Project Report
In
Deep Learning (CSI_7_DEL)

By
Adeoluwa Oyinlola
(4214719)

MSc. Applied Artificial Intelligence

Smart Garbage Classifier: A Neural Network Approach

January 3, 2024

Contents

1 Executive Summary

This research project focuses on addressing the urgent issue of a working garbage classification system through the adoption of Neural Networks. Automating the sorting and classification process for sustainable waste management becomes crucial and extremely of great importance as the global waste and garbage disposal crisis keep increasing. The neural networks which were trained using a wide range of datasets of garbage images, clearly demonstrates a remarkable progress in systemic identification and classification; thereby, making waste processing much more efficient and easy to manage.

The aim of this project is to improve the efficiency of waste management systems by building a smart AI solution for sorting garbage based on its type. This will definitely reduce human error and optimise resource distribution and allocation. To actualize this goal, two neural networks were carefully trained and supervised using carefully chosen hyper-parameters and necessary data augmentation techniques considering the rather unique problems posed by garbage identification. The dataset was gotten from different environments to ensure the model adapt to real-world situations.

The CNN model for garbage classification shows encouraging results compared to the ANN approach using binary classification (One-vs-Rest) model even beyond some well known CNN models like RecycleNet, MobileNet and ResNet-50. With more time for testing and fine-tuning, it has the potential to revolutionise garbage classification accuracy. The deployment stage strives to seamlessly incorporate the model into existing waste management infrastructures, focusing on practical application for real-world use.

As it will be shown in this report, this project contributes to the field of deep learning and computer vision but additionally, it contributes immensely to environmental sustainability. The CNN model stands as a promising innovation for improving waste management processes by automating and improving the accuracy of classification; thereby, reducing the environmental consequence of mismanaged waste.

2 Introduction

The increasing global waste crisis has fundamentally called for an innovative and strategic approach to tackling waste management, with garbage identification and classification at the epicenter (*Gupta et al. 2022*). The issue of waste disposal and mismanagement has not only resulted to environmental degradation but has also been a burden for several other existing waste management systems in operation (*Mao et al. 2022*). As a result, this research project identifies a need to use deep learning method, particularly a Deep Neural Network (**DNN**) approach, to automate and enhance garbage classification for an healthy and sustainable environment. According to *www.afrik21.africa 2023*, the image below shows the problem of waste management which has been of great concern particularly in the sub-Saharan Africa.

Figure 1: Poor waste management

The environmental advantage of accurate and timely garbage classification cannot be overemphasized (*Majchrowska et al. 2022*). A waste sorting system that is inefficient would not only disrupt recycling but also results into increased landfill usage, pollution, and wastage (*Chen et al. 2022*).

The primary objective of this deep learning project is to radically revolutionize waste management by developing an automated system that can intelligently categorize different types of garbage almost accurately. The process is in line with the broad mission of optimizing resource allocation, saving the planet and promoting sustainable waste management practices.

2.1 Project's Approach

The approach adopted involves training two neural networks: a convolutional neural network (CNN) and an artificial neural network (ANN) using One-vs-All binary classification approach. It is a technique of transforming a multi-class classification problem into multiple binary classification problems on a comprehensive dataset that cuts across all sort of garbage in different context: cardboard, glass, paper, metal, plastic and trash (*Putra et al. 2023*). The dataset imitates the real-world complexity of waste in different environment. One important advantage of the CNN architecture over every other neural networks is that it is intelligent enough to learn and extract the features from images automatically (*G. Rishma and R. Aarthi 2022*). The model architectures were fine-tuned through a number of iterative processes to ensure that they adapt to the intricacies of garbage imagery. The image below shows a simple neuron which is also known as a perceptron where x_i are the input features which get multiplied by their individual weights, w_i. The weighted sum is then passed through an activation function which then computes the probability of an output, y.

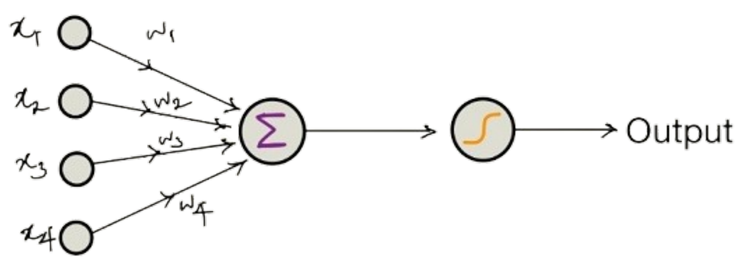


Figure 2: A Simple Perceptron

2.2 Business Value

The business implications of the project extend beyond technological advancements, encompassing multifaceted benefits for the industry. The automation of garbage classification holds the potential to yield advantages such as heightened environmental responsibility, reduced costs, and enhanced operational efficiency. The augmented accuracy in garbage sorting is poised to be advantageous for diverse stakeholders, ranging from waste management firms, municipalities, and environmental authorities to educational institutions and households. The cumulative effect of these benefits is anticipated to manifest in elevated recycling rates and a consequential reduction in environmental damage. This project thus represents a significant stride towards fostering sustainability and efficiency within the waste management sector.

2.3 Structure of Report

The subsequent phases of the project are meticulously structured to emphasize critical aspects, encompassing business comprehension, data preparation, understanding, modeling, evaluation, and eventual deployment. Each phase contributes synergistically towards the overarching objective of crafting a dependable and functional waste classification system. This comprehensive framework not only aspires to make substantial strides in the field of deep learning but also seeks to deliver tangible, enduring benefits to the waste management sector and the broader environmental landscape. Through this systematic approach, the project aims to make a meaningful contribution to the integration of advanced technologies for sustainable waste management practices.

3 Business Understanding

The operations of an effective garbage classification is within the context of escalating environmental challenges propelled by rapid urbanization and population growth index. The short-comings of traditional waste management systems, coupled with enormous amount of waste generated, undermines the urgency to adopt innovative solutions to tackle these problems (*Ahmed et al. 2023*). Garbage identification and classification, when automated through AI technologies in deep learning, has the potential to significantly impact environmental sustainability and waste management (*Mishra et al. 2022*).

3.1 Business Benefits

1. Environmental Concerns: The importance of accurate garbage classification comes from the environmental consequences of inefficient waste sorting methods. Mismanaged waste has resulted to pollution, resource depletion, and an over-reliance on landfills (*Abubakar et al. 2022*). For instance, solid wastes poses severe risks to ecosystems and marine life. The project's core objective is to address these concerns by developing an efficient and working neural network model to accurately identify and classify various kinds of garbage, thereby encouraging more effective wast management.

2. Potential Business and Societal Impacts: There are significant financial and societal benefits to using an automated waste classification system in addition to environmental ones. From a corporate perspective, the system provides streamlined garbage sorting procedures that offer operational efficiency and may result in cost savings. In terms of society, the project contributes to cleaner environments, fewer health hazards related to poor garbage managements, and proper awareness of appropriate waste disposal techniques.

3. Stakeholders and their Interest: The project success depends on knowing the stakeholders and their interests:

   - Waste Management Companies: They are committed to operational efficiency, reduction in cost, and keeping up with regulations (*Putra et al. 2023*). An accurate system of garbage classification can improve their service and market competitiveness.
   - Environmental Agencies: Environmental agencies and entities inherently concerned with the environmental ramifications of waste management,

find resonance in the objectives of the project. The alignment between the project and their advocacy for environmental sustainability practices positions it as a pertinent contribution to their overarching goals.

- Technology Providers: Technology providers stand poised for significant market growth and heightened demand for advanced technologies through the successful implementation of an automated garbage classification model. The efficacy of such a model in accurately categorizing waste materials holds the potential to catalyze an expansion in the market, fostering an increased appetite for cutting-edge technological solutions (*Jin et al. 2023*).
- Educational Institutions: Educational institutions stand to gain significant advantages through the implementation of automated garbage sorting systems. A conceptual prototype involves depositing waste into the sorting system, wherein an automated scan is initiated to classify the waste into distinct categories. Subsequently, the system systematically directs each item to its appropriate container, effectively segregating recyclable from non-recyclable waste.
- Household usage: The implementation of automated waste sorting systems in households presents a compelling business proposition. Particularly targeting homes, this system offers an innovative solution to the ubiquitous challenge of waste management, especially in environments where children contribute to the generation of diverse waste materials.

3.2 Business Objectives

The objectives of the garbage classification project are multi-dimensional:

1. Operational Efficiency: Making waste sorting processes easier and faster, minimizing labour and related costs.

2. Cost Reduction: Minimizing costs associated with waste mismanagement, disposal, and inefficient sorting. The figure below shows the projected waste generation according to *datatopics.worldbank.org 2023.*

3. Corporate Social Responsibility (CSR): The project is in fulfillment of CSR commitments by contributing to environmental sustainability and waste reduction.

4. Market Competitiveness: It solidifies the firm's standing in the waste management industry through innovative and technology-driven solutions.

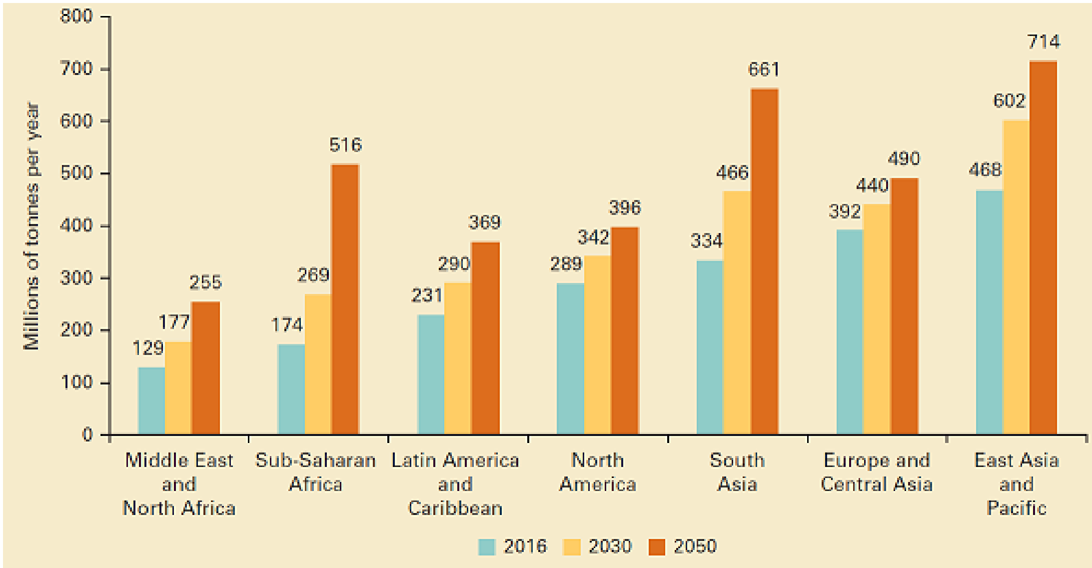


Figure 3: Projected waste generation, by region (millions of tonnes/year)

3.3 Assessment of the Current Situation

A brief assessment of the current situation illustrates negative environmental effects, a growing dependency on landfills, and inefficient garbage sorting are the hallmarks of the existing waste management systems (*Bhattacharya et al. 2023*). The process of manual sorting methods have limited scalability, requires a lot of resources, and are prone to errors. These drawbacks emphasise the requirement for an automated waste classification system that can handle the difficulties presented by the current situation of waste management today.

