IDUMP IMAGE CLASSIFICATION REPORT

1 BUSINESS UNDERSTANDING

1.1 Business Overview

Konza City Technopolis is a burgeoning smart city in Kenya, designed to be a hub of technology, innovation, and sustainability. As part of its commitment to environmental sustainability, Konza City aims to implement advanced waste management solutions. iDump, a waste management company, has been contracted to develop and deploy an efficient waste segregation system that separates waste into different categories for recycling. This initiative will play a crucial role in maintaining the city's cleanliness, reducing environmental impact, and promoting recycling practices.

Solid waste encompasses all materials that result from human and societal activities but have lost their utility or desirability. This waste may be items belonging to three primary groups:
Recyclable Inorganics: Fit for repurposing, e.g., plastics, metals.
Divertible Organics: From which energy and fertilizer can be derived, e.g., food and vegetation.
Inorganic Materials: Requiring Landfill e.g., textiles, treated wood.

Improper management of waste disposal poses significant risks to environmental sustainability and human health, resulting from toxic wastewater byproducts and the global warming potential of methane in landfill gas. In 2020, 3.48% of global greenhouse gas production was attributed to the waste disposal sector, with methane from landfill gases accounting for 20% of worldwide methane emissions the following year. The chemical composition and production of both wastewater and landfill gases heavily depend on the organic content within the waste processed, with significant variations arising in the presence of food and vegetation materials.

Detection of the waste types under meticulously refined material classes in both an accurate and timely manner is essential for sustainable waste management, ensuring accountability for both seasonal variations and recycling uptake.

1.2 Problem Statement

Konza City Technopolis faces significant challenges in managing the diverse and large volume of waste generated by its residents and businesses, projected to peak at 240,000 inhabitants. Traditional waste management methods, reliant on manual sorting and segregation, have proven to be inefficient, labor-intensive, and unable to cope with the scale of waste produced. This inefficiency has led to increased environmental pollution, with a substantial portion of waste being improperly disposed of in landfills or open dumps, contributing to soil contamination, water pollution, and increased greenhouse gas emissions. Furthermore, the lack of effective segregation and recycling facilities hampers the city's ability to recycle valuable materials, leading to significant loss of resources and economic opportunities.

1.3 Proposed Solution

To address these challenges, we propose developing and deploying a robust image recognition system utilizing machine learning algorithms. The system will be trained on a diverse dataset of waste images and evaluated based on its accuracy and F1 score.

1.4 Metrics of Success

The metrics that we considered were accuracy and f1 score. Accuracy was used as it gives a baseline understanding of the dataset. While an F1 score accounts for the slight class imbalance that we experienced. Due to the problem being a multi class classification problem, we also relied on a confusion matrix to see the discrepancies within the classes.
With the computational capabilities currently in use we set the metrics as below:

Model Accuracy:	We aim for at least 80% accuracy in waste classification.
F1 Score:	We've set a target F1 score of 80% or higher to balance precision and recall.

1.5 Objectives

Main Objective
Develop and deploy a robust image recognition system to enhance waste management efficiency in Konza City.

Specific Objectives:

i. Gather and preprocess a diverse dataset of waste images to ensure robust model training, including various types and conditions of waste items.
ii. Analyze and report the most common types of waste collected.
iii. Accurately classify waste into their distinct material types and categorize them into recyclable inorganics, divertible organics, and non-recyclable inorganic materials to improve waste management.
iv. Deploy a user-friendly interface that allows iDump personnel/equipment to upload images and receive real-time classification results from the image recognition model.

1.6 Constraints

Diverse Waste Composition: The waste generated in Konza City includes a wide variety of materials, making it challenging to accurately classify each type.
Data Quality: Ensuring the dataset used for training the model is comprehensive and representative of real-world waste conditions.
Scalability: The system must be able to handle large volumes of waste efficiently, potentially requiring integration with a conveyor belt systems for real-time processing.
Computational Resources: Limited computational resources for model training and evaluation.

2. DATA UNDERSTANDING

The images used for this project were sourced from a Kaggle dataset by Joakim Arvidsson([1]) titled RealWaste Image Classification. It is a dataset of 8 major material types commonly collected within a landfill environment.

