Name: Mohammed Ahmed Shalaby

MISIS: M01035318

Email: MS3579@live.mdx.ac.uk

Course: MSc Robotics

Module: PDE4433 — Machine Learning for Robotics

Assessment: Coursework 2 — Report

Professor: Dr. Maha Saadeh

Introduction

The increasing growth in worldwide garbage generation endangers both environmental sustainability and public health (Patil, De, Patil, & Kulkarni, 2024). Traditional garbage sorting systems are labor-intensive, subject to human error, and frequently inefficient, particularly in metropolitan locations with significant waste quantities (Olawade et al., 2024). To overcome these difficulties, there is an increasing demand for intelligent systems capable of automatically classifying and sorting garbage with high accuracy and efficiency (Lakhouit, 2025).

This project investigates the use of machine learning and robotics to develop an autonomous garbage categorization and sorting system. The system uses computer vision and deep learning to precisely classify waste items such as plastic, metal, paper, and glass, which are subsequently separated for recycling or suitable disposal. The ultimate objective is to improve waste management operations in smart cities, decrease environmental impact, and support circular economy initiatives.

This study is relevant beyond environmental management since it demonstrates how AI-driven automation may be applied to real-world sustainability concerns. Users engage with the system via a web-based interface by submitting garbage photos, which are subsequently identified and graphically displayed using feedback methods such as animations and result summaries. This interactive and intelligent method has the potential to help both homes and companies embrace more responsible waste habits.

To do this, the research makes use of the UCI RealWaste dataset, which has a broad collection of tagged rubbish photos. The EfficientNet-B0 model, noted for its balance of accuracy and computing economy, was chosen to serve as the classification system's core. This combination provides a solid foundation for creating a scalable and successful smart trash sorting solution.

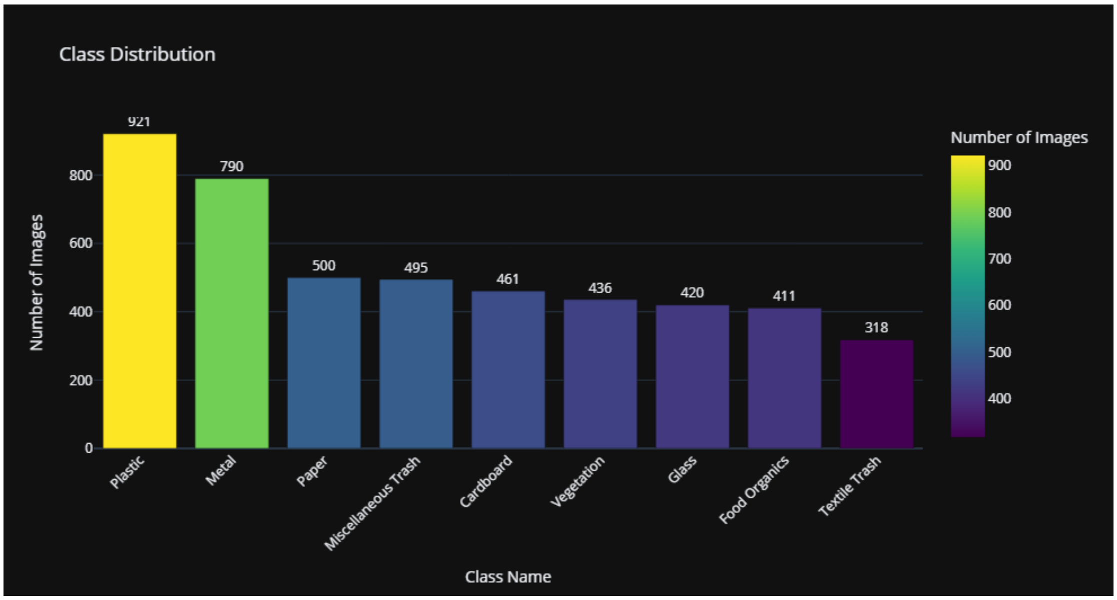
Dataset Description and Preprocessing

Dataset Overview

The RealWaste dataset, obtained from the UCI Machine Learning Repository, is a comprehensive collection of real-world garbage photos intended to train and evaluate machine learning models in waste categorization tasks. The collection comprises photos of different garbage products that have been classified depending on their material type. Each class represents a specific type of garbage, such as plastics, metals, paper, glass, and organic stuff. The photographs are collected in real-world settings, making them difficult and diverse enough to imitate realistic events. After importing the dataset, I discovered the following number of waste categories, which aids in understanding how balanced and unbalanced different classes are.