An analysis of the current situation of garbage classification demonstrates an urgent need for an accurate and efficient waste classification. In the CNN model developed by *Sidharth et al. 2020* with 5 convolution layers and 100 epochs, a test accuracy of 76% was achieved. ResNet-50 by *Adedeji and Z. Wang 2019* achieved an accuracy of 87% using 50 convolution layers and 12 epochs. *Bircanoğlu et al. 2018* in incorporating RecycleNet used 121 convolution layers and ran 200 epochs to achieve 81% test accuracy. Using VGG16, the MobileNet by *H. Wang 2020* achieved 84% accuracy. With 159 convolution layers and 18 epochs, the CompostNet by *Frost et al. 2019* achieved 77.3% test accuracy. The table below shows the current situation of waste and garbage classification.

Method	Layer	Epochs	Accuracy
CNN Model	5	100	76%
ResNet-50	50	12	87%
RecycleNet	121	200	81%
MobileNet	28	210	84%
CompostNet	159	18	77.3%

Table 1: Accuracy of different models with their hyperparamters

3.4 Project Plan

1. Data Collection and Understanding: A diverse range of datasets that cuts across different garbage types and scenarios was gathered. Thorough checks and understanding of the dataset were carried out.

2. Data Preparation: To make the dataset fit for modelling and equal representation of each class, proper techniques of augmentation were employed.

3. Model Development: Designing and training a CNN model and also comparing its efficiency with a binary classification model with the specific purpose of correctly classifying garbage images while taking into account the subtleties of actual situations. Various models were developed to check and compare different accuracy results for different hyper-parameters.

4. Evaluation: Assess the model's performance using relevant metrics like f1 score, accuracy, precision, recall and confusion matrix and then iterative improvements were carried out based on the results.

5. Deployment: Integrating the trained model into current waste management system to ensure compatibility and functionality.

The above project plan illustrates a concise road map for the successful implementation of an automated garbage classification model that aligns with stakeholders interests and achieves significant environmental, societal, and business impacts. It also includes all of the essential components needed to overcome current waste management challenges.

4 Data Understanding

For the garbage classification, the dataset is crucial and extremely important to the efficiency and durability of the model. For this project, the Trashnet dataset collected by Thung and Yang (*Rishma and Aarthi 2022*) was utilized. This collection of different kinds of garbage images aligns with the project's objectives. This section explains the perculiarities of data collection, data description, exploratory data analysis (EDA), and data quality verification.

4.1 Data Collection

The success of the project lies majorly in the Trashnet dataset, which was carefully chosen for its diversity and importance. The dataset consists of images of six different classes: cardboard, paper, plastic, trash, glass and metal. Each image represents a real-world waste item, which contributes to a broad understanding of garbage scenarios. To have a model that has the ability of capturing the nuances among different waste types, the inclusion of multiple classes is extremely important. These images, sized at 512 x 384 pixels, reflects the essence of the objects in detail. Typically, for a CNN model, image sizes ranging from 64 x 64 to 256 x 256 are often a good option; but for this CNN model, an image reduction was carried out. The images were reduced to 64 x 64 pixels while for One-vs-Rest binary classification model, the images were reduced to 28 x 28 pixels; first to make training the model faster and easier and secondly, to have a model trained not to expect high resolution images but on image quality just good enough for accurate classification. To imitate real-world conditions, each image features a single object placed against a white or plain background, signifying clarity and reducing possible distractions and noise.

4.2 Data Description

The trashnet dataset, with its six distinct classes, corresponds to the diverse range of garbage encountered in the real-world. Each image represents a specific waste type, labeled with its corresponding class type. The labels serve as the ground truth for training the neural networks, allowing the models to identify and classify visual features with each garbage class precisely.

Image						
Class	Cardboard	Trash	Glass	Metal	Plastic	Paper
Number	403	137	501	410	482	594

Total = 2527

Figure 4: Description of Original dataset and sample images with total images

4.3 Exploratory Data Analysis (EDA)

Conducting Exploratory Data Analysis (EDA) on the Trashnet dataset plays a fundamental role in comprehending its distribution, composition, and inherent challenges. This comprehensive examination incorporates qualitative assessments, statistical analysis, and visualizations to yield crucial insights into the intricacies that exert a profound influence on model development.

The EDA process uncovers a notable imbalance in distribution among the six garbage classes, necessitating meticulous analysis and the implementation of data augmentation techniques to ensure equitable representation during the model training phase. While minor variations in lighting conditions were observed, they were adjudged as inconsequential to the training of neural network models.

The examination highlights the imperative need for augmentation strategies. Rotation, flipping, and zooming emerge as pivotal methods for bolstering the model's generalization capabilities across a spectrum of waste scenarios, thereby ensuring equal representation of each class. The initial focus of data augmentation on the trash class dataset is a strategic measure to rectify the observed imbalance before extending augmentation to the entire dataset. This methodological choice is driven by the dataset's limited number of images, aiming to imbue the models with a more diverse range of training examples.

4.4 Data Quality Verification

Maintaining the Trashnet dataset's quality is crucial to our garbage categorization model's dependability. Extensive examinations were carried out to evaluate the uniformity of labelling, image integrity, and the existence of anomalies.

- Image integrity: Images were put through a rigorous quality check process to find and fix any corruption or distortion. In order to provide a strong basis for model training, it was important to make sure the dataset included high-quality images.

- Labelling Consistency: Ground truth labels were carefully examined to find and correct flaws or discrepancies. The rigorous verification procedure attempted to preserve the dataset's dependability by precisely matching labels to each image's content.

- Outliers: Images with uncommon viewpoints or extreme lighting were searched for. Outliers add diversity, but in order to balance variety and model generalisation, their effects on model performance were carefully considered. As far as the dataset is concerned, none was found.

5 Data Preparation

A pivotal phase within our garbage classification methodology is the Data Preparation step, wherein the raw pixel data from the Trashnet dataset undergoes meticulous restructuring to render it suitable for subsequent analysis. This process involves making discerning decisions regarding data splitting, resizing, normalization, and the integration of data augmentation techniques. Each stage is thoughtfully orchestrated to systematically mold the dataset, facilitating the nuanced comprehension of garbage categorization by the neural networks. The strategic planning at each juncture of the data preparation step is instrumental in preparing a dataset that optimally aligns with the complexities inherent in garbage classification.

5.1 One-Hot Encoding

One pivotal decision involved the one-hot encoding of the training, validation, and testing datasets. This method offers several advantages, with a significant impact on the modeling procedure being the equitable treatment of variables. Notably, one-hot encoding ensures that no class is unfairly prioritized; for instance, the order of the trash class does not diminish its importance compared to other classes. However, a drawback arises in the form of potential multi-collinearity issues, particularly in large datasets, leading to a reduction in model accuracy. To mitigate this, a strategic choice was made to retain the original structure of the majority of the dataset and solely augment the underrepresented trash class. The ensuing figures present a comparative table illustrating the one-hot encoding before and after augmenting the trash class. The initial table reveals a pronounced underrepresentation of the trash class (Class 5), with all instances interpreted as zero. To address this, data augmentation exclusively targeting the trash class was implemented, resulting in a corrected representation with equalized instances across all classes, as depicted in Figure 5(b).

Training Set One-Hot Encoding:

	Class_0	Class_1	Class_2	Class_3	Class_4	Class_5
0	1.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	1.0	0.0	0.0	0.0
2	0.0	1.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	1.0	0.0
4	1.0	0.0	0.0	0.0	0.0	0.0

Validation Set One-Hot Encoding:

	Class_0	Class_1	Class_2	Class_3	Class_4	Class_5
0	0.0	0.0	0.0	1.0	0.0	0.0
1	0.0	0.0	0.0	0.0	1.0	0.0
2	0.0	1.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	1.0	0.0
4	1.0	0.0	0.0	0.0	0.0	0.0

Test Set One-Hot Encoding:

	Class_0	Class_1	Class_2	Class_3	Class_4	Class_5
0	0.0	0.0	1.0	0.0	0.0	0.0
1	0.0	1.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	1.0	0.0
3	0.0	0.0	1.0	0.0	0.0	0.0
4	0.0	0.0	1.0	0.0	0.0	0.0

(a) One-hot Encoding before augmentation

Training Set One-Hot Encoding:

	Class_0	Class_1	Class_2	Class_3	Class_4	Class_5
0	0.0	0.0	0.0	0.0	0.0	1.0
1	0.0	1.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	1.0
3	1.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	1.0	0.0	0.0	0.0

Validation Set One-Hot Encoding:

	Class_0	Class_1	Class_2	Class_3	Class_4	Class_5
0	0.0	0.0	0.0	0.0	0.0	1.0
1	0.0	0.0	0.0	1.0	0.0	0.0
2	0.0	0.0	0.0	0.0	1.0	0.0
3	0.0	0.0	0.0	0.0	0.0	1.0
4	0.0	0.0	0.0	1.0	0.0	0.0

Test Set One-Hot Encoding:

	Class_0	Class_1	Class_2	Class_3	Class_4	Class_5
0	0.0	0.0	0.0	0.0	0.0	1.0
1	0.0	0.0	0.0	1.0	0.0	0.0
2	1.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	1.0
4	1.0	0.0	0.0	0.0	0.0	0.0

(b) On-hot Encoding after augmentation

Figure 5: 2 Figures of one-hot encoding before and after augmentation

5.2 Data Augmentation

A pivotal stage in the preparatory phase before model construction and training involves data augmentation. This process, essential for enhancing the robustness of the model, encompasses several advantages. Foremost, it contributes to the expansion of the dataset by generating additional images. Moreover, data augmentation promotes diversity within the dataset, thereby fortifying the model's capacity to generalize effectively. The transformative procedures applied during data augmentation encompass a spectrum of techniques, which include:

1. rotation_range=20
2. width_shift_range=0.2
3. height_shift_range=0.2
4. shear_range=0.2
5. zoom_range=0.2
6. horizontal_flip=True
7. vertical_flip=True

The fill mode was set to nearest because this interpolation method assigns the value of the nearest pixel in the original image. Also, this augmentation techniques were carried out only on the trash class dataset because it appears to be the only under-represented class in the dataset.