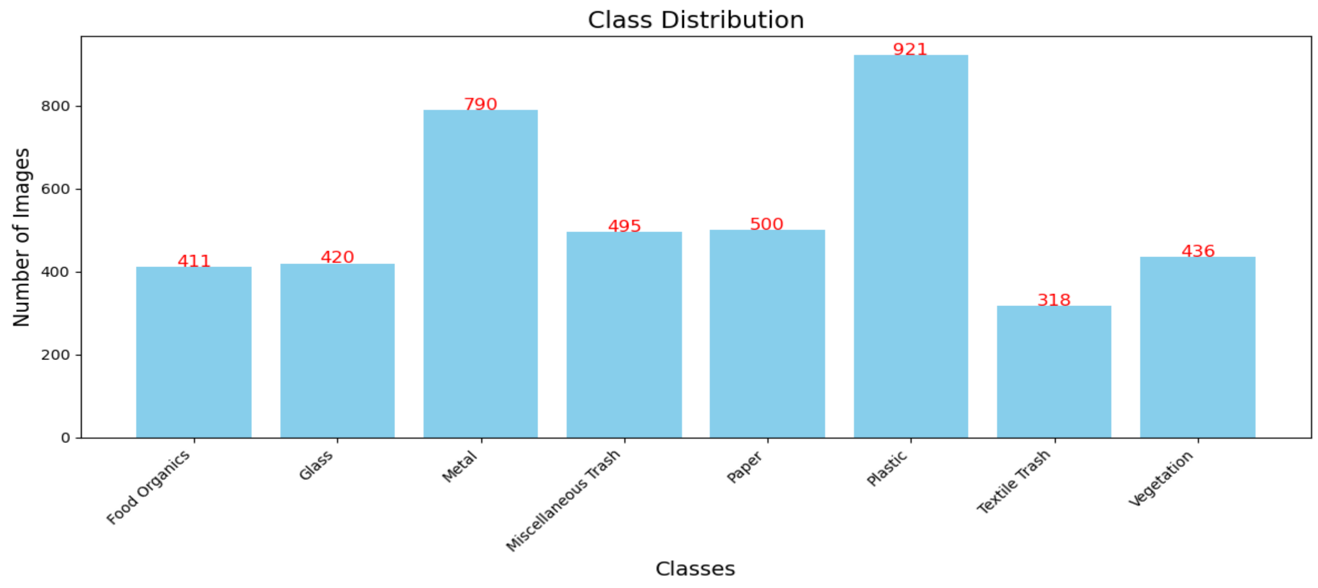
The data which was saved in a folder "RealWaste" had 4291 image that can be further classified into the following classes, Food Organics, Glass, Metal, Miscellaneous Trash, Paper, Plastic, Textile Trash, and Vegetation. The images all have a image resolution of 524x524 and visual inspection of 3 random images from each class was plotted to get a better understanding of the images. These images were pickled to be used for deployment

Further data understanding showed that there were no corrupted images by using the pillow(PIL) library to verify if the image can open and displaying an error message if it can't open. Inspection for duplicated images was also carried out by leveraging perpetual hashing where each image creates a hash value. If images are similar, the hash value will be similar regardless of resolution.

3. DATA PREPARATION

In preparation for modeling to be carried out, the following steps were taken to prepare the images.

The initial step was to consider any class imbalance that may skew the data by causing the model to favor one class over the others. This lead to a countplot of the classes and it was evident that plastic and metal class greatly outnumbered the rest of the classes and would need to be considered during modeling.



The data was split into 3 folders of train, validation and test data. In each of these files, the 8 classes of images were represented in a ratio of 70% train data, 20% validation data and 10% test data. Splitfolder Library was used to ensure the ratio was kept and a seed was included to ensure the slit could be replicated. These new split data was saved in a folder "RealWaste2".

4. DATA AUGMENTATION AND MODELING

While data augmentation can be considered part of data preparation, this step varied based on the models built and each underwent different augmentations based on the modeling method used.

We utilized Convolutional Neural Networks (CNNs) models as they are a class of deep learning models specifically designed for analyzing visual data. CNNs are characterized by their use of convolutional layers, which apply convolution operations to the input data to extract features.