Data Preprocessing

Examining the information revealed that there was a class imbalance, with certain categories such plastic and metal having noticeably greater numbers of images than others, including cardboard and glass. This disparity may cause the model to be biased in favor of overrepresented classes, which would lower the accuracy of underrepresented categories.



To guarantee a strong training pipeline, I first established a data balancing procedure. The approach began with an examination of image dimensions to ensure uniformity throughout the collection. The results indicated that all images had the same proportions, which is perfect for consistent model training. Following that, the dataset was inspected for faulty or unreadable files, which luckily were not identified. To acquire an early grasp of the dataset's structure, I visually evaluated a sample of photographs from each class, ensuring that the labels matched the content.

During this research, it became clear that the dataset had a class imbalance, with certain categories, such as plastic and metal, containing much more photos than others, such as glass or cardboard. This increased the possibility of the algorithm becoming biased toward majority classes, potentially lowering classification accuracy for minority groups.

To address this issue, data augmentation techniques were used to enhance the minority classes. Transformations such as rotation, flipping, scaling, and color shifts were employed to artificially extend the dataset while maintaining the semantic integrity of each waste category. This not only improved class balance, but also increased the model's generalization potential by exposing it to a broader range of training data.

Class Name	Initial Count	Final Count
Cardboard	461	500
Food Organics	411	500
Glass	420	500
Metal	790	790
Miscellaneous Trash	495	500
Paper	500	500
Plastic	921	921
Textile Trash	318	500
Vegetation	436	500

Following preprocessing, the data was divided into training, validation, and test sets at a 70:20:10 ratio. This script constructed the proper directory structure, shuffled the images, and distributed them in the appropriate order, ensuring that the dataset is ready for successful training, assessment, and testing. The operation ended with a notification confirming a successful separation.

```
# Split function
def split_dataset(class_name, image_paths):
    random.shuffle(image_paths)
    total = len(image_paths)
    train_end = int(total * train_ratio)
    val_end = train_end + int(total * val_ratio)

    splits = {
        'train': image_paths[:train_end],
        'val': image_paths[train_end:val_end],
        'test': image_paths[val_end:]
    }

    for split, paths in splits.items():
        split_dir = os.path.join(output_base_dir, split, class_name)
        os.makedirs(split_dir, exist_ok=True)
        for img_path in paths:
            shutil.copy(img_path, split_dir)

for class_name in os.listdir(balanced_dataset_dir):
    class_dir = os.path.join(balanced_dataset_dir, class_name)
    if os.path.isdir(class_dir):
        images = [os.path.join(class_dir, img) for img in os.listdir(class_dir)
                  if img.lower().endswith(('.jpg', '.jpeg', '.png'))]
        split_dataset(class_name, images)

print("Dataset successfully split into train, validation, and test sets!")
```

```
Dataset successfully split into train, validation, and test sets!
```

Model Transformation, Setup and Evaluation

Model Development

To create an effective and computationally efficient trash categorization system, EfficientNet-B0 was chosen as the starting model. EfficientNet-B0 is known for its good balance of accuracy and resource efficiency, making it ideal for deployment in situations with limited computing capability, such as embedded devices or web applications, while yet providing cutting-edge performance.

The model was built with PyTorch, a popular deep learning framework that provides flexibility, straightforward debugging, and extensive community support. PyTorch's dynamic computation graph and flexible design made it an excellent candidate for combining transfer learning with bespoke training methods.

Transfer learning was used to speed up training and use existing knowledge by starting the model with pre-trained ImageNet weights. The final classification layer was changed to reflect the dataset's number of trash types. During training, the early layers were initially frozen to preserve core visual characteristics, while the later layers and classifier head were fine-tuned using preprocessed garbage data.