After augmenting the trash class dataset, the augmented images were saved into the same folder as the original images and they were all renamed accordingly in increasing order starting from 001. To allow the CNN model to learn properly and effectively from a wide range of images, another set augmentation was then carried out on the entire dataset as it was observed that the dataset was too small to train a neural network. The same augmentation techniques were carried out and the original dataset was increased from 2527 to 10902. The distribution of the images is shown below:

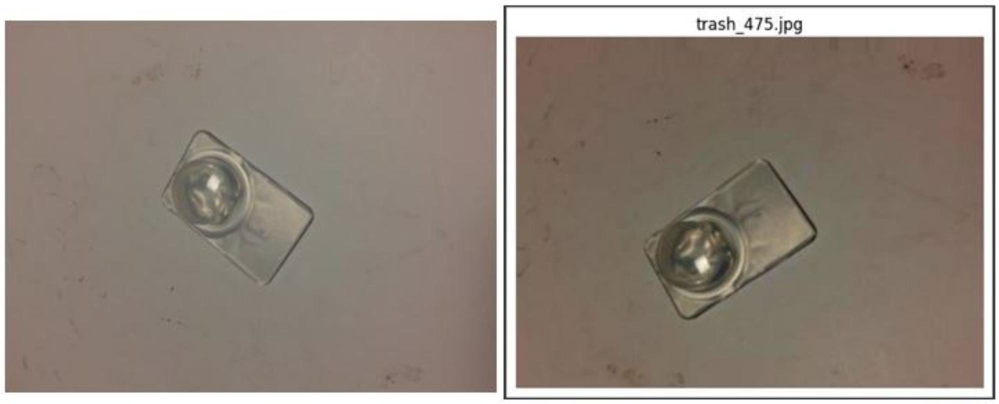


Figure 6: Original and augmented trash image

- Cardboard = 1529
- Paper = 2241
- Glass = 1912
- Metal = 1555
- Plastic = 1833
- Trash = 1812

5.3 Resizing and Normalization

The images in the dataset are resized to a reasonable 64 × 64 pixels through a smart change of their dimensions. This downsizing is a conscious decision to speed up model training without sacrificing the essential elements of each image's visual content.

Next comes normalization, an important step before training the model. The image pixel values are normalised to fall between 0 and 1 during this phase. In order to

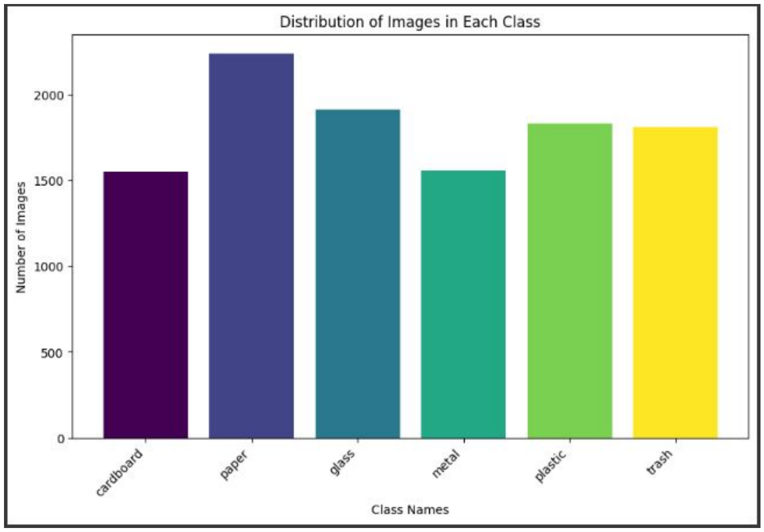


Figure 7: Distribution plot of the dataset

ensure scale homogeneity and preserve the integrity of the image, this normalization entails dividing each pixel value by 255. The pixels are arranged in uniform scales, preparing the CNN to recognise the subtleties of trash classifications.

```
#Data Normalization
X_train = X_train/255.0 #reducing the pixel values
X_val=X_val/255.0
X_test=X_test/255.0
```

Figure 8: Train set normalization

5.4 Data Splitting

We start the data preparation process by carefully dividing up our dataset. In this case, 20% of the image dataset serve as a specific validation and testing set, and the remaining 80%are designated for training. This is done in accordance to the Pareto Principle. This separation is a calculated move meant to strike a compromise between the requirement for an impartial benchmark to assess CNN's performance in the real world and its learning phase. The testing subset functions as an unbiased arbiter, evaluating the model's abilities on never-before-seen data, while the training subset acts as the furnace in which the model refines its understanding of garbage classes.

The way it has been done was to divide 80% for training and the remaining 20% was temporarily stored in a variable called **X_temp** and **y_temp**. These data was then divided equally for testing and validation.

6 Modeling

In this section, we will discuss extensively the modeling techniques that have been carried out to train and test the garbage dataset. Modeling is a very important part of building a system that accurately classifies waste and the choice of an architecture would determine how effective and robust the system operates. But the strength of the model is pivoted on a thorough data preparation process which has been done excellently. Two model architectures were explored for this research based on their effectiveness and simplicity:

- Convolutional Neural Network
- Artificial Neural Network (One-vs-Rest)

The two models will be explored by varying different hyper-parameters and then comparing the test accuracy for each model.

6.1 Model techniques selection

There are various models and techniques that would have been explored with more depth but time and and limited resources were a major constraint. Two models were shortlisted for experimentation: Convolutional Neural Network and One-vs-all. Convolutional neural networks excel in image classification due to their specialized architecture, which adopts concepts like localized receptive fields, hierarchical feature learning, weight sharing, and effective parameter learning *Yu, Jia, and Xu 2017*. These characteristics make CNNs suitable for interpreting visual patterns and have made them important in the field of computer vision.

To balance the scale and to see how another architectures will fair compared to CNN, a decision to use binary classification was made. When dealing with binary problems, i.e, problems that can be formulated as yes/no, spam emails/ no spam emails and so on, binary classification techniques are adopted. The above instances are relatively easy compared to when we have a multi class image classification task with more than two outputs.

6.2 Model 1: Convolutional Neural Network

The success of a Convolutional Neural Network (CNN) hinges on a carefully crafted architecture that navigates the complexities of garbage classification. In this section, we delve into the complexities of the CNN architecture, designed to understand the nuances within the Trashnet images. This architecture comprises of convolutional layers, fully connected layers, and a final output layer driven by a 6-way softmax function, offering a comprehensive approach to identification and classification.

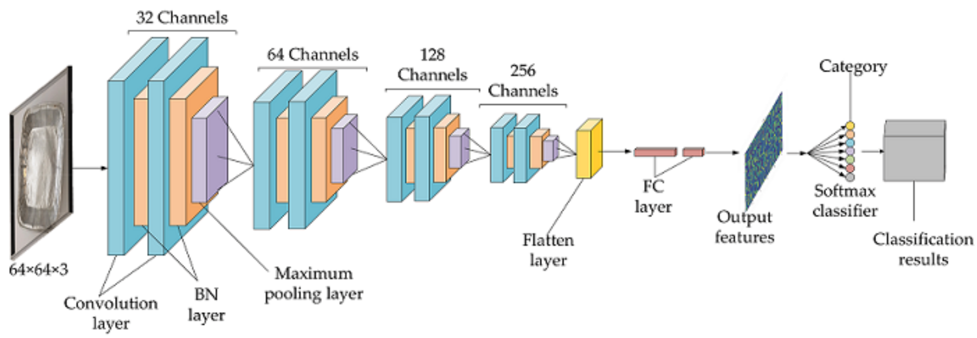


Figure 9: CNN architecture for Image classification

The image above from *Shi et al. 2021* is a typical CNN architecture for image classification. Now, we shall discuss the architecture we have used for our model. The CNN architecture comprises of the following:

1. Convolutional Input Layer: Our CNN architecture opens by making an instance of the Sequential function which refers to a linear stack of layers for our model. It allows to add one layer at a time, starting from the input layer and progressing through hidden layers until reaching the output layer. This first convolutional layer has 16 filters of dimensions 3 x 3 to discern basic visual patterns. Subsequent layers intensify the convolutional process, with 32, 64, 128, and 256 filters for different cases we have explored. Each filter, just like a visual receptor, extracts features and capture the intricacies of garbage items. This input layer receives input images, sized at 64 x 64 x 3.

2. Padding: We have used **padding="same"** here to preserve the spatial dimensions of the input volume in the output volume. Since the images in the trashnet dataset are of good quality and distinct, the details around the edges of the images are of no relevance for the classification. *Nanos 2023* provides a figure for an excellent description below.

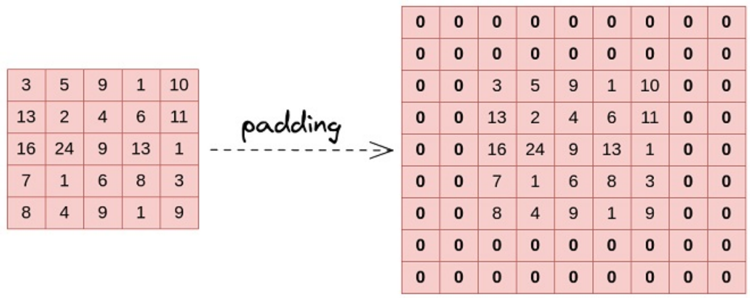


Figure 10: Padding technique

3. Strides: In the context of CNN, **Strides=(1,1)** refers to the movement of the convolutional filter or kernel across the input image during the convolution process. The notation "1,1" specifies the number of pixels the filter moves horizontally and vertically after each convolution.

4. Max-Pooling Layers: The pooling layers uniformly use a pool size of 2 x 2, initiating the extraction of salient features while disregarding redundant information.

5. Flattening and Fully Connected Layers: This is where the multi-dimensional output image is transformed to a single channel of neurons. For our CNN architecture, 1024 and 512 neurons have been used in different cases for a better understanding of complex features in the images.

6. Dropout: This is a technique used to train the model to avoid over-fitting. This way, the model learns efficiently, the hidden features in the data. We have varied this between using 0.25 or 25% dropout and 0.5 0r 50% dropout.

7. Loss Function: The loss function calculates the difference between the original input value and the predicted output value to estimate the error/loss during training and validation. For our CNN model, we have used categorical cross entropy and for one-vs-all, we have used binary cross entropy.

8. Optimizer: The optimizer uses the gradients of the model parameters with respect to the loss function to determine the direction and magnitude of updates. It aims to find the optimal parameter values that minimizes the loss. We have used **Adaptive Moment Estimation (Adam)** optimizer which is a type of stochastic gradient descent algorithm because it converges faster. For

the binary classification method, we have used Stochastic Gradient Descent (SGD).

