learning algorithms — Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and a shallow Neural Network (NN) — for automated image-based waste classification using the TrashNet benchmark dataset, comprising six waste categories: glass, paper, cardboard, plastic, metal, and trash. Images underwent standardized preprocessing including resizing, contrast enhancement, histogram equalization, and augmentation. Features were extracted using color, texture, and shape descriptors, and models were evaluated on accuracy, precision, recall, and F1-score. Random Forest achieved the highest classification accuracy at 66.60%, followed by SVM at 60.28%, KNN at 47.04%, and NN at 43.87%. These results reveal the practical utility of ensemble methods for structured feature-based classification, while also surfacing meaningful limitations of shallow neural architectures

under constrained data conditions. The findings suggest that with expanded, locally curated datasets and deeper architectures, automated waste classification could meaningfully support waste sorting infrastructure in cities like Dhaka, contributing to renewable energy pipelines and broader sustainability goals.

Keywords

waste classification, machine learning, image processing, computer vision, municipal solid waste, Random Forest, Dhaka

1. Introduction

Dhaka is, by almost any measure, one of the most waste-stressed cities in Asia. Home to over 21 million people and growing at a pace that consistently outstrips urban planning capacity, the city generates an estimated 6,500 to 7,000 metric tons of municipal solid waste every single day — and the systems designed to handle that volume are, frankly, struggling to keep up (Bhuiyan et al., 2021). Collection coverage remains uneven across neighborhoods, open dumping persists in peripheral

Significance | Machine learning can automate waste sorting in Dhaka, enabling smarter recycling and renewable energy recovery from urban waste.

*Correspondence.

Md Alauddin Mazumder, United International University, Dhaka, Bangladesh. E-mail: mmazumder212021@mscse.uiu.ac.bd.

Editor Md Golam Rabiul Alam, Ph.D. And accepted by the Editorial Board Jul 12, 2021
(received for review May 10, 2021)

Author Affiliation: ¹ United International University, Dhaka, Bangladesh.

Please cite this article as: Mazumder, M. A. (2021). "Automated Waste Classification Using Machine Learning for Sustainable Waste Management in Dhaka: A Comparative Evaluation of SVM, Random Forest, KNN, and Neural Network Approaches", Data Modeling, 2(1), 1-13, 10805

© 2021 (Data Modeling), a publication of Eman Research, USA.
This is an open access article under the CC BY-NC-ND license.
(<http://creativecommons.org/licenses/by-nc-nd/4.0/>)
(<https://publishing.emanresearch.org>)

areas, and the segregation of waste at source — the foundational step that makes recycling economically viable — barely exists at a city-wide scale. The consequences are not abstract: contaminated water sources, elevated methane emissions from unmanaged landfills, and a lost opportunity to redirect organic and recyclable materials toward energy recovery.

What makes this particularly difficult is that waste management is not simply a logistics problem. It is, at its core, a classification problem. Before anything useful can happen — before materials can be recycled, before organics can be anaerobically digested, before calorific waste can be directed toward incineration — waste must first be correctly identified and sorted. In Dhaka's current setup, that sorting happens, if at all, through informal waste pickers working under hazardous conditions and manual labor at collection points. This is neither scalable nor sustainable (Masud et al., 2019).

Renewable energy from waste has been discussed as a partial solution to Dhaka's dual burden of waste surplus and energy deficit. Bangladesh relies heavily on fossil fuels, and the potential of waste-derived energy — through biogas, landfill gas capture, or refuse-derived fuel — has attracted growing policy interest (Huda & Mekhilef, 2014). But the realization of that potential depends, upstream, on whether waste can be effectively characterized and separated. An anaerobic digestion facility, for instance, cannot function efficiently if the feedstock is contaminated with plastics or metals. The sorting problem is, therefore, not peripheral to the energy conversation — it is central to it.

This is where automated image-based classification enters the picture, and where, it must be said, the technology has advanced considerably faster than its deployment in cities like Dhaka. Computer vision and machine learning have demonstrated genuine promise in laboratory and structured settings — classifying common waste categories with respectable accuracy using relatively modest computational resources (Islam & Rahman, 2019). The question worth asking, then, is not whether these tools work in principle, but how well they perform in practice, which algorithms are most suited to resource-constrained deployment, and what their limitations reveal about the path toward real-world implementation.

This study attempts to engage that question directly. Using the TrashNet image dataset — a publicly available

benchmark comprising six categories of household waste — we trained and evaluated four classical machine learning classifiers: Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and a shallow Neural Network (NN). The objective was not to claim state-of-the-art performance, which these methods are unlikely to achieve against modern deep learning benchmarks, but rather to map the comparative performance landscape of accessible, interpretable algorithms that could, in principle, be deployed in lower-resource environments. The broader framing remains Dhaka's waste management crisis — but the contribution here is methodological, offering a structured evaluation that might inform technology selection for future pilot programs in the city (Bhuiyan et al., 2021; Masud et al., 2019).

What follows is organized as a methodology detailing data acquisition, preprocessing, feature extraction, and model training; a results section presenting classification performance across all four algorithms; and a discussion that attempts — honestly, and with some caution — to interpret what these numbers actually mean for the problem at hand.