Below are some of the models built for these projects:

i. **Baseline CNN model**
The first model built was a baseline CNN model which would act as a guide for future models. This model only underwent a rescaling as this increases the stability in training as well as normalizing the images, and speeds up the training process. A standard batch size of 32 was used and class mode was set to categorical since we are predicting for a multi class classification problem.
The model was built on the keras sequential API and had 3 convolutional layers, each with a relu activation and a pool size of 2X2. The final layers were flattened and fully connected with an activation of softmax since we are dealing with 8 classes as are all other subsequent models.
The model was compiled with an adam optimizer and a loss function set at categorical crossentropy since the target labels are one hot encoded. The metric used was accuracy which is in line with our metric of choice. The model was then fit while running 10 epochs.
We were able to achieve an accuracy of 53% on the test images and an f1 score of 53% as well.

ii. **Baseline CNN model with Augmentation**
The baseline model underwent some image augmentation to better improve the results. Zoom range, shear range, rotation range, horizontal flip, width shift range and height shift range were all introduced to the train image datagenerator. This was to assist the model to train on data that had undergone different transformations with the assumption that it shall generalize better on unseen data. This augmentation was only carried out on the train data.
The rest of the model was kept the same and the models results improved to 57% accuracy and f1 score. This however did not show promise and so pre trained CNN models were considered to see if the accuracy improved.

iii. **Pre-trained model using InceptionV3 architecture**
Due to the suboptimal performance of our baseline models with custom convolution layers, which failed to meet our predefined success thresholds. Inception V3 from the Inception family was chosen for its proven performance record and computer efficiency. It has 48 layers including convolutional layers, pooling layers, fully connected layers, and softmax layers.
The train images were rescaled and image augmentation was performed which included Zoom range, shear range, rotation range, horizontal flip, width shift range and height shift range augmentations. The InceptionV3 model was called from ImageNet as the base model with the base layers frozen and the custom layers were added. Early stopping and ReduceLRonPlateau were used as callback methods on the model to ensure the model did not over fit.
The model was then fit and then evaluated for the test accuracy. Plots for accuracy and loss function were plotted for comparison. It was then saved for future use. This model achieved a substantial improvement over the custom baseline and augmented CNN models with an accuracy of 76% with a weighted f1 score of 74%. This however, did not yet achieve our threshold.

iv. **Pre-trained model using DenseNet121 architecture**

A DenseNet121 model from the DenseNet family was then used to try and improve the overall accuracy of the models. It has a total of 121 layers and has improved efficiency, performance and east of training models. The model utilized a standard batch size of 32 and target size of (224, 224) which was a similar to the pre trained model images used. The model was fit and trained on 10 epochs and used an adam optimizer.

The model was then fit and then evaluated for the test accuracy. Plots for accuracy and loss function were plotted for comparison. It was saved into a h5 file for future use. The DenseNet121 model demonstrated stable convergence and improved accuracy over the training epochs while achieving an accuracy of 79% and a weighted f1 score of 78%.

This again however falls short of our target 80% accuracy threshold and exhibits class imbalance in minority classes, which leads us to try out more efficient architectures.

v. **Pre-trained model using EfficientNetV2B0 architecture**

This model from the EfficientNet family was used as it achieves high accuracy with efficient use of parameters and computational performance. This version has 237 layers.

In the image generator, a preprocessing function from the effecientnetV2 preprocess input was used as it is essential to ensure the images are preprocessed in a way similar to how the model was trained. If this is not used, it could greatly degrade the performance of the model. The validation imagegenerator and test imagegenerator underwent the same process.

The train generator also had other parameter augmented, i.e. rotation range, zoom range, shear range etc…A target size of (224,224) was used as this is the ideal size for this pre-trained model and was used for train, test and validation data. A batch size of 32 was used for efficiency in modeling considering computational capabilities.

The model also took into account class weights to account for any class imbalance to try and improve the performance of the model. A modelcheckpoint was used to save the best model during training while using the validation accuracy to determine the best model to save. ReduceLROnPlateau callback is also used to monitor plateauing in the validation accuracy for two epochs after which the learning rate will be reduced determined by the minimum delta.