9. Output Layer: The final layer serves as the classification stage. Here, a 6-way softmax function has been used. It ensures the model's confidence in assigning a specific label to the output image. The equation is given as:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{1}$$

10. Activation Function: The convolution layers in our architecture uses Rectified Linear Unit (Relu) activation functions. One specific use case of the Relu activation function is that it introduces non-linearity to the model enabling it to capture subtle patterns and correlations within the data. The graph is shown below:

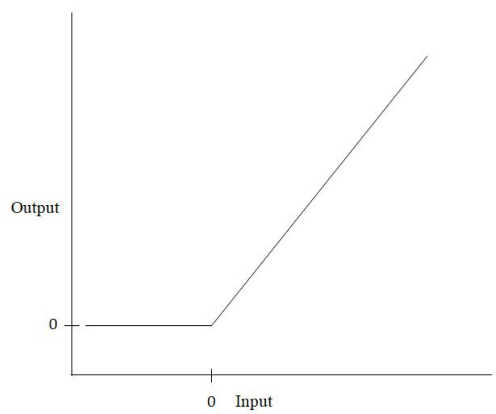


Figure 11: relu activation function

where

$$R(z) = max(0, z) \tag{2}$$

11. Learning rate: The learning rate inherently influences the swiftness with which the training process progresses. A judicious selection of the learning rate is imperative, as a high value may expedite training but risks yielding sub-optimal

learning outcomes. In the present study, a learning rate of 10^{-3} has been adopted for the model, reflecting a deliberate choice to balance the trade-off between training speed and learning efficacy.

12. Batch size: The batch size is a hyper-parameter that defines the number of samples processed in one iteration during training. In each epoch or training step, the model processes a batch of data, calculates the gradients, and updates the model's parameters. A large batch size could mean better convergence but can also lead to over-fitting. We have used different batch sizes like 32, 64 and 128 and compared the result in each case.

For the convolutional neural network architecture, we have tried to vary the hyper-parameters to achieve optimal results. We varied the convolution layers(1, 3, and 5), the epochs(32, 64, and 128), the dropouts(0, 0.25, and 0.5) and the batch size(32, 64, and 128).

Conv. layer	Epochs	Dropout	Batch Size
1	32	0	32
3	64	0.25	64
5	128	0.5	128

Table 2: Convolution layers used and the various hyper-parameters explored

The training results and evaluation are discussed in the next section. For further references, the complete colab notebook for the CNN model can be found using the link in appendix A.

6.3 Model 2: ANN Using One-vs-All (OvA)

Another method employed to tackle the image classification problem is to formulate the problem into multiple binary classification problems. That way, we can have each model to have a binary output where one class is trained against the other classes. This approach is called One-vs-all classification. The diagram below illustrates the approach.

The technique proposed is a simple binary classifier which is relatively straightforward compared to the CNN architecture. The major difference between this two architectures is that the one-vs-all technique proposed used a sigmoid function while the CNN model used a softmax instead. The sigmoid activation function computes

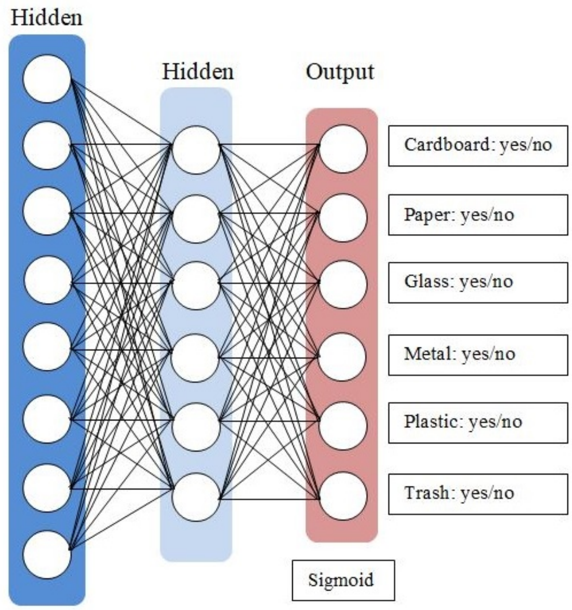


Figure 12: Multiple binary classifier

the probabilities of the image belonging to a certain class. If this value crosses the threshold, it is positive. If it is below the threshold, negative.

The function is given as:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

where S(x) = sigmoid function
e = Euler's number (e=2.71828)

To formulate the image classification problem in the form of a binary classification problem that uses one-vs-all, another set of data preparation procedures were carried out.

1. Pre-processing: A number of augmentation techniques have been carried and the image size has been reduced. To train the one-vs-all model to work effectively, we imported the original dataset again. The "load_and_preprocess_data" function is defined to load images from the specified directory and preprocess them. It uses the "ImageDataGenerator" from TensorFlow for image augmen-

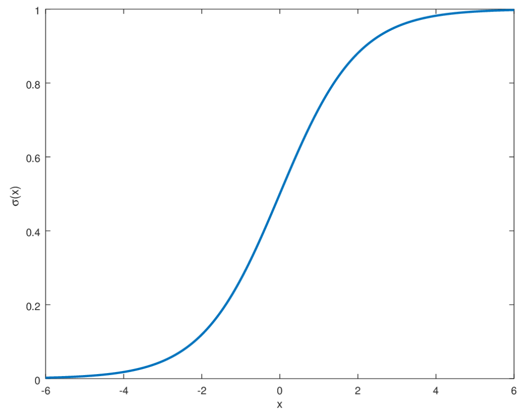


Figure 13: A sigmoid graph

tation. Image augmentation has been applied only to the "trash" class by setting the augment parameter. For the complete python code of the preprocessing stage, use the link in appendix C.

2. Augmentation: The trash class still appears to be lower than the rest of the classes, so necessary data augmentation techniques were carried just like before. The trash class images was then increased to 548. The table below shows the distribution of images in each class with a total of 2938 images.

Class	No. of Images
Cardboard	403
Glass	501
Paper	410
Metal	594
Plastic	482
Trash	548

Table 3: Distribution of class after augmenting 'trash' class

3. Pareto Principle: We then split the new dataset into training, and test set. 80% of the dataset was used for training while 20% was used for testing.

4. Normalization: It was also important to normalize the dataset by dividing train and test data by 255.0. This normalizes the pixel values between 0 and 1.

5. Label Encoding: We have used the 'LabelEncoder' function to convert the class labels to binary (0 and 1), and the model is trained to predict this binary labels.

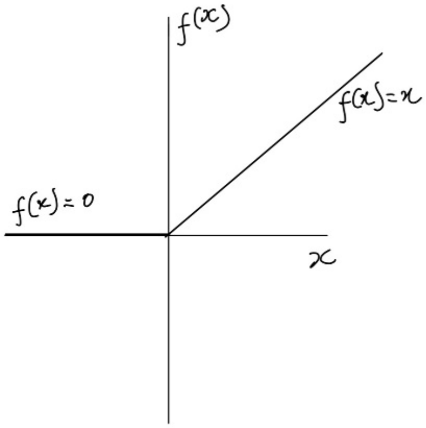
```python
# Normalize pixel values to be between 0 and 1
data = data / 255.0

# Encode labels
label_encoder = LabelEncoder()
labels = label_encoder.fit_transform(labels)
labels = to_categorical(labels)

return data, labels
```

Figure 14: Label encoding the dataset

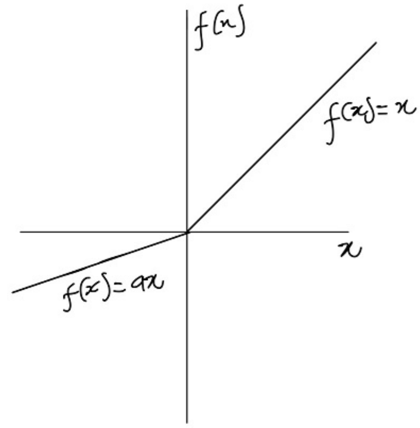
6. Modeling: The model architecture is a simple feed-forward neural network with three dense layers. For each class, a binary classifier is trained using a one-vs-all approach. The label for the current class is converted to binary (0 or 1), and the model is trained to predict this binary label. The first layer opens with a flatten layer that receives an input shape of 28 x 28 x 3. The flatten layer is used to transform 3D output into a 1D vector. It is necessary because fully connected layers expects a flat input.

   Rectified Linear Unit (Relu) has been used as the activation function in the dense layers mainly because it helps mitigate the vanishing gradient problem, which normally occur with activation functions that squash their input like the sigmoid function. Dying Relu is a common problem with this activation function, that is the reason leaky Relu or Parametric Relu was introduced to address these issues.

   Because we have a binary formulated problem, the binary cross entropy has been used as the loss function. The sigmoid function was used as the output's activation function for the binary classification. The training regimen further encompasses specific configurations, including a batch size of 32 and a training duration spanning 64 epochs. Additionally, an early stopping criterion has been instituted, characterized by a patience parameter set to 5. This measure has been implemented strategically to avert unnecessary computational and training time expenditures.



(a) dying Relu



(b) Leaky Relu

6.4 Models' Assessment

The following metrics have been adopted to assess the model's performance during training, validation and testing.

1. Accuracy: This is an important metric that calculates the ratio of correctly predicted instances to the total instances. It is a measure of how good the model is performing in every instance of prediction.

   Mathematically,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

   where TP= True Positives
   TN= True negatives
   FP= False positives
   FN= False negatives

2. Precision: Precision measures the accuracy of the positive prediction. That is, how precise are the predictions the model have predicted positive. It is the ratio of the true positives to the predicted positives.
   Mathematically,

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

3. Recall: Also known as sensitivity, or true positive rate measures the ability of the model to capture relevant instances. It is the ratio of the true positives to the actual positives.
   Mathematically,

$$Recall(TPR) = \frac{TP}{TP + FN} \tag{6}$$

4. F1 score: It is a function of precision and recall. F1 score is the harmonic mean of these two metrics. It is a good measure to find a balance between precision and recall especially when dealing with unbalanced dataset.

   Mathematically,

$$f1 - score = \frac{Precision * Recall}{Precision + Recall} \tag{7}$$

5. Specificity: It is also called true negative rate. Although, it is a very important and relevant metric for assessing our model, it measures how good a model is in identifying and classifying the negative classes as negative.

   Mathematically,

$$Specificity(TNR) = \frac{TN}{TN + FP} \tag{8}$$

   There are several other metrics used in machine learning to assess a model's performance but these are the relevant ones within the scope of this study.

6. Confusion Matrix: The last metric and arguably the most important metric for assessment is the confusion matrix. It gives a detailed breakdown of correct and incorrect predictions. It shows the true positives, true negatives, false positives and false negatives.