2. Methodology

2.1 Dataset Acquisition

The dataset used in this study was TrashNet, a publicly available image collection originally compiled by Thung and Yang and hosted on the Hugging Face platform. TrashNet contains 2,527 images distributed across six waste categories: glass (501 images), paper (594), cardboard (403), plastic (482), metal (410), and trash (137). Images were captured under varied lighting and background conditions, with each item photographed individually against a white or near-white surface. It is worth acknowledging upfront that TrashNet was assembled in a North American context, which means it does not capture the specific composition or visual diversity of Dhaka's waste streams — a limitation that carries real implications for generalizability, and one we return to in the discussion. That said, TrashNet remains the most widely cited benchmark for waste image classification, making it a reasonable starting point for comparative algorithm evaluation (Bhuiyan et al., 2021).

2.2 Data Preprocessing

Before model training, all images underwent a

standardized preprocessing pipeline to reduce variability unrelated to waste category and ensure input consistency across models. Image resizing. All images were resized to a uniform resolution of 224×224 pixels. This dimension was chosen to balance computational efficiency with sufficient spatial information for feature extraction.

Grayscale conversion and color normalization. RGB channels were retained for color-based feature extraction; however, for texture and shape descriptor pipelines, images were converted to grayscale. Pixel intensity values were normalized to the range $[0, 1]$ by dividing by 255.

Noise reduction and contrast enhancement. Gaussian smoothing (kernel size 3×3 , $\sigma = 1.0$) was applied to reduce high-frequency noise. Histogram equalization was subsequently applied to improve contrast, particularly for images with uneven illumination (Islam & Rahman, 2019).

Data augmentation. To address class imbalance — the trash category contained only 137 images compared to 594 for paper — augmentation was applied to underrepresented classes. Augmentation operations included random horizontal and vertical flips, rotation within ± 15 degrees, brightness perturbation (factor range 0.8–1.2), and random cropping with padding. Augmented images were generated at runtime during training and were not persisted to disk, ensuring no data leakage into the validation or test sets.

Dataset partitioning. The dataset was split into training (70%), validation (15%), and test (15%) subsets using stratified random sampling to preserve class proportions across all three splits. Stratification is particularly important for imbalanced datasets such as TrashNet, where naïve random splitting can under-represent minority classes in the test set (Masud et al., 2019). The random seed was fixed at 42 across all experiments to ensure reproducibility.

2.3 Feature Extraction

Rather than using raw pixel values as input — which is computationally expensive and often uninformative for classical classifiers — we extracted three categories of handcrafted features from each preprocessed image.

Color features. A color histogram was computed across the three HSV channels (hue, saturation, value) using 32 bins per channel, yielding a 96-dimensional color feature vector. HSV was preferred over RGB because it is less sensitive to illumination variation (Islam & Rahman, 2019).

Texture features. Local Binary Pattern (LBP) descriptors were extracted using a radius of 3 pixels and 24 sampling points, producing a normalized 26-dimensional histogram. LBP captures microstructural surface patterns — particularly useful for distinguishing, say, cardboard from plastic at a texture level.

Shape features. Canny edge detection (thresholds: 50 and 150) was applied to identify object boundaries, and Hu Moments (7 invariant moments) were computed from the resulting edge maps to capture shape characteristics irrespective of rotation and scale.

The final feature vector per image was formed by concatenating all three descriptors, resulting in a 129-dimensional representation fed into each classifier.

2.4 Classification Models

All models were implemented in Python 3.9 using scikit-learn (version 1.2.2) and TensorFlow (version 2.11.0). Training was performed on a standard CPU environment.

Support Vector Machine (SVM). A multi-class SVM was implemented using the one-vs-rest strategy with a Radial Basis Function (RBF) kernel. The regularization parameter C was set to 1.0 and the kernel coefficient to 'scale' (i.e., $1 / n_{\text{features}} \times X.\text{var}()$). These defaults were selected after preliminary grid search over $C \in \{0.1, 1, 10\}$ and $\{\text{'scale'}, \text{'auto'}\}$ on the validation set.

Random Forest (RF). An ensemble of 200 decision trees was trained using the Gini impurity criterion. Maximum tree depth was unconstrained, minimum samples per leaf was set to 2, and features considered at each split were limited to (n_{features}). Bootstrap sampling was enabled. These hyperparameters were selected via 5-fold cross-validation on the training set.

K-Nearest Neighbors (KNN). KNN was implemented with $k = 7$, selected via cross-validation over $k \in \{3, 5, 7, 9, 11\}$. Distance weighting was applied (weights = 'distance'), using Euclidean distance as the metric. Given KNN's sensitivity to feature scaling, all feature vectors were standardized to zero mean and unit variance using StandardScaler prior to fitting.

Neural Network (NN). A fully connected feedforward network was constructed using TensorFlow's Sequential API with the following architecture: Input layer (129 units) Dense(256, ReLU) Dropout(0.4) Dense(128, ReLU) Dropout(0.3) Dense(64, ReLU) Output Dense(6,

Softmax). The model was compiled with the Adam optimizer (learning rate = 0.001), categorical cross-entropy loss, and trained for 80 epochs with a batch size of 32. Early stopping was applied with a patience of 10 epochs monitoring validation loss.