The model achieved an accuracy of 83% on the test images with a weighted f1 score of 83%. We noted that most classes generalized well except plastics which largely classified as metal.

vi. **Pre-trained model using ResNet50 architecture**

For this model an architecture from the ResNet family was utilized. It is best used for applications that require very deep networks and robust performance and are suitable for large scale image classification problems. This version used has a total of 50 layers.

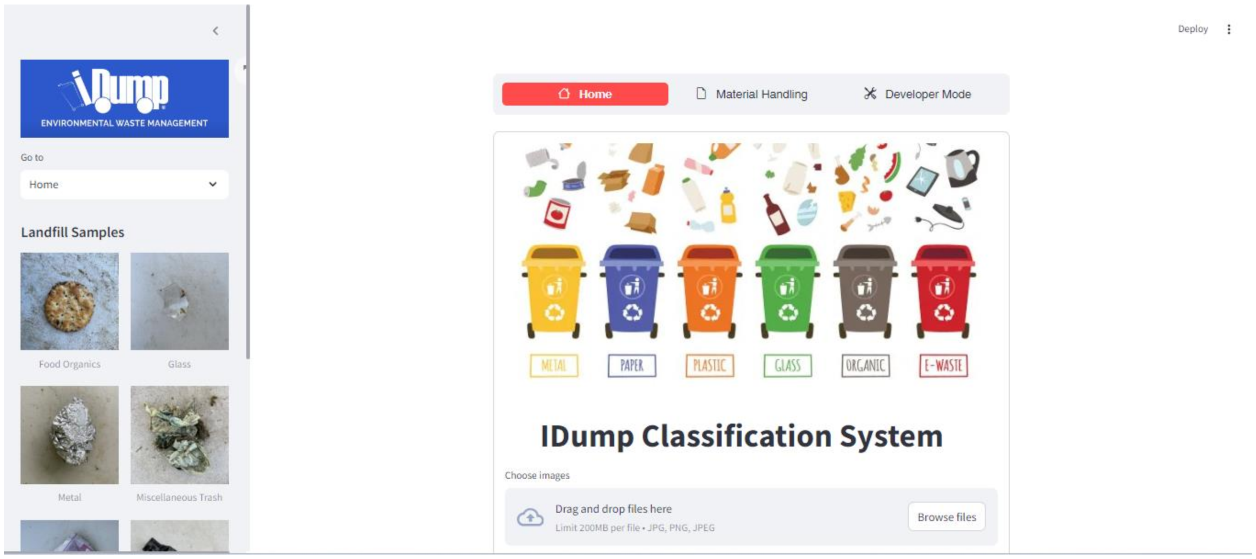
The model underwent similar augmentation to the EffecientNet model however the appropriate preprocessing function was used i.e. resnet50 preprocess input. A similar target size of 224x224 was used as well as a similar batch size of 32.All train image generator augmentation were also kept same as the EffecientNet model. The model had its class weights balanced and callbacks were similar.

After fitting the model, the model had an accuracy of 85% on its test data with a weighted f1 score of 85% as well. This was the best model we were able to achieve and was later deployed in our app.