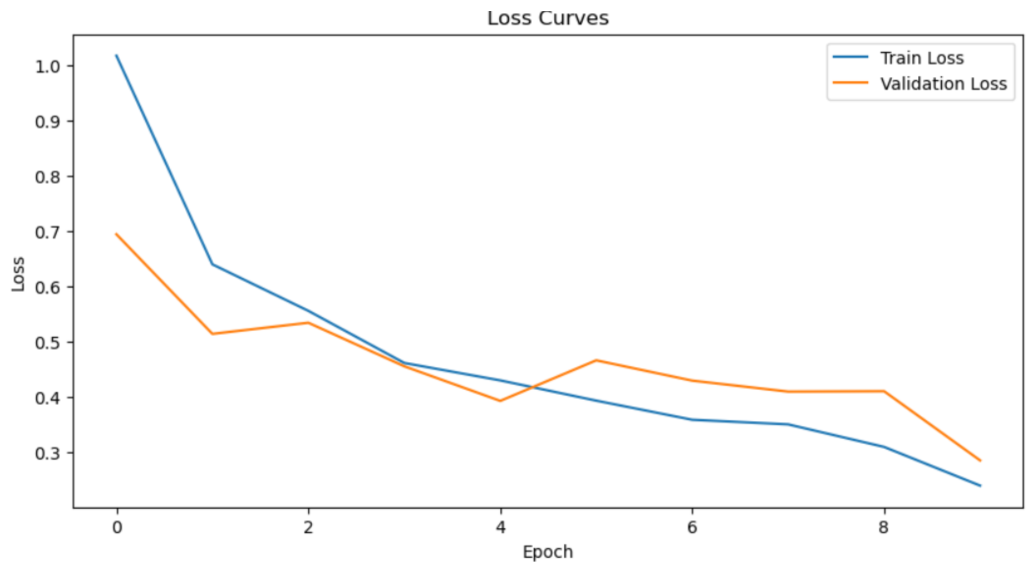
To achieve convergence, the training pipeline contained typical components such as a cross-entropy loss function, the Adam optimizer, and a learning rate scheduler. The dataset was enhanced and loaded using proprietary Dataset and DataLoader classes, allowing for efficient batch processing and modification within 10 epochs.

```
Using device: cpu
Number of classes: 9

Model Architecture:
Epoch [1/10] Train Loss: 1.0173 Train Acc: 65.56% Val Loss: 0.6942 Val Acc: 76.20%
Best model saved at Epoch 1 with Val Acc: 76.20%
Epoch [2/10] Train Loss: 0.6398 Train Acc: 78.26% Val Loss: 0.5139 Val Acc: 82.73%
Best model saved at Epoch 2 with Val Acc: 82.73%
Epoch [3/10] Train Loss: 0.5556 Train Acc: 81.22% Val Loss: 0.5341 Val Acc: 82.82%
Best model saved at Epoch 3 with Val Acc: 82.82%
Epoch [4/10] Train Loss: 0.4613 Train Acc: 84.21% Val Loss: 0.4553 Val Acc: 86.66%
Best model saved at Epoch 4 with Val Acc: 86.66%
Epoch [5/10] Train Loss: 0.4298 Train Acc: 85.55% Val Loss: 0.3927 Val Acc: 88.39%
Best model saved at Epoch 5 with Val Acc: 88.39%
Epoch [6/10] Train Loss: 0.3931 Train Acc: 86.24% Val Loss: 0.4661 Val Acc: 86.76%
Epoch [7/10] Train Loss: 0.3585 Train Acc: 87.30% Val Loss: 0.4293 Val Acc: 86.18%
Epoch [8/10] Train Loss: 0.3502 Train Acc: 88.65% Val Loss: 0.4095 Val Acc: 86.66%
Epoch [9/10] Train Loss: 0.3093 Train Acc: 90.07% Val Loss: 0.4101 Val Acc: 87.33%
Epoch [10/10] Train Loss: 0.2393 Train Acc: 91.77% Val Loss: 0.2850 Val Acc: 90.79%
Best model saved at Epoch 10 with Val Acc: 90.79%
```