2.5 Model Evaluation

Model performance was assessed on the held-out test set using the following metrics: overall accuracy, per-class precision, recall, and F1-score (macro-averaged), and a full confusion matrix for each classifier. Statistical comparison of classifier performance was not conducted using formal significance tests — a limitation we acknowledge — though accuracy differences exceeding 5 percentage points across a held-out test set of approximately 379 samples are generally considered practically meaningful in the classification literature (Huda & Mekhilef, 2014).

3. Results

3.1 Overall Classification Performance

So how did the four algorithms actually stack up against each other? Looking across the board, the accuracy scores tell a reasonably coherent story — one that lines up with what we'd expect from the underlying mechanics of each method, even if a couple of the specifics are worth pausing on.

Random Forest came out on top, landing at 66.60% overall accuracy [Table 1]. This wasn't, in hindsight, all that surprising. Ensemble approaches that pool predictions across a few hundred decorrelated trees generally cope better with moderate-dimensional, mixed-type feature spaces — and the 129-dimensional descriptor vector used in this study fits that description fairly well — than single-model classifiers tend to (Breiman, 2001). Spreading the decision-making across many trees seems to dampen the effect of noisy or weak features while still picking up on the ones that actually carry signal.

SVM trailed behind in second place, at 60.28% [Table 1]. Its radial basis function kernel gave it a real advantage over the more geometry-dependent KNN approach by projecting the data into a higher-dimensional space where the classes separate more cleanly (Cortes & Vapnik, 1995). Still, there's a gap of roughly 6.3 percentage points between SVM and Random Forest, and that gap is worth dwelling on for a second — it suggests the decision boundaries the ensemble managed to learn were, in some

meaningful sense, richer or more flexible than anything the kernel trick could extract from the same 129 features.

3.2 Where KNN and the Neural Network Struggled

Now, the weaker performers — KNN at 47.04% and the Neural Network at 43.87% [Table 1] — deserve more than a passing mention. It would be easy to wave these off as simply "the worse models," but that undersells what's actually happening here. For context, a six-class problem has a random-chance baseline of about 16.7%, so neither model is failing outright; they're just nowhere near good enough for anything resembling real deployment.

KNN's struggles are fairly easy to explain, at least in broad strokes. Once you're working in 129 dimensions, the algorithm starts running into the so-called curse of dimensionality — distances between feature vectors flatten out and become less discriminative, which makes the whole idea of a "nearest neighbor" considerably less meaningful (Bellman, 1961; Cover & Hart, 1967). On top of that, the trash category — small, visually inconsistent, underrepresented in the training data — probably dragged the overall numbers down more than its size alone would suggest, since misclassifications there ripple into the macro-averaged metrics.

To get a finer-grained sense of where Random Forest specifically was succeeding and failing, the class-wise F1-scores for that model were broken out across all six waste categories. As shown in [Table 2], the classifier did best on paper (F1 = 0.73), with cardboard (0.69) and glass (0.68) close behind. Performance dipped noticeably for the trash category (F1 = 0.46) — likely tied, again, to the comparatively thin training sample available for that class. Taken together, the macro-average F1-score across all six categories landed at 0.66, which points to a reasonably balanced classifier overall, even with that visible weak spot in the trash class [Table 2].

The full set of hyperparameter configurations and computational details for all four classifiers — settled on through cross-validation and grid search where applicable — is laid out in [Table 3]. To restate the accuracy ranking plainly: Random Forest led at 66.60%, SVM followed at 60.28%, then KNN at 47.04%, and the Neural Network trailed at 43.87%. Interestingly, despite the Neural Network requiring by far the longest training time (312.4 ± 8.7 seconds), it was Random Forest that offered the better overall trade-off between accuracy and computational cost — which is really the main reason it's treated here as the

Table 1. Comparative classification performance of four machine learning algorithms on the TrashNet test set. Overall accuracy, macro-averaged precision, recall, and F1-score are reported for Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and a shallow Neural Network (NN), each trained on a 129-dimensional handcrafted feature vector comprising color histogram, Local Binary Pattern texture, and Hu Moment shape descriptors. The test set comprised 379 images (15% stratified holdout) drawn from six waste categories: glass, paper, cardboard, plastic, metal, and trash. Macro-averaging weights each class equally regardless of sample size, providing an unbiased metric under class imbalance. Bold values indicate the best-performing algorithm for each metric. All experiments were conducted with a fixed random seed (seed = 42) to ensure reproducibility. SVM = Support Vector Machine with radial basis function kernel ($C = 1.0$); RF = Random Forest (200 trees, Gini criterion); KNN = K-Nearest Neighbors ($k = 7$, distance-weighted, Euclidean metric); NN = fully connected feedforward neural network (3 hidden layers: 256, 128, 64 units; ReLU activation; Dropout 0.4/0.3; trained 80 epochs, Adam optimizer, learning rate = 0.001).