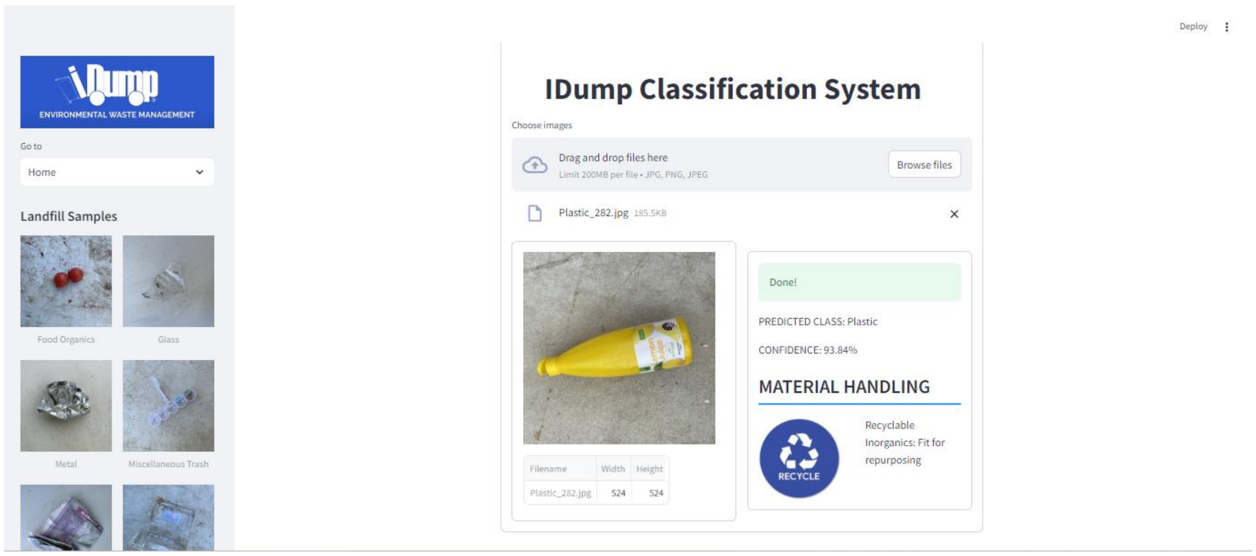
5. DEPLOYMENT

The model was pickled into a file that was then used to deploy the model. Streamlit which is an open source python library was used to create a custom web app for use in the classification problem. This is a great and simple way to integrate images either saved on your computer or from a camera and there after producing an interpretable result.

Below we can see the home page of the web app before any processes have been performed.



The app is able to correctly classify majority of the images into their correct classes as below as well as give the procedures for material handling:



Plastic_282.jpg 185.5KB

Filename	Width	Height
Plastic_282.jpg	524	524

Done!

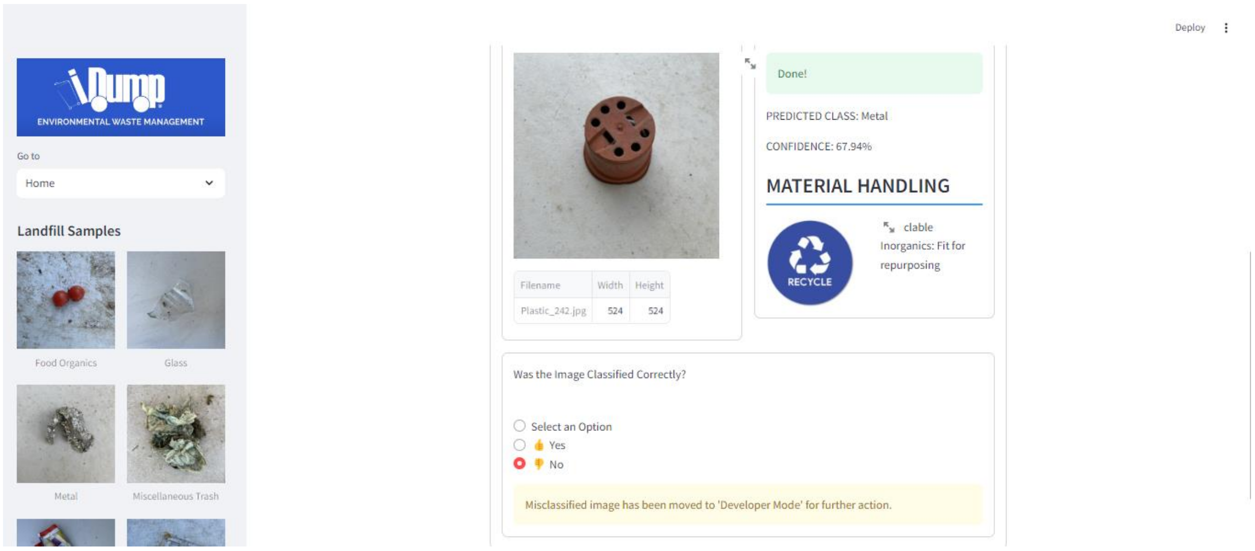
PREDICTED CLASS: Plastic

CONFIDENCE: 93.84%

MATERIAL HANDLING

Recyclable Inorganics: Fit for repurposing

However for cases where it incorrectly classifies the material as below, we have incorporated a means to of acknowledging if the model was correctly classified as below, if the materials image is incorrectly classified, it is then moved to the developers sections


Deploy

Go to

Home

Landfill Samples

Filename	Width	Height
Plastic_242.jpg	524	524

Done!