Evaluation

The training results show a steady improvement in both training and validation accuracy during the 10 epochs, with validation accuracy increasing from 76.20% in epoch 1 to a peak of 90.79% in epoch 10. Similarly, the loss curves indicate a consistent drop in training loss and a general decreasing trend in validation loss, indicating good learning with minimal overfitting. The model obtained its highest validation accuracy during the last epoch, and when tested on the test set, it earned a remarkable test accuracy of 92.15%, indicating the trained model's good generalization and durability.




```
Testing the best model...
Test Accuracy: 92.15%
```

The model's performance was tested using important measures like as accuracy, precision, recall, and the F1-score. Following training, a detailed classification report was prepared to summarize these metrics, which were then displayed using Plotly to provide an interactive, class-wise perspective of the model's performance. This visualization was created in a Jupyter Notebook to facilitate exploratory investigation and then incorporated into the Flask web interface for smooth user interaction.

Classification Report

Class	Precision	Recall	F1-score	Support
Cardboard	0.92	0.92	0.92	50
Food Organics	1	0.94	0.9691	50
Glass	0.8889	0.96	0.9231	50
Metal	0.9036	0.9494	0.9259	79
Miscellaneous Trash	0.8269	0.86	0.8431	50
Paper	0.9773	0.86	0.9149	50
Plastic	0.9121	0.8925	0.9022	93
Textile Trash	0.9423	0.98	0.9608	50
Vegetation	0.9592	0.94	0.9495	50
macro avg	0.9256	0.9224	0.9232	522
weighted avg	0.9233	0.9215	0.9216	522

The confusion matrix demonstrates high performance in various categories. Plastic, metal, food organics, textile garbage, plants, and glass were identified with remarkable precision, as seen by the large numbers along the diagonal. There was some misunderstanding between cardboard and paper. Similarly, a small number of plant samples were incorrectly labeled as miscellaneous rubbish.

Confusion Matrix

Predicted Label

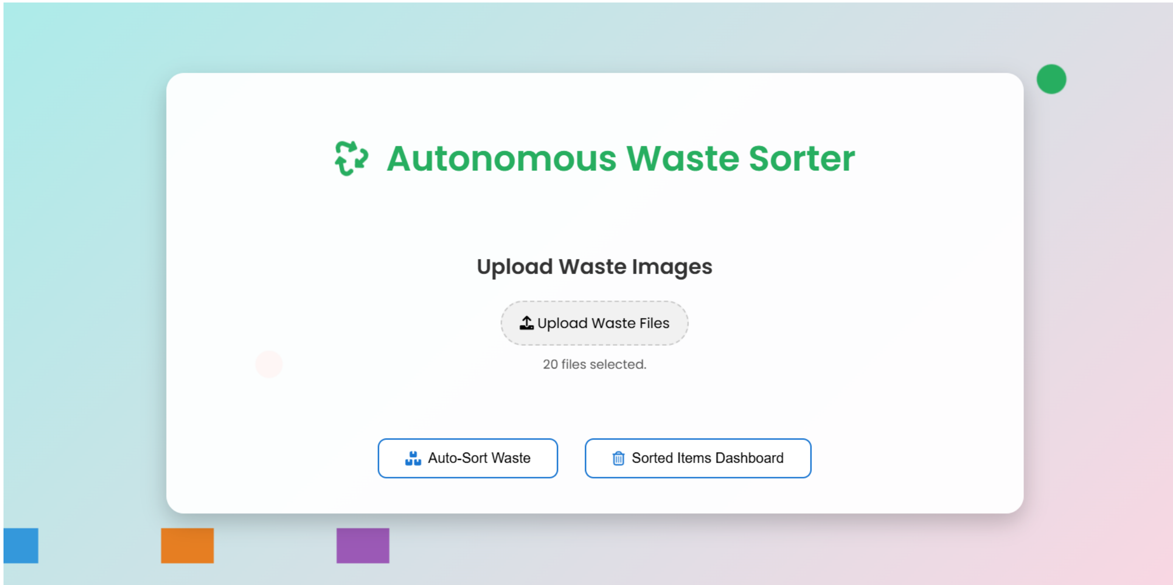
True Label	Cardboard	Food Organics	Glass	Metal	Miscellaneous Trash	Paper	Plastic	Textile Trash	Vegetation
Vegetation	0	0	0	0	3	0	0	0	47
Textile Trash	0	0	0	0	1	0	0	49	0
Plastic	0	0	4	5	1	0	83	0	0
Paper	4	0	0	1	2	43	0	0	0
Miscellaneous Trash	0	0	1	1	43	0	3	1	1
Metal	0	0	0	75	0	0	4	0	0
Glass	0	0	48	1	0	0	1	0	0
Food Organics	0	47	0	0	1	0	0	1	1
Cardboard	46	0	1	0	1	1	0	1	0