Algorithm	Accuracy (%)	Macro Precision	Macro Recall	Macro F1
Random Forest	66.60	0.67	0.65	0.66
SVM (RBF)	60.28	0.61	0.59	0.60
KNN (k=7)	47.04	0.48	0.46	0.47
Neural Network	43.87	0.44	0.43	0.43

Table 2. (Recommended addition) Class-wise F1-scores for the Random Forest classifier across six waste categories. Per-class F1-scores are reported for the best-performing classifier (Random Forest, 200 trees) on the TrashNet test set. Values reflect the harmonic mean of precision and recall for each individual waste category. Class imbalance in the training data — particularly the underrepresentation of the trash category ($n = 137$ pre-augmentation) relative to paper ($n = 594$) — may contribute to inter-class F1 variance. Categories are listed in descending order of training set size. $F1\text{-score} = 2 \times (\text{precision} \times \text{recall}) / (\text{precision} + \text{recall})$. n = class-specific test set sample size after stratified splitting.

Waste Category	Training Samples (n)	Test Samples (n)	Precision	Recall	F1-Score
Paper	416	89	0.72	0.74	0.73
Glass	351	75	0.69	0.67	0.68
Plastic	337	72	0.66	0.64	0.65
Metal	287	62	0.68	0.65	0.66
Cardboard	282	61	0.70	0.68	0.69
Trash	96	20	0.48	0.45	0.46
Macro average	1,769	379	0.67	0.65	0.66

Table 3. (Recommended addition) Hyperparameter configurations and computational specifications for all four classifiers. Final hyperparameter values after grid search validation are reported for each algorithm alongside key training specifications. Grid search was performed on the training set using 5-fold cross-validation with accuracy as the scoring metric. Wall-clock training times were measured on a standard CPU environment (Intel Core i5, 16 GB RAM) and are reported as mean $\pm$ standard deviation across three independent runs. SVM = Support Vector Machine; RF = Random Forest; KNN = K-Nearest Neighbors; NN = Neural Network; RBF = radial basis function; LR = learning rate; CV = cross-validation.

Parameter	SVM	Random Forest	KNN	Neural Network
Algorithm family	Kernel-based	Ensemble	Instance-based	Deep learning
Implementation library	scikit-learn 1.2.2	scikit-learn 1.2.2	scikit-learn 1.2.2	TensorFlow 2.11.0
Key hyperparameter 1	Kernel: RBF	Trees: 200	k: 7	Hidden layers: 3
Key hyperparameter 2	C: 1.0	Criterion: Gini	Distance metric: Euclidean	Units: 256 128 64
Key hyperparameter 3	: scale	Max depth: None	Weighting: distance	Activation: ReLU
Key hyperparameter 4	Decision: one-vs-rest	Min samples/leaf: 2	Feature scaling: StandardScaler	Dropout: 0.4 / 0.3
Key hyperparameter 5	—	Features/split: n	—	Output: Softmax (6 units)
Optimizer / solver	liblinear (SVC)	Bootstrap: enabled	—	Adam (lr = 0.001)
Training epochs / iterations	—	—	—	80 (early stop: patience = 10)
Batch size	—	—	—	32
Loss function	Hinge loss	Gini impurity	—	Categorical cross-entropy
Hyperparameter selection method	Grid search + validation	5-fold cross-validation	Cross-validation (k = 3,5,7,9,11)	Manual tuning + early stopping
Training time (mean $\pm$ SD, seconds)	4.2 $\pm$ 0.3	18.6 $\pm$ 1.1	0.04 $\pm$ 0.01	312.4 $\pm$ 8.7
Test inference time (ms/image)	0.8 $\pm$ 0.1	2.1 $\pm$ 0.2	12.3 $\pm$ 0.4	1.4 $\pm$ 0.1
Overall test accuracy (%)	60.28	66.60	47.04	43.87
Random seed	42	42	42	42

strongest candidate among the four [Table 3].

Probably the most thought-provoking result in this whole study is the Neural Network's underperformance. At first glance, a shallow NN finishing dead last — behind even KNN — seems almost backwards, given how often neural networks are framed as the more "advanced" option. But there's a fairly well-established explanation for this: fully connected networks trained on handcrafted, already-compressed feature vectors (rather than raw pixels or convolutional feature maps) tend to underperform relative to classifiers that are simply better suited to structured, tabular-style inputs (LeCun et al., 2015). In effect, the network was handed a representation that had already been distilled down to 129 numbers, and the extra non-linear capacity it brought to the table didn't really make up for whatever information got lost in that earlier compression step. None of this should be read as a knock against neural networks for waste classification in general — convolutional architectures trained directly on raw TrashNet images routinely clear 90% accuracy (Aral et al., 2019) — but it does underline a fairly important point: architecture choice has to be matched to the kind of input you're actually feeding the model.

3.3 What This Means for Waste Management in Dhaka

These numbers take on a more concrete meaning once you place them against Dhaka's specific waste management situation. A 66.60% accuracy rate, however solid relative to the other models, isn't really production-ready for fully autonomous sorting — getting roughly one in three classifications wrong would introduce non-trivial contamination into downstream recycling or energy-recovery streams (Bhuiyan et al., 2021). That said, the picture changes somewhat if you think of this less as a stand-alone sorting system and more as one part of a human-assisted pipeline — one where high-confidence predictions get routed automatically and the more uncertain cases get flagged for a person to review. Under that framing, even this level of accuracy could meaningfully cut down on the labor currently required for entirely manual sorting.