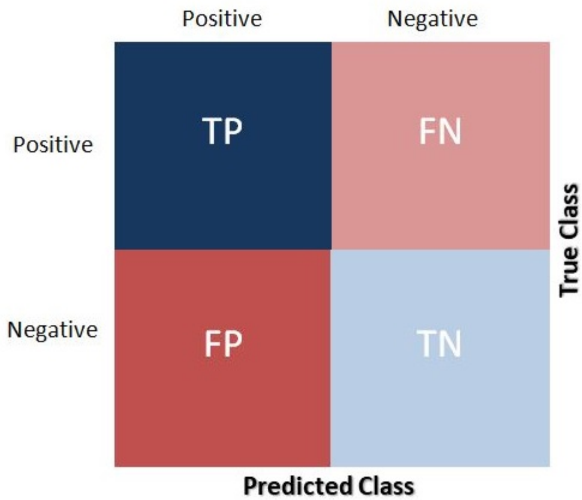


Figure 16: Confusion matrix

7 Evaluation

Assessing and evaluating the model's performance is an important section in building an efficient garbage classification model (*Vujović et al. 2021*). The choice of evaluation metrics used will determine how efficient the model performs as they play an important role for optimal results during classification (*Hossin and Sulaiman 2015*). There are different evaluation metrics and each of them are problem specific but according to (*Choi et al. 2021*), there are six important metrics to always pay attention to: accuracy, precision, recall, F1 score, receiver operating characteristic curve (ROC), and area under ROC (AUC). The accuracy serves as the most important of them all and many pay attention to this metric but depending on the problem the model is trying solve, other metrics might be suitable (*Grandini, Bagli, and Visani 2020*). Overall, the decision to use one metric over another seem or which numbers to pay attention to seems like an impossible task but setting a baseline score for the model addresses the problem of indecision or uncertainty.

7.1 Evaluation Results

We shall be presenting and discussing the results for each model and the respective evaluation metrics that have been used.

7.1.1 Model 1: Convolutional Neural network

The initial experiment involved the systematic variation of convolution layers, constituting the first phase of exploration. Subsequent experiments, constituting the second and third phases, were conducted by manipulating epochs and dropouts, followed by a comprehensive investigation into the impact of varying batch sizes in the final architectural configuration. This structured approach enabled a nuanced examination of the convolutional neural network's performance under distinct architectural configurations, contributing valuable insights into the interplay between these architectural elements and model outcomes.

Epochs = 64, Learning Rate = 0.001, Optimizer = Adam
Batch Size = 32, Loss = Categorical Cross entropy, Dropout = 0

Conv. layer	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Testing Accuracy	Testing Loss
1	0.8382	0.4536	0.7963	0.5891	0.8423	0.4632
3	0.9559	0.1295	0.9110	0.2664	0.9413	0.2042
5	0.9357	0.1913	0.8945	0.3091	0.9065	0.3070

Conv. layers = 3, Learning Rate = 0.001, Optimizer = Adam
Batch Size = 32, Loss = Categorical Cross entropy, Dropout = 25%

Epochs	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Testing Accuracy	Testing Loss
32	0.8814	0.3326	0.8651	0.3917	0.8836	0.3317
64	0.9396	0.1805	0.9083	0.2754	0.9102	0.2568
128	0.9650	0.1108	0.9450	0.2265	0.9615	0.1761

Epochs = 64, Learning Rate = 0.001, Optimizer = Adam
Batch Size = 32, Loss = Categorical Cross entropy, Conv. layers = 3

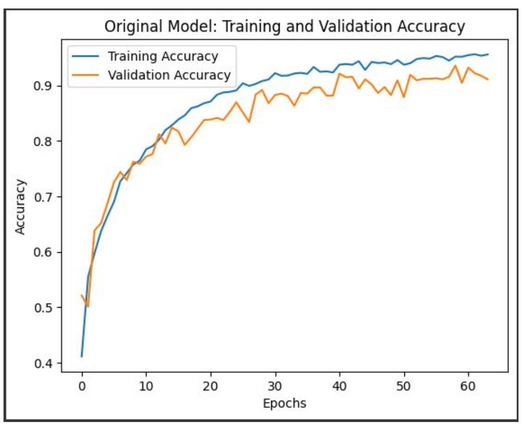
Dropout (%)	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Testing Accuracy	Testing Loss
0	0.9559	0.1295	0.9110	0.2664	0.9413	0.2042
25	0.9396	0.1805	0.9083	0.2754	0.9102	0.2568
50	0.9205	0.2302	0.9055	0.2643	0.9212	0.2483

Epochs = 64, Learning Rate = 0.001, Optimizer = Adam
Conv. Layer = 3, Loss = Categorical Cross entropy, Dropout = 25%

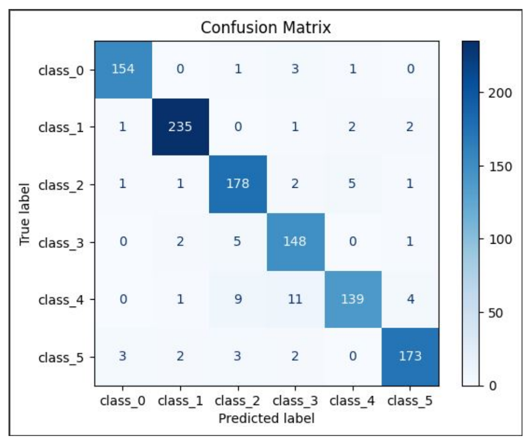
Batch Size	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss	Testing Accuracy	Testing Loss
32	0.9396	0.1805	0.9083	0.2754	0.9102	0.2568
64	0.9265	0.2136	0.9037	0.2638	0.9505	0.1652
128	0.9419	0.1712	0.9156	0.2573	0.9395	0.1848

The first table shows the accuracy and loss recorded during training, validation and testing after varying the convolution layers. We have used 1-convolution layer, 3-convolution layers and 5-convolution layers. In each variant, we have kept some hyper-parameters constant: epochs, batch size, learning rate, optimizer, loss function and dropout. In the second table, we have varied the epochs by using 32, 64 and 128. In these different cases, we have kept the hyper-parameters constant likewise. The third and fourth tables shows the results while varying the dropout and the batch size.

The test accuracy achieved in each of the different cases are promising results and this was because proper attention was given to the dataset at the pre-processing stage. More time was spent processing and making the dataset fit for training and modeling. Three models were saved after considering the accuracy and precision of the models in each of the different cases. The model with 3-convolution layers achieved a test accuracy of 94%, higher than the rest of the model in that variant. The confusion matrix and classification report is shown below.



(a) Accuracy plot of CNN using 3 convolution layers



(b) Confusion matrix

7.1.2 Calculations

We will be calculating the precision and recall for class 2 which belongs to the metal dataset and then compare with the values shown above.

Precision, P

$$= \frac{TP}{TP + FP}$$

$$= \frac{178}{178 + 5 + 9 + 3}$$

$$= \frac{178}{195} \approx 91\%$$

Also,
Recall, R

$$= \frac{TP}{TP + FN}$$

$$= \frac{178}{178 + 2 + 5 + 1}$$

$$= \frac{178}{186} \approx 95\%$$

and,

f1-score

$$= 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$= 2 * \frac{91 * 95}{91 + 95}$$

$$= 2 * \frac{8645}{186} \approx 93\%$$

These values are consistent with the one shown in the classification report for class 2.

Classification Report:

	precision	recall	f1-score	support
0	0.97	0.97	0.97	159
1	0.98	0.98	0.98	241
2	0.91	0.95	0.93	188
3	0.89	0.95	0.92	156
4	0.95	0.85	0.89	164
5	0.96	0.95	0.95	183
accuracy			0.94	1091
macro avg	0.94	0.94	0.94	1091
weighted avg	0.94	0.94	0.94	1091

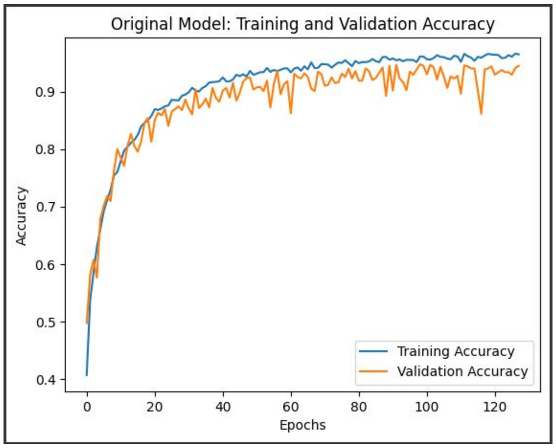
Figure 18: Classification report for CNN variant with 3 convolution layers

The other variant of the CNN architecture that gave excellent results was the one with 128 epochs and the one with 64 batch sizes as seen in the tables above. The classification report, confusion matrix and accuracy plot for the 128 epochs variant are shown below.

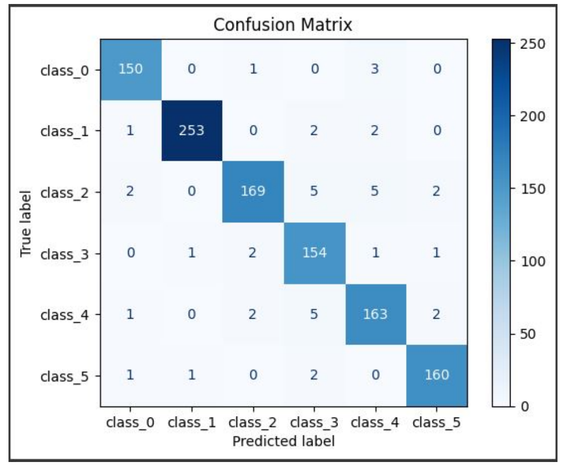
Classification Report:

	precision	recall	f1-score	support
0	0.97	0.97	0.97	154
1	0.99	0.98	0.99	258
2	0.97	0.92	0.95	183
3	0.92	0.97	0.94	159
4	0.94	0.94	0.94	173
5	0.97	0.98	0.97	164
accuracy			0.96	1091
macro avg	0.96	0.96	0.96	1091
weighted avg	0.96	0.96	0.96	1091

Figure 19: Classification report for the CNN variant with 128 epochs

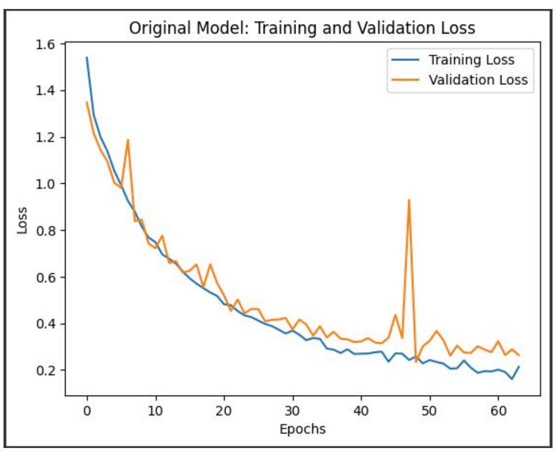


(a) Accuracy plot of CNN using 128 epochs

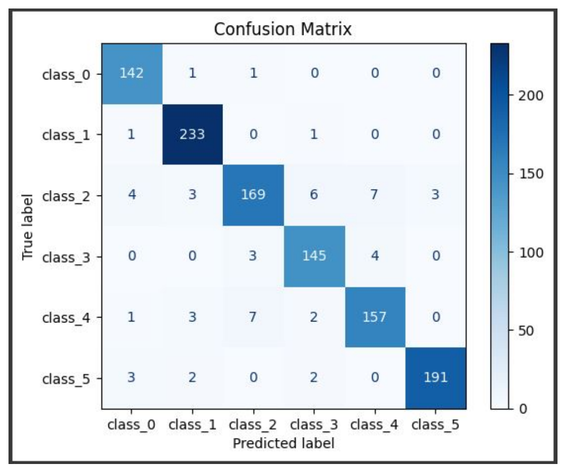


(b) Confusion matrix

The classification report, confusion matrix and accuracy plot for the 64 batch size variant are shown below.