To evaluate the model's performance using previously unknown real-world data, I created a final prediction pipeline with 10 randomly picked photos from the original dataset. This stage entailed loading the best-performing EfficientNet-B0 model, implementing relevant preprocessing operations (resizing, normalization, and tensor conversion), and making predictions for each picture. The findings were shown using annotated outputs, which included both the actual and predicted labels. The model showed good generalization, reliably predicting the proper class in most circumstances. This last test confirms the model's preparedness for implementation in a real-world waste-sorting application and highlights its dependability when subjected to a variety of input situations.

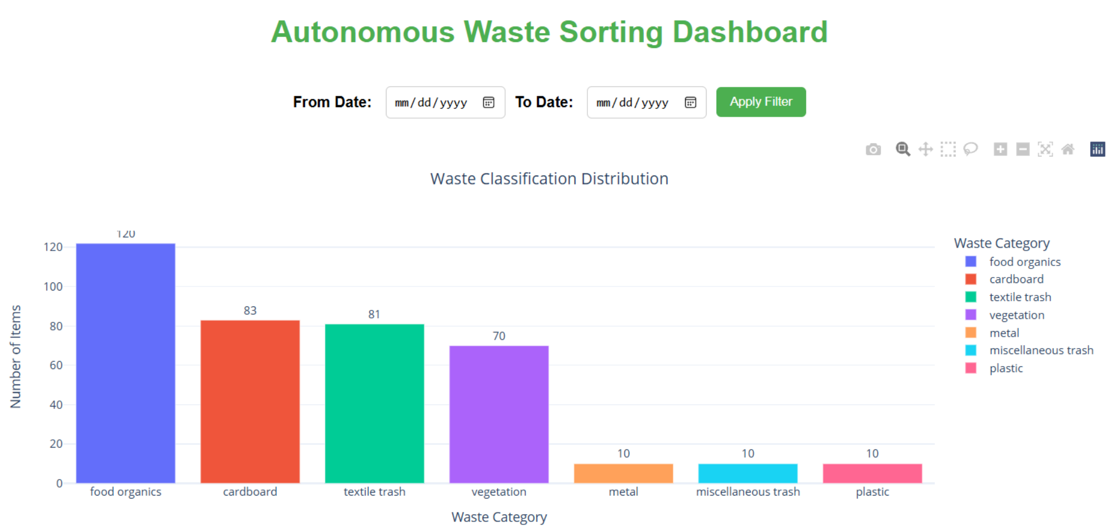
Autonomous Sorting Model Deployment

To highlight the practical potential of automated waste classification, an end-to-end web-based application was constructed and deployed using Flask. This deployment uses the fine-tuned EfficientNet-B0 convolutional neural network that was pre-trained on ImageNet and then trained on a custom waste classification dataset that included nine waste categories: cardboard, food organics, glass, metal, miscellaneous trash, paper, plastic, textile trash, and vegetation.