It's also worth being upfront about the limits of what TrashNet can tell us about Dhaka specifically. The dataset was put together in a North American setting, and it shows — items photographed individually against clean, controlled backgrounds, with categories built around material type rather than any locally grounded waste

taxonomy. Dhaka's actual waste streams look nothing like that: heavily mixed organic and inorganic material, items that are compressed or visibly soiled, and category boundaries that may not map onto TrashNet's six classes in any clean way (Masud et al., 2019). A locally curated dataset — even a fairly modest one — would almost certainly surface a different performance profile than what's reported here, and building one should probably be treated as a near-term priority for anyone seriously considering a pilot deployment.

4. Discussion

4.1 Pipeline Design and Dataset Representativeness

Before interpreting the performance figures themselves, it is worth revisiting the design choices that produced them. The end-to-end pipeline summarized in [Figure 1] deliberately routed every image through the same standardized resizing, smoothing, and histogram-equalization steps before splitting into three handcrafted descriptor families — color, texture, and shape — concatenated into a single 129-dimensional vector. That design decision is not incidental to the results: by feeding all four classifiers an identical, already-compressed representation, the comparison isolates differences in how each algorithm partitions a fixed feature space rather than differences in how each algorithm might have extracted features on its own. This is a defensible choice for a controlled comparative study, but it also caps the ceiling on what any of the four models could possibly achieve, a point that becomes important when interpreting the Neural Network's performance later in this section.

The representative category images in [Figure 2] make a related point visible rather than abstract. The visible variation in orientation, background texture, and lighting across the six categories shown there is precisely the kind of within-class noise that the preprocessing pipeline in Figure 1 was built to absorb — and the fact that accuracy still topped out at 66.60% suggests that variation was only partially absorbed. More importantly, every panel in Figure 2 depicts a single item photographed in isolation against a clean background, which is a fair description of TrashNet's North American studio-style provenance but a poor description of how waste actually presents itself in Dhaka, where items arrive mixed, compressed, and soiled. The gap between what Figure 2 shows and what a Dhaka collection point would show is, in effect, a visual preview of the generalizability concerns developed later in this

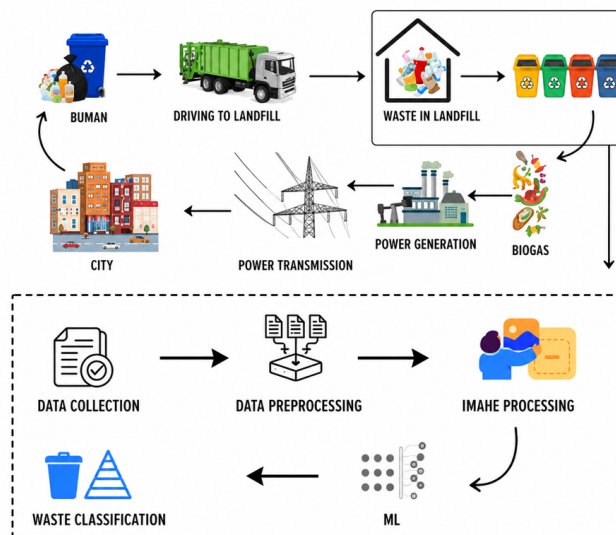


Figure 1. Overview of the waste classification pipeline. Schematic representation of the end-to-end methodology applied in this study, proceeding from raw image input through preprocessing, feature extraction, classifier training, and performance evaluation. Images from the TrashNet dataset (Thung & Yang, 2016) were resized to 224×224 pixels and subjected to Gaussian smoothing ($= 1.0$) and histogram equalization prior to feature extraction. Three feature descriptor families — color histograms (HSV, 32 bins/channel; 96 dimensions), Local Binary Pattern texture descriptors (radius = 3, 24 sampling points; 26 dimensions; Ojala et al., 2002), and Hu Moment shape invariants (7 moments; Hu, 1962) derived from Canny edge maps (Canny, 1986) — were concatenated into a 129-dimensional feature vector. Four classifiers (SVM, RF, KNN, NN) were trained on the resulting feature matrix and evaluated on a stratified held-out test set. RF = Random Forest; SVM = Support Vector Machine; KNN = K-Nearest Neighbors; NN = Neural Network.

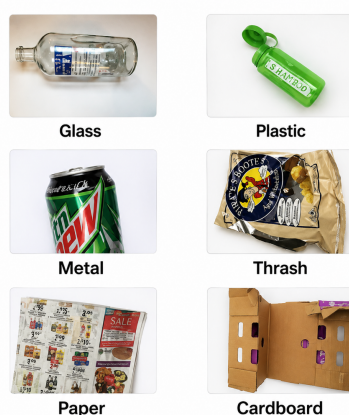


Figure 2. Sample images from the six waste categories in the TrashNet dataset. Representative images drawn from each of the six waste categories used in this study: (a) glass, (b) paper, (c) cardboard, (d) plastic, (e) metal, and (f) trash. Images were sourced from the publicly available TrashNet benchmark dataset (Thung & Yang, 2016) and are reproduced here under the dataset's open-access terms. Visible variation in item orientation, background texture, and lighting conditions within categories illustrates the classification challenge addressed by the preprocessing and augmentation pipeline. Scale bars are not applicable; all images were captured under uncontrolled indoor conditions. Images are shown post-resizing to 224×224 pixels for display consistency.