PREDICTED CLASS: Metal

CONFIDENCE: 67.94%

MATERIAL HANDLING

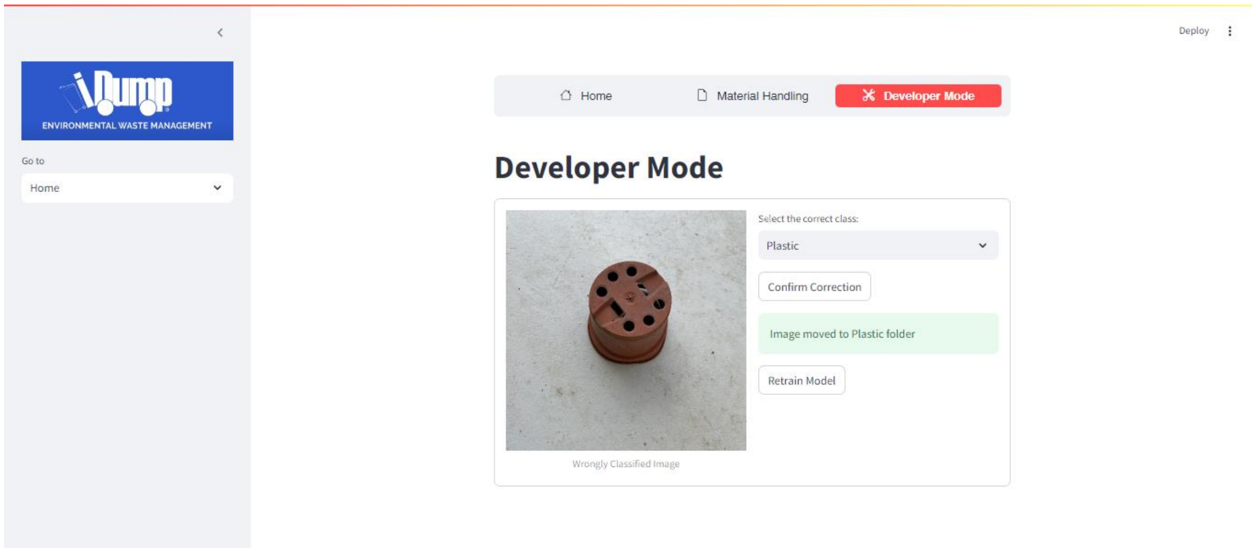
clable
Inorganics: Fit for repurposing

Was the Image Classified Correctly?

Select an Option
👍 Yes
[✓ 👎 No]

Misclassified image has been moved to 'Developer Mode' for further action.

In the developers section, the image is classified to its correct class and the image is added to the train data of the correct class and the model is retrained with the new image. Hence continuously updating out models efficiency.


Deploy

Go to

Home

Developer Mode

Select the correct class:

Plastic

Confirm Correction

Image moved to Plastic folder

Retrain Model

Wrongly Classified Image

6. CONCLUSIONS

The models utilized showed a thought out iterative approach to improvement of models to the desired outcome. While multiple other models were used with different parameters, the ResNet50 model demonstrated the highest performance. This highlights the advantages that pre trained models have in image recognition.

With this we can conclude that we achieved our desired metrics and improved on them. For further clarity on models used, more information shall be in the appendix.

7. RECOMMENDATIONS

There is room for improvement on the model to achieve even higher accuracy levels. As such we would recommend:

i. Outsourcing compute to cloud servers services with GPUs
ii. Utilizing models of higher accuracy e.g. ResNet152, Inceptionv4, DenseNet201, EfficientNetB7
iii. Implementing more model monitoring and management tools
iv. Automating Model Retraining Pipelines
v. Expanding dataset Diversity

APPENDIX

1. https://www.kaggle.com/datasets/joebeachcapital/realwaste - DataSet used for the project
2. https://keras.io/api/applications/inceptionv3/- InceptionNet information on Keras
3. https://keras.io/api/applications/densenet/- DenseNet information on Keras
4. https://keras.io/api/applications/efficientnet/ - EffecientNet information on Keras
5. https://keras.io/api/applications/resnet/- Resnet Information on Keras