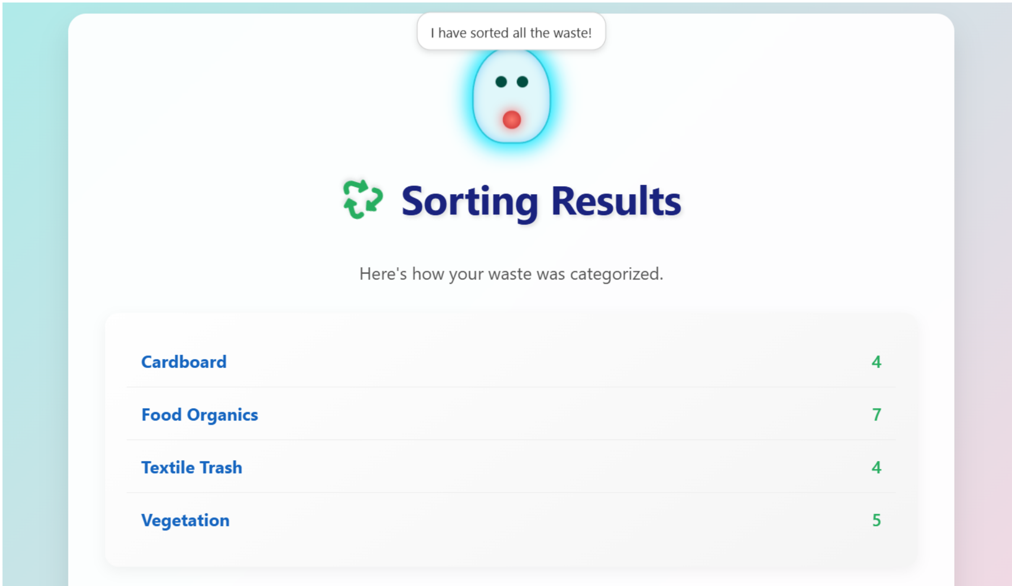
The implemented approach allows customers to upload several trash pictures at once via a simple online interface. Each image is preprocessed before being sent into the trained model, which categorizes the garbage accordingly. Images are saved in category folders with timestamped filenames to aid with traceability and future analysis.



A custom dashboard module was added to the program to show the distribution of categorized garbage. Using Plotly, an interactive bar chart displays the counts by category, and users may refine the results by date range. This graphic helps to monitor trends, audit categorization performance, and assess environmental sorting behaviors.



To increase engagement and input, a results page with graphic summaries was created. A robot assistance animation demonstrates effective sorting, making the interface more friendly. Furthermore, modest canvas-based animations simulate the action of goods dropping into color-coded bins, emphasizing the notion of self-sorting in a gamified and instructional way.



Importantly, the high validation accuracy (90.79%) transferred into real-world performance, as the deployed model demonstrated great predictive consistency throughout practical testing. While the present deployment is geared for local usage, future scalability (e.g., cloud hosting) and basic security mechanisms (e.g., file type validation and input sanitization) have been addressed to assure production-level resilience.

Conclusion and Future Work

This project, which combines an EfficientNet-B0 model with an intuitive Flask web application, demonstrates the potential of AI and robotics in automated garbage sorting. With a test accuracy of 92.15% and a validation accuracy of 90.79%, the system provides dependable real-world performance that is improved by the seamless interface design and visual feedback. It provides a scalable solution for more intelligent environmental management by successfully bridging the training to implementation gap. Future developments in robotic automation, IoT integration, real-time video-based detection, and larger datasets may transform the system into a completely flexible, end-to-end waste management platform, opening the door for intelligent, sustainable urban solutions.

References

Lakhouit, A., 2025. Revolutionizing Urban Solid Waste Management with AI and IoT: A review of smart solutions for waste collection, sorting, and recycling. Results in Engineering, p.104018.

Olawade, D.B., Fapohunda, O., Wada, O.Z., Usman, S.O., Ige, A.O., Ajisafe, O. and Oladapo, B.I., 2024. Smart waste management: A paradigm shift enabled by artificial intelligence. Waste Management Bulletin.

Patil, K.D., De, J., Patil, V.K. and Kulkarni, M.M., 2024. Environmental Effects and Threats of Waste: Understanding Threats and Challenges to Ecosystem, Health, and Sustainability and Mitigation Strategies. In From Waste to Wealth (pp. 37-69). Singapore: Springer Nature Singapore.