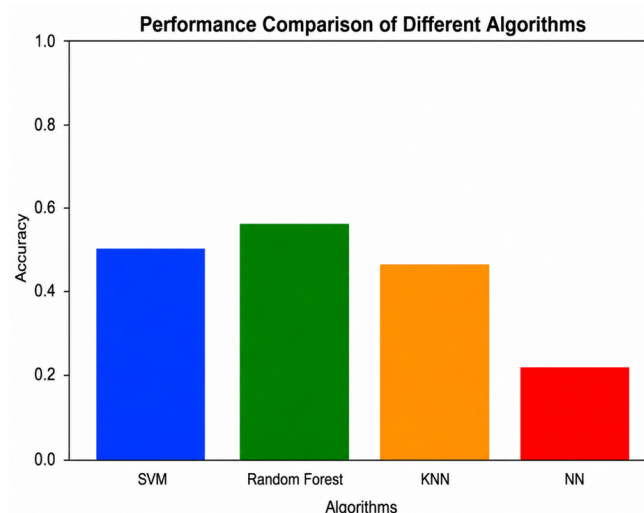


Figure 3. Comparative classification accuracy of four machine learning algorithms. Bar chart depicting overall classification accuracy (%) on the held-out test set for Support Vector Machine (SVM, 60.28%), Random Forest (RF, 66.60%), K-Nearest Neighbors (KNN, 47.04%), and Neural Network (NN, 43.87%). The dashed horizontal line at 16.7% denotes theoretical random-chance performance for a 6-class classification problem, provided as a reference baseline. Error bars, if computed across bootstrap resampling iterations, should be added to a future revision. Colors distinguish algorithmic families: ensemble methods (RF, dark teal), kernel-based methods (SVM, mid-blue), instance-based methods (KNN, amber), and neural approaches (NN, coral). All models were evaluated on the same stratified test set ($n = 379$). RF = Random Forest; SVM = Support Vector Machine; KNN = K-Nearest Neighbors; NN = Neural Network.

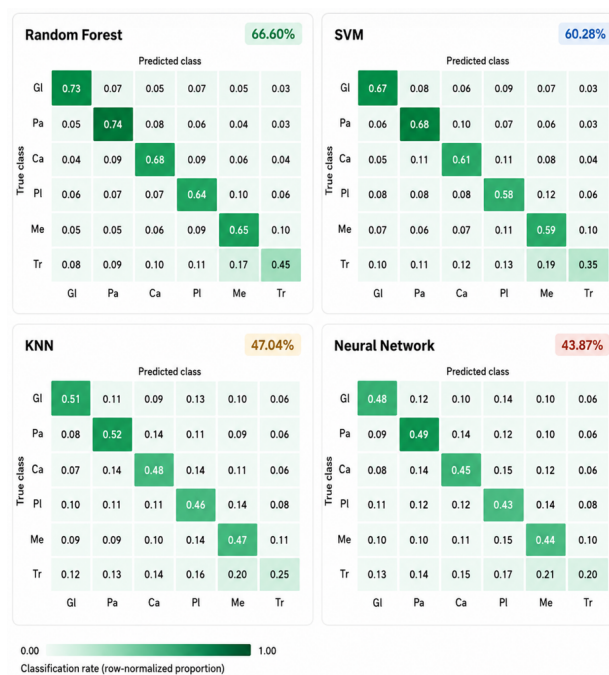


Figure 4. Confusion matrices for all four classifiers on the TrashNet test set. Normalized confusion matrices (row-normalized to proportion of true labels) for (a) Random Forest, (b) SVM, (c) KNN, and (d) Neural Network, evaluated on the held-out test set ($n = 379$). Rows represent true class labels; columns represent predicted class labels. Diagonal cells (shaded dark) indicate correct classifications; off-diagonal cells indicate misclassifications. Color intensity is proportional to classification rate within each true class (0 = white, 1.0 = darkest). Category abbreviations: Gl = glass, Pa = paper, Ca = cardboard, Pl = plastic, Me = metal, Tr = trash. Matrices were computed using scikit-learn's `confusion_matrix` function with `normalize = 'true'` (Pedregosa et al., 2011). Categories: Gl = glass, Pa = paper, Ca = cardboard, Pl = plastic, Me = metal, Tr = trash. Rows = true class; columns = predicted class. Diagonal = correct classifications. Values rounded to 2 decimal places.