(a) Accuracy plot of CNN using 64 batch sizes



(b) Confusion matrix

Classification Report:

	precision	recall	f1-score	support
0	0.94	0.99	0.96	144
1	0.96	0.99	0.98	235
2	0.94	0.88	0.91	192
3	0.93	0.95	0.94	152
4	0.93	0.92	0.93	170
5	0.98	0.96	0.97	198
accuracy			0.95	1091
macro avg	0.95	0.95	0.95	1091
weighted avg	0.95	0.95	0.95	1091

Figure 22: Classification report for the CNN variant with 64 batch size

7.1.3 Analysis of result

In the first experimental variation, where the number of convolution layers was systematically altered, the configuration employing three convolution layers demonstrated superior performance, yielding the highest accuracy, precision, and recall. Notably, as the number of convolution layers increased, a subsequent decline in accuracy was observed. The choice of optimizer emerged as a pivotal determinant in achieving high accuracy, with the Adam optimizer outperforming stochastic gradient descent. Dropout was initially omitted due to concerns about limited training data; however, a subsequent observation of favorable results prompted the introduction of a 25% dropout rate in the subsequent variant.

In the second experimental variant, where the number of iterations was manipulated, a consistent three convolution layer configuration was maintained, given its optimal performance in the initial experiment. Commencing with a modest epoch of 32, the training progression was monitored to ascertain the evolution of accuracy. An escalation in dropout from 25% to 50% resulted in a modest accuracy increase but was not deemed substantially impactful.

In the CNN variant of varying batch sizes, a test accuracy of 95% was achieved with a batch size of 64. The batch size states the volume of images to pick per epoch for training, calculating and updating the weights. This number appears to be in synergy with the three convolution layers used. Adam optimizer has been shown to produce effective performance with the softmax activation function compared to the stochastic gradient descent.

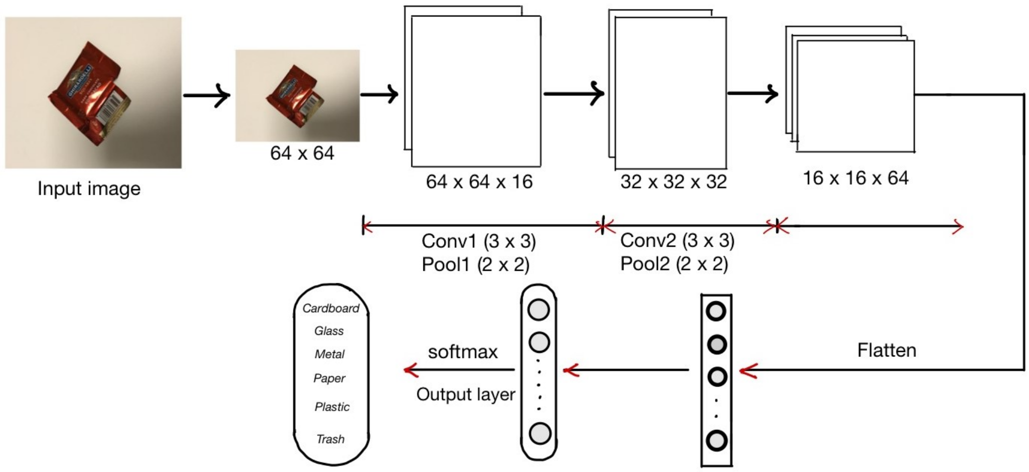


Figure 23: Proposed CNN architecture

7.1.4 Model 2: Artificial Neural Network (OvA)

The one-vs-all binary classifier model developed uses Adam optimizer. The model has been fine-tuned using the optimal hyper-parameters to achieve a good performance. The first table below shows the accuracy, precision, recall and the f1-score for all the classes from class 0 to class 5 that were correctly identified to belong to the true class, i.e. those classes that the binary classifier predicted to be the true classes.

Class	Accuracy	Precision	Recall	f1-score
0	0.92	0.95	0.96	0.95
1	0.81	0.81	1.00	0.89
2	0.85	0.89	0.93	0.91
3	0.87	0.88	0.96	0.92
4	0.85	0.91	0.90	0.91
5	0.96	0.96	0.99	0.98

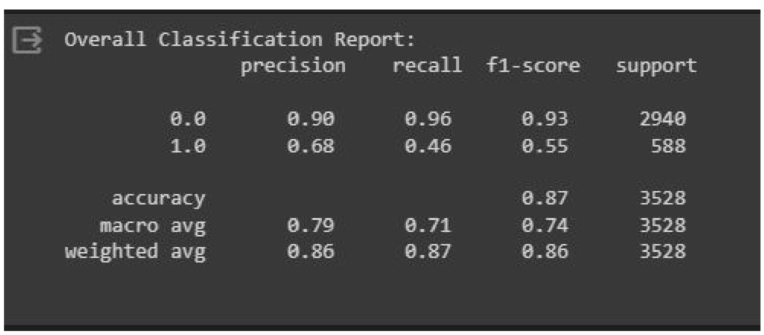
Table 4: Classification report of Classes 0 to 5

For the classes of images who were correctly classified as not belonging to the true class, the table below shows the accuracy, precision, recall and f1-score for this classification.

Class	Accuracy	Precision	Recall	f1-score
0	0.92	0.68	0.65	0.66
1	0.81	1.00	0.00	0.00
2	0.85	0.47	0.34	0.39
3	0.87	0.77	0.50	0.61
4	0.85	0.49	0.53	0.51
5	0.96	0.97	0.81	0.88

Table 5: Classification report of Classes 0 to 5

Apparently, the model is failing to correctly classify images as not belonging to a particular class; however, with a higher degree of accuracy and precision, classifies the true class of images to their correct classes. The classification report showing the overall test accuracy is displayed below.

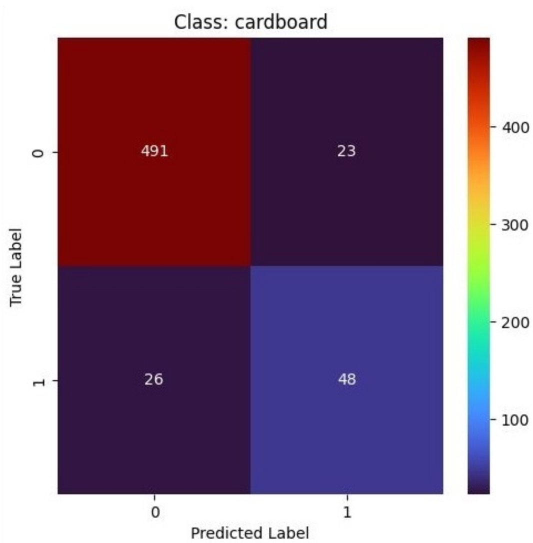


```
Overall Classification Report:
              precision    recall  f1-score   support

         0.0       0.90      0.96      0.93      2940
         1.0       0.68      0.46      0.55       588

    accuracy                           0.87      3528
   macro avg       0.79      0.71      0.74      3528
weighted avg       0.86      0.87      0.86      3528
```

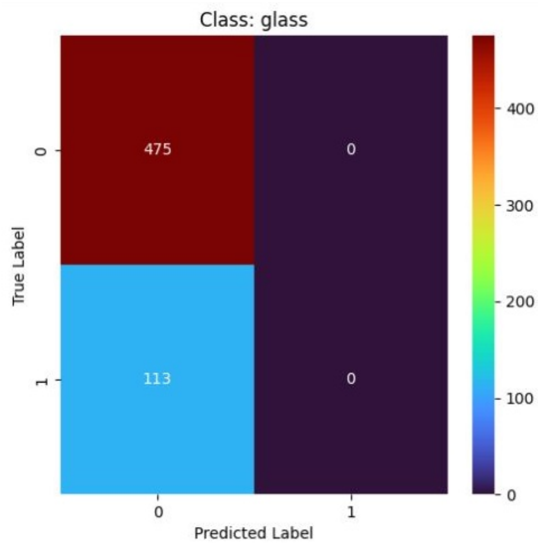
Figure 24: Diagram showing the overall classification report

Compared to the CNN model, the binary classification model using one-vs-rest is not performing optimally using Adam optimizer. The stochastic gradient descent was used in another variant of the model as an optimizer. For further references, the complete python code is available in the link in Appendix B.

We can visualize the confusion matrix of the cardboard and glass class below,



(a) Confusion matrix of Cardboard class



(b) Confusion matrix of glass class

More visualization can be accessed from the appendix section where the python notebooks have been listed.

7.1.5 Analysis of result

From the classification report presented above, we can infer that the binary classification model is not fit to handle the intricacies and detect the nuances in images for a multi-class image classification problem. An attempt to improve the accuracy was carried out by changing the optimizer from Adam to SGD, instead the accuracy declined.

The field of computer vision in deep learning is a rather complex and unique field where specialized tools are utilized that can identify and detect the subtle features of images. This is the area where convolutional neural network thrives. We have managed to get the overall test accuracy to 46.77%. Although, the training accuracy was in the $90th$ percentile, the test performance was low and this is as a result of the difficulty of a binary classifier to handle complex multi-class classification problem as encountered in the trashnet dataset.