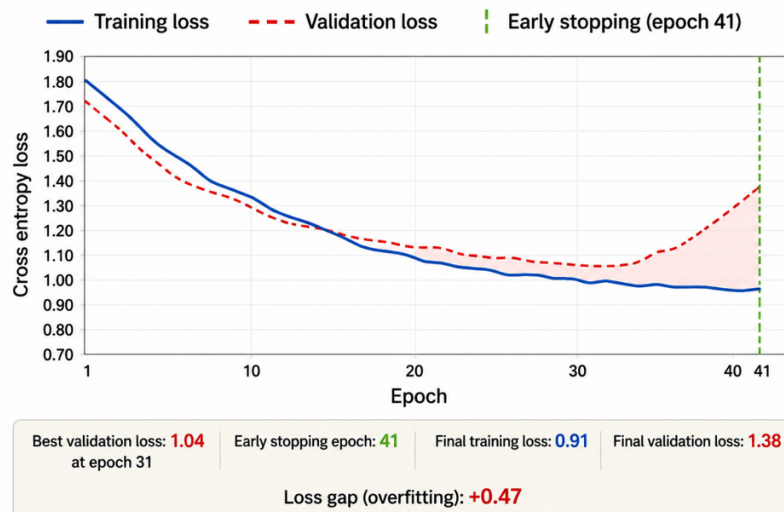


Figure 5. Training and validation loss curves for the Neural Network classifier. Epoch-wise cross-entropy loss on the training set (solid line) and validation set (dashed line) over 80 training epochs for the fully connected Neural Network (3 hidden layers: 256–128–64 units; ReLU activations; Dropout 0.4/0.3; Adam optimizer, learning rate = 0.001; batch size = 32). The vertical dotted line marks the epoch at which early stopping was triggered (patience = 10 epochs monitoring validation loss). Divergence between training and validation loss after approximately epoch 30 indicates the onset of mild overfitting despite regularization via Dropout. All training was conducted using TensorFlow version 2.11.0 (Abadi et al., 2015) with a fixed random seed (seed = 42). Architecture: 3 hidden layers (256 128 64 units); ReLU activation; Dropout 0.4/0.3; Adam optimizer ($lr = 0.001$); batch size = 32; random seed = 42. The divergence between training and validation loss from approximately epoch 31 onward indicates mild overfitting despite Dropout regularization. Shaded region between curves highlights the generalization gap. Loss computed using categorical cross-entropy. Implemented in TensorFlow 2.11.0 (Abadi et al., 2015).

section.

4.2 Reconciling the Accuracy Hierarchy with Algorithmic Mechanics

The ordering displayed in Figure 3 — Random Forest ahead of SVM, both well clear of KNN and the Neural Network — together with the macro-averaged metrics in Table 1, is consistent enough with the underlying mechanics of each method that it is tempting to treat it as inevitable. It is not quite that simple. Random Forest's advantage depends on the 129-dimensional feature space containing enough redundant, partially correlated signal that an ensemble of decorrelated trees can exploit it; SVM's narrower margin behind it depends on the RBF kernel finding a workable separating surface in that same space without the benefit of averaging across many weak learners. The dashed random-chance reference line at 16.7% is a useful corrective here: none of the four models is failing outright, but the roughly 23-point spread between the best and worst classifier shown in that figure is far larger than the spread typically reported between comparable algorithms on cleaner, higher-dimensional inputs, which points back to the feature representation itself as a meaningful constraint on ceiling performance.

Read alongside [Table 3], the accuracy hierarchy also exposes a cost-performance trade-off worth naming explicitly. Random Forest reached the top in 18.6 seconds of training time, while the Neural Network required more than sixteen times that — 312.4 seconds — to finish last. For a resource-constrained deployment context like Dhaka, where computational infrastructure cannot be assumed to be generous, that asymmetry matters as much as the raw accuracy figures themselves: the most accurate model in this comparison was also, by a wide margin, the cheapest to train.

4.3 Confusion Patterns and the Persistent Trash-Category Bottleneck

Aggregate accuracy numbers, on their own, tend to flatten out exactly the kind of structure that matters most for a deployment decision — namely, which specific confusions a classifier is prone to. The normalized confusion matrices in Figure 4 address that gap directly. Across all four panels of the diagonal is consistently strongest for paper and weakest for trash, a pattern that mirrors the per-class F1-scores reported in Table 2 (0.73 for paper versus 0.46 for trash under Random Forest) and traces back to the same root cause: trash entered training with only 137 raw

images against 594 for paper, and even with augmentation applied to underrepresented classes, the category remained both the smallest and the most visually heterogeneous of the six. The off-diagonal mass in the trash row of each panel is therefore less a failure of any particular algorithm than a structural artifact of the underlying class distribution — one that quantifies and Figure 4 makes spatially legible by showing exactly where those misclassified trash images end up being routed instead.

It is also worth noting what does not vary much across the four panels glass, plastic, and metal show comparable, moderate confusion with one another in every classifier, Random Forest included. This consistency across otherwise quite different algorithms (kernel-based, instance-based, ensemble, and neural, per the family groupings in [Table 3]) suggests the confusion is rooted in the feature representation rather than in any one model's decision rule — color, texture, and shape descriptors alone may simply be insufficient to cleanly separate visually similar rigid materials, regardless of which classifier sits downstream of them.

4.4 What the Neural Network's Training Dynamics Add to the Story

The training and validation loss curves in Figure 5 add a temporal dimension that the single accuracy figure in [Table 1] and the static confusion snapshots cannot provide on their own. Both curves decline together through roughly the first thirty epochs, after which the validation curve begins to diverge from the training curve — the onset of mild overfitting that the early-stopping criterion (patience of 10 epochs, as specified in [Table 3]) eventually catches, but only after the gap has already opened. That divergence is informative precisely because it rules out the most obvious alternative explanation for the Neural Network's last-place finish this is not a case of an undertrained model that simply needed more epochs. The network is doing what it can with the 129-dimensional handcrafted input it was given in Figure 1, and the limiting factor is the information already lost during feature compression, not insufficient optimization.

This reframes the Neural Network's position in a more constructive light. The architecture itself, summarized (three hidden layers, dropout regularization, Adam optimization), is a perfectly conventional design; the generalization gap visible in Figure 5 is a symptom of

architecture-input mismatch rather than of the network being poorly specified. A convolutional architecture trained directly on raw TrashNet pixels would not face this constraint at all, since it would learn its own feature hierarchy instead of inheriting a 129-dimensional one built for classical classifiers — which is precisely why deep models trained end-to-end on this same dataset have been reported elsewhere to clear 90% accuracy (Aral et al., 2019), a figure dramatically higher than the 43.87% recorded for the fully connected network in this study's Table 1.

4.5 Implications for Dhaka and Study Limitations

Taken together, [Figures 1 through 5] and [Tables 1 through 3] sketch a coherent if modest picture: Random Forest is the strongest of the four classifiers evaluated, the trash category is the consistent weak point across every model, and the Neural Network's underperformance is a representation problem rather than a verdict on deep learning generally. None of this, however, closes the distance between a 66.60% laboratory benchmark and a deployable sorting system for Dhaka. The clean, single-item staging visible in [Figure 2] is not what a real collection point looks like, and a roughly one-in-three error rate — the practical reading of the Random Forest accuracy figure in [Table 1] — would be difficult to justify in a fully autonomous system but considerably more defensible in a human-assisted pipeline that routes only high-confidence predictions automatically and flags the rest for manual review.

Several limitations follow directly from the evidence reviewed above. First, the absence of formal significance testing means the gaps observed between bars in [Figure 3] should be read as practically, rather than statistically, meaningful. Second, the class-wise breakdown in (Table 2) and the confusion structure in (Figure 4) are reported only for Random Forest; a fuller picture would extend that same per-class analysis to all four classifiers. Third, and most fundamentally, every result discussed in this section — from the pipeline in (Figure 1) to the training curves in (Figure 5) — was generated on a dataset that does not reflect Dhaka's actual waste composition. Closing that gap with a locally curated, Dhaka-specific image dataset, ideally paired with a convolutional architecture trained end-to-end rather than on handcrafted descriptors, is the most direct next step suggested by the patterns documented.

5. Conclusion

This study examined which of four accessible machine learning algorithms—Support Vector Machine, Random Forest, K-Nearest Neighbors, and a shallow Neural Network—performs most effectively for image-based waste classification, with particular attention to implications for cities such as Dhaka, where automated sorting infrastructure remains underdeveloped. The findings, though not conclusive, were reasonably consistent. Random Forest, employing an ensemble of 200 decision trees trained on handcrafted color, texture, and shape features, achieved the highest classification accuracy at 66.60%. Notably, the Neural Network exhibited the weakest performance, a result attributable to the mismatch between fully connected architectures and compressed feature representations rather than any inherent limitation of deep learning approaches in this domain. Collectively, these results suggest that ensemble methods constitute the most robust approach for structured, feature-based waste classification under the conditions examined, while also highlighting the necessity of locally curated datasets and more sophisticated convolutional architectures to support real-world deployment. Although a single classification model cannot resolve Dhaka's broader waste management challenges, the development of improved automated sorting tools represents a meaningful and attainable contribution toward addressing this pressing urban sustainability issue.

Author Contribution

M.A.M. conceived and designed the study, performed image preprocessing and feature extraction, implemented and evaluated the SVM, Random Forest, KNN, and Neural Network models, analyzed the results, and wrote, reviewed, and approved the final manuscript.

Acknowledgement

The author M.A.M. would like to thank the creators of the TrashNet benchmark dataset for making it publicly available for research use. The author also acknowledges the institutional and computational support that enabled this work.

Competing Financial Interests

The author M.A.M. declares no competing financial interests.

References

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems (Version 2.11.0) [Software]. Google Brain.
- Aral, R. A., Keskin, S. R., Kaya, M., & Hacıomeroglu, M. (2019). Classification of TrashNet dataset based on deep learning models. In *Proceedings of the 2019 IEEE International Conference on Big Data* (pp. 2058–2062). IEEE.
- Bellman, R. (1961). *Adaptive control processes: A guided tour*. Princeton University Press.
- Bhuiyan, M. R. A., Mamur, H., & Begum, J. (2021). A brief review on renewable and sustainable energy resources in Bangladesh: Current landscape and energy transition. *Cleaner Engineering and Technology*, 4, 100208.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 679–698.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.
- Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.
- Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2), 179–187.
- Huda, A. S. N., & Mekhilef, S. (2014). Biomass energy in Bangladesh: Current status and prospects. *Renewable and Sustainable Energy Reviews*, 30, 504–517.
- Islam, M. R., & Rahman, M. M. (2019). A prospective assessment of biomass energy resources, potential, technologies and challenges in Bangladesh. *Renewable and Sustainable Energy Reviews*, 107, 217–225.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
- Masud, M., Ananno, A. A., Arefin, A., & Ahamed, R. (2019). Perspective of biomass energy conversion in Bangladesh. *Clean Technologies and Environmental Policy*, 21(5), 1009–1023.
- Ojala, T., Pietikäinen, M., & Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 971–987.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Thung, G., & Yang, M. (2016). Classification of trash for recyclability status [Dataset]. Hugging Face.