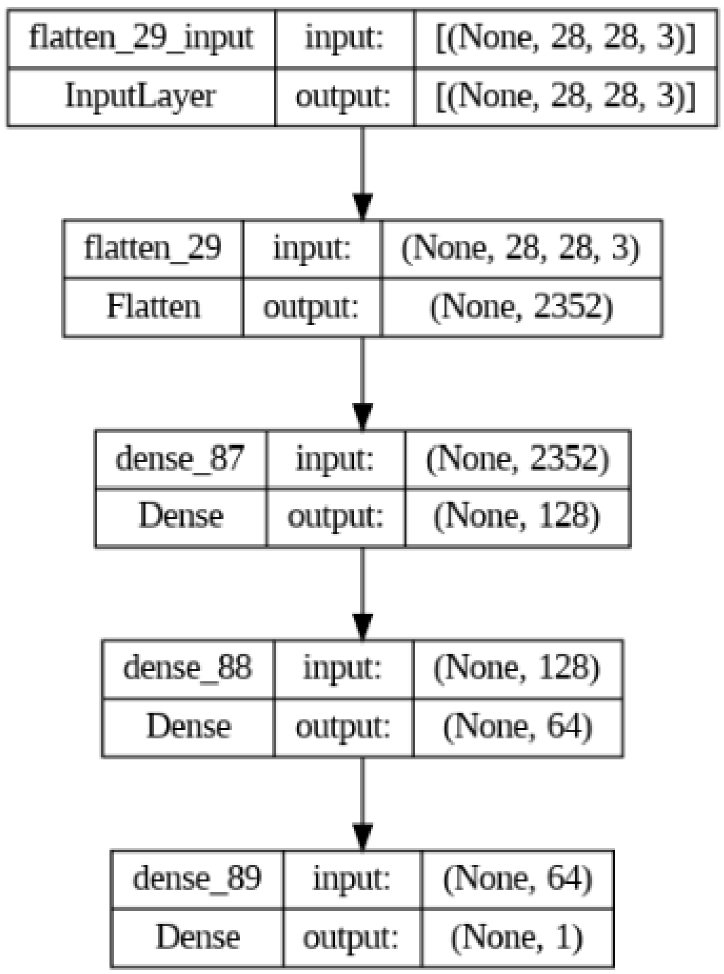


Figure 26: Model plot showing the model's structure

7.1.6 Saving the model

There are various formats of saving a neural network model, but for our architecture, we have the model with the highest test accuracy of 96% as a standard HDF5 file used by keras. It includes both the model architecture and the learned weights.

```
[ ]  model.save('smart_garbage_B.h5')

     /usr/local/lib/python3.10/dist-packages/keras/src/engine
       saving_api.save_model(
```

Figure 27: saving the model in h5 format

This model can then be used to make predictions or further training by simply calling the load_model function in keras to load it.

```
[11] model_path= '/content/drive/MyDrive/smart_garbage_B.h5'
     model=load_model(model_path)

[12] img_path='/content/metal_010.jpg'
     img= image.load_img(img_path, target_size=(64, 64))
     img_array=image.img_to_array(img)
     img_array=np.expand_dims(img_array, axis=0)
     img_array /= 255.0

     prediction=model.predict(img_array)

     predicted_class_index=np.argmax(prediction)

     class_name=['cardboard', 'paper', 'glass', 'metal', 'plastic', 'trash']

     predicted_class_name=class_name[predicted_class_index]

     print('Raw prediction score:', prediction)

     print("predicted class index:", predicted_class_name)

     1/1 [==============================] - 0s 88ms/step
     Raw prediction score: [[5.1450138e-12 2.7233231e-12 1.0598724e-06 9.9999738e-01 1.5855991e-06
       1.6244422e-10]]
     predicted class index: metal
```

Figure 28: Loading the model

8 Deployment

The deployment is an important part of building an effective model and every choice and decision are deliberate and calculated. For the CNN architecture, three models were saved that gave 94%, 95% and 96% test accuracy. The binary classifier model gave a test Accuracy of 46.77% and a decision to deploy the model with the highest test accuracy was made. There are several web frameworks like Flask for python, Django and others which are excellent, flexible and easy to use but a last minute decision was made to try something differently. We have created a GUI application for the deployment of the model using an open source library called Taipy. The figure below shows the deployment flowchart that has been utilized.

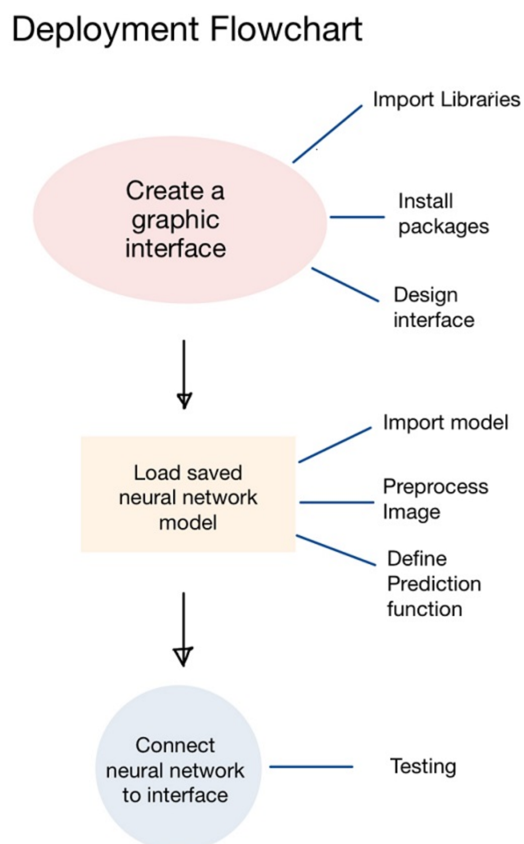


Figure 29: Deployment flowchart

8.1 Deployment Plan

The deployment plan encompasses several key steps to ensure a successful integration and user experience (*Moolayil and Moolayil 2019*):

1. Graphical User Interface Development: The initial phase of our developmental roadmap involves the establishment of a user-friendly web interface (GUI) utilizing Taipy. Taipy, a Python open-source library, has been harnessed for the creation of an intuitive GUI web application to deploy our Convolutional Neural Network (CNN) model. Leveraging Taipy significantly simplifies the web application creation process, circumventing the necessity for an in-depth understanding of HTML, CSS, and JavaScript. The graphical elements are crafted utilizing the Anaconda code editor as the command window and Bracket, a dedicated source code editor for web development.

   Using the Anaconda editor, a new environment labeled "dl_env" was established, and Python 3.11 was installed within this environment. Following the installation, the environment was activated, and the Taipy library was incorporated. Subsequently, a foundational folder was established to house essential dependencies, including the model, web logo, placeholder image, and a requirements file. The designated path to the Anaconda Integrated Development Environment (IDE), where subsequent work would be conducted, was then documented. In the Brackets environment, imperative libraries such as GUI, NumPy, Image from the Python Imaging Library (PIL), and models from the TensorFlow.Keras framework were imported to facilitate the development process.

```
from taipy.gui import Gui
from tensorflow.keras import models
from PIL import Image
import numpy as np
```

Figure 30: Libaries for deployment on Bracket

   For the conceptualization of the user interface in the deployment phase, an instantiation of the Graphical User Interface (GUI) was effectuated. The implementation incorporated the utilization of the use_reloader=True function, a strategic measure aimed at circumventing the need to reload the Anaconda terminal repetitively. Given the web-based nature of the project, HTML was employed to construct integral components, including an image control facilitating the upload of the garbage classifier's logo, a file selector with an associated "on change" function—a button linked to a callback function intended for updating the image element—and finally, an indicator control conveying the degree of confidence associated with the model's prediction.

```
content = ""
img_path = "placeholder_image.png"
prob = 0
pred = ""

index = """
<|text-center|
<|{"logo.png"}|image|width=15vw|>

<|{content}|file_selector|extensions=.jpg|>
Upload a garbage image from your file system

<|{pred}|>

<|{img_path}|image|>

<|{prob}|indicator|value={prob}|min=0|max=100|width=25vw|>
>
"""
```

Figure 31: Processes for Deployment

2. Loading Saved Model: To instantiate the previously trained Convolutional Neural Network (CNN), the "load_mode"l function from the TensorFlow.Keras library was employed. The outcome of this loading process was subsequently presented and automatically visualized within the Anaconda prompt environment.

```
flatten (Flatten)            (None, 4096)            0

dense (Dense)                (None, 1024)            4195328

activation_3 (Activation)    (None, 1024)            0

dropout (Dropout)            (None, 1024)            0

dense_1 (Dense)              (None, 6)               6150

=================================================================
Total params: 4225062 (16.12 MB)
Trainable params: 4225062 (16.12 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 32: Anaconda prompt of the model architecture

After this, we defined a "predict_image" function that takes "model" and "path_to_img" as argument to preprocess the images. The preprocessing includes resizing the image uploaded to 64 x 64 which is what the model expects, converting the "img_path" to an array of image using "Image" from PIL which is then normalised by dividing by 255.0.

```python
model = models.load_model("smart_garbage_B.h5")

def predict_image(model, path_to_img):
    img = Image.open(path_to_img)
    img = img.resize((64, 64))
    data = np.asarray(img)
    data = data / 255
    probs = model.predict(np.array([data])[:1])

    top_prob = probs.max()
    top_pred = class_name[np.argmax(probs)]

    return top_prob, top_pred
```

Figure 33: Creating function for prediction

Uploading an image to the web page for prediction looks like this in the anaconda terminal window

```
before [203 194 179]
after [0.79607843 0.76078431 0.70196078]
1/1 [==============================] - 0s 154ms/step
[[2.6704177e-07 1.0272467e-05 7.2708271e-02 9.0749320e-03 9.1787881e-01
  3.2744082e-04]]
0.9178788
4
```

Figure 34: Testing interface on Anaconda

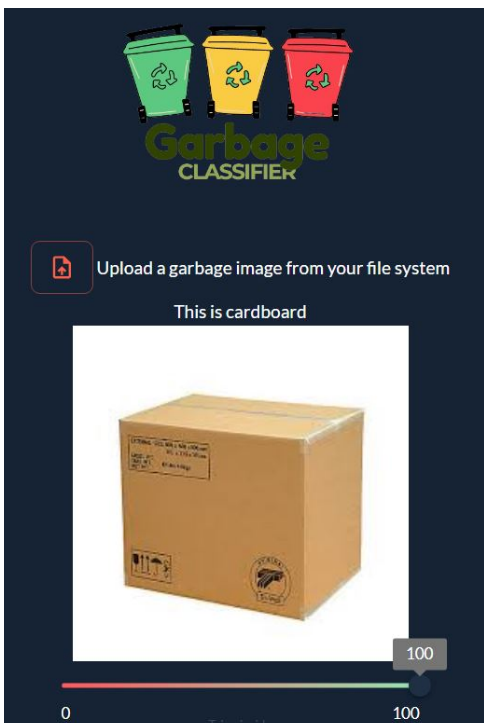
Additionally, the predictive outcome manifests as an array, wherein the classification output designates class 4 with a corresponding confidence level of 0.9178788. The interpretative process involves mapping the softmax output onto a dictionary comprising class names, subsequently scaling the values by a factor of 100 to yield a percentage representation, and rounding the outcome to an integer value.

3. Connecting Neural Network to Interface: The free url we have used uses port 5000 as the localhost: http://127.0.0.1:5000/.

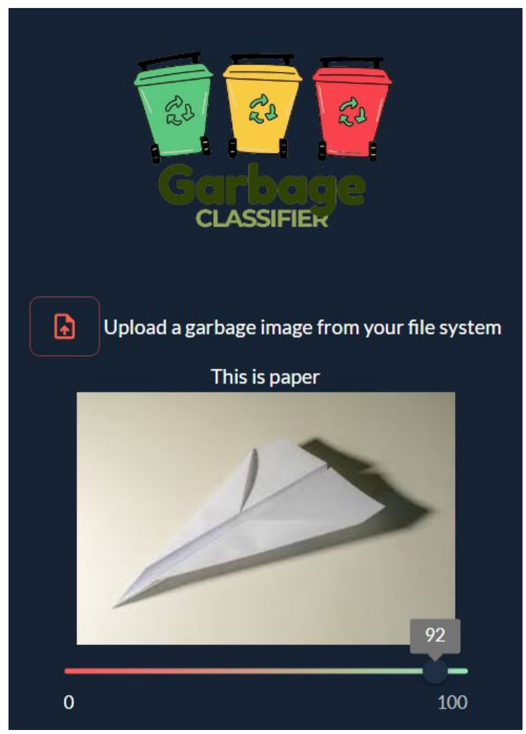
```
class_name = {
      0: 'cardboard',
      1: 'paper',
      2: 'glass',
      3: 'metal',
      4: 'plastic',
      5: 'trash'
  }
```

Figure 35: Mapping of classes

We test the model by uploading a random image which has not been used for training nor seen by the model before. The web interface displays the percentage of confidence of this prediction. For example, we test the model by uploading a cardboard and paper image.



(a) Test result (Cardboard)



(b) Test result (Paper)

8.2 Deployment Consideration

Several considerations are essential for the successful deployment of the garbage classification model:

1. Web Framework Deployment (Taipy): Leveraging Taipy's simplicity and versatility for web deployment ensures a smooth integration process. Additionally, Taipy enables the creation of RESTful API endpoints with the adoption of GUI interface, facilitating seamless interaction with the garbage classification model.

2. HDF5 Model Integration: The HDF5 format aligns seamlessly with Keras model, ensuring compatibility with the garbage classification model. Implementing a robust loading mechanism for the HDF5 model within Taipy ensures efficient and reliable model integration.

3. Front-End Optimization: Focusing on enhancing the front-end with an intuitive design and responsive features to optimize the user experience is essential. Incorporating asynchronous request handling to improve responsiveness during image uploads and inference creates a smooth user interaction.

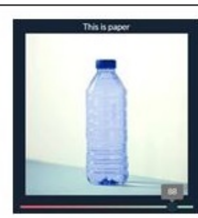
Class	Cardboard	Paper	Metal
Correctly classified			
Incorrectly classified			

Figure 37: Model's Prediction results

9 Conclusion

In this project, the implementation of two distinct neural networks are implemented. The first network adopts a Convolutional Neural Network (CNN) architecture, while the second employs a binary classifier designed for the categorization of garbage images into six distinct classes: cardboard, glass, metal, paper, plastic, and trash. The experimentation encompassed the development and evaluation of three CNN models, wherein varying hyper-parameters were manipulated to discern optimal outcomes. The exploration of a One-vs-All approach for multi-class classification underscores the comparative inefficiency of binary classifiers in handling the complexities inherent in multi-class image classification, particularly when juxtaposed with the robust capabilities of the convolutional neural network. Notwithstanding the potential for refinement and enhancement through further fine-tuning, experimentation with additional hyper-parameters, and the incorporation of data augmentation techniques, the CNN model emerges as an intuitively adept tool for the nuanced demands of image classification.

The prospective benefits of deploying this automated garbage classification model extend beyond the research domain, offering substantial business value. These advantages encompass heightened operational efficiency realized through the optimization of waste management processes and resource allocation, the fostering of environmental sustainability by championing recycling initiatives and curbing contamination, and the prospect of cost reduction in waste disposal, achieved through labour cost savings and the mitigation of penalties associated with non-compliance.

This CNN model can be used in waste management applications especially as a Smart Garbage Bin where only one image will be in the frame at a single time. The limitations of this model are:

- it cannot be used to classify multiple images in a single frame.
- there are some prediction errors despite the high test accuracy it achieved which indicates a possible over-fitting.

The extant model architecture exhibits potential for refinement through the augmentation of training data with a more expansive dataset than has hitherto been utilized. Furthermore, there exists an opportunity to optimize the binary classifier model employing the one-vs-rest strategy in tandem with the convolutional neural network (CNN) model. This joint enhancement initiative aims to augment the model's proficiency in classifying multiple waste items within a singular frame.

A Appendix 1

For the complete google colab notebook on preprocessing and CNN model used, use the link in bracket (CNN model and Preprocessing).

B Appendix 2

For complete google colab notebook on the One-Vs-All binary classification model, use the link in the bracket (One-vs-all model).

C Appendix 3

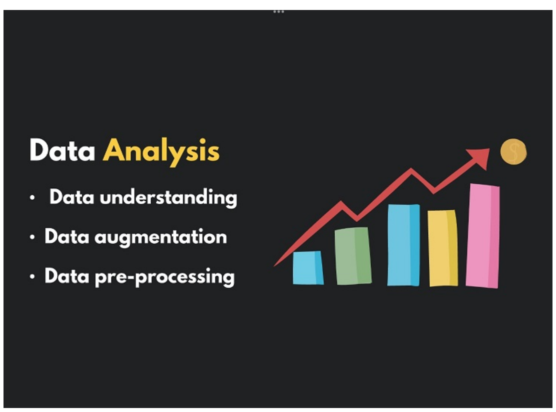
For the complete data preprocessing and augmentation notebook used for the one-vs-all binary classification model, use the link in the bracket (Data Augmentation for OvA model).

D Appendix 4

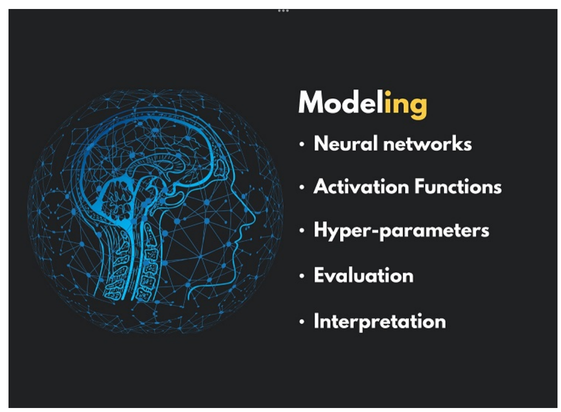
D.1 Presentation Slides

(a) slide 1

(b) slide 2



(a) slide 3



(b) slide 4

(a) slide 5

(b) slide 6

(a) slide 7

References

Abubakar, Ismaila Rimi et al. (2022). "Environmental sustainability impacts of solid waste management practices in the global South". In: *International Journal of Environmental Research and Public Health* 19.19, p. 12717.

Adedeji, Olugboja and Zenghui Wang (2019). "Intelligent waste classification system using deep learning convolutional neural network". In: *Procedia Manufacturing* 35, pp. 607–612.

Ahmed, Mohammed Imran Basheer et al. (2023). "Deep Learning Approach to Recyclable Products Classification: Towards Sustainable Waste Management". In: *Sustainability* 15.14, p. 11138.

Bhattacharya, Sabitabrata et al. (2023). "Automated Garbage Classification using Deep Learning". In: *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*. IEEE, pp. 404–410.

Bircanoğlu, Cenk et al. (2018). "RecycleNet: Intelligent waste sorting using deep neural networks". In: *2018 Innovations in intelligent systems and applications (INISTA)*. IEEE, pp. 1–7.

Chen, Zhichao et al. (2022). "Garbage classification system based on improved ShuffleNet v2". In: *Resources, Conservation and Recycling* 178, p. 106090.

Choi, Jung-Gu et al. (2021). "Machine learning framework for multi-level classification of company revenue". In: *IEEE Access* 9, pp. 96739–96750.

datatopics.worldbank.org (2023). *what-a- waste/trends in solid waste managemen*. World Bank - Trends in Solid Waste Management. URL: `https://datatopics.worldbank.org/what-a-waste/trends-in-solid-waste-management.htm`.

Frost, Sarah et al. (2019). "Compostnet: An image classifier for meal waste". In: *2019 IEEE Global Humanitarian Technology Conference (GHTC)*. IEEE, pp. 1–4.

Grandini, Margherita, Enrico Bagli, and Giorgio Visani (2020). "Metrics for multi-class classification: an overview". In: *arXiv preprint arXiv:2008.05756*.

Gupta, Tanya et al. (2022). "A deep learning approach based hardware solution to categorise garbage in environment". In: *Complex & Intelligent Systems*, pp. 1–24.

Hossin, Mohammad and Md Nasir Sulaiman (2015). "A review on evaluation metrics for data classification evaluations". In: *International journal of data mining & knowledge management process* 5.2, p. 1.

Jin, Shoufeng et al. (2023). "Garbage detection and classification using a new deep learning-based machine vision system as a tool for sustainable waste recycling". In: *Waste Management* 162, pp. 123–130.

Majchrowska, Sylwia et al. (2022). "Deep learning-based waste detection in natural and urban environments". In: *Waste Management* 138, pp. 274–284.

Mao, Wei-Lung et al. (2022). "Deep learning networks for real-time regional domestic waste detection". In: *Journal of Cleaner Production* 344, p. 131096.

Mishra, Sushruta et al. (2022). "Prioritized and predictive intelligence of things enabled waste management model in smart and sustainable environment". In: *PloS one* 17.8, e0272383.

Moolayil, Jojo and Jojo Moolayil (2019). "Tuning and deploying deep neural networks". In: *Learn Keras for Deep Neural Networks: A Fast-Track Approach to Modern Deep Learning with Python*, pp. 137–159.

Nanos, Georgios (2023). *Deep Neural Networks Padding in Convolutional Neural Networks*. Source from Baeldung website. URL: `https://www.baeldung.com/cs/deep-neural-networks-padding`.

Putra, Fredi Ganda et al. (2023). "What are the advantages of using leftover cooking oil waste as an aromatherapy candle to prevent pollution?" In: *Jurnal Inovasi Dan Pengembangan Hasil Pengabdian Masyarakat* 1.2, pp. 59–63.

Rishma, G and R Aarthi (2022). "Classification of waste objects using deep convolutional neural networks". In: *ICDSMLA 2020: Proceedings of the 2nd International Conference on Data Science, Machine Learning and Applications*. Springer, pp. 533–542.

— (2022). "Classification of Waste Objects Using Deep Convolutional Neural Networks". In: *ICDSMLA 2020*. Ed. by Amit Kumar, Sabrina Senatore, and Vinit Kumar Gunjan. Singapore: Springer Singapore, pp. 533–542. ISBN: 978-981-16-3690-5.

Shi, Cuiping et al. (2021). "A waste classification method based on a multilayer hybrid convolution neural network". In: *Applied Sciences* 11.18, p. 8572.

Sidharth, R et al. (2020). "Deep learning based smart garbage classifier for effective waste management". In: *2020 5th International Conference on Communication and Electronics Systems (ICCES)*. IEEE, pp. 1086–1089.

Vujović, Ž et al. (2021). "Classification model evaluation metrics". In: *International Journal of Advanced Computer Science and Applications* 12.6, pp. 599–606.

Wang, Hao (2020). "Garbage recognition and classification system based on convolutional neural network vgg16". In: *2020 3rd International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)*. IEEE, pp. 252–255.

www.afrik21.africa (2023). $shutterstock_7 66623799 - 1$. Image source from Afrik21 website. URL: `https://www.afrik21.africa/wp-content/uploads/2020/05/shutterstock766623799-1.jpg`.

Yu, Shiqi, Sen Jia, and Chunyan Xu (2017). "Convolutional neural networks for hyperspectral image classification". In: *Neurocomputing* 219, pp. 88–98.