Visvesvaraya Technological University

JnanaSangama, Belagavi – 590018, Karnataka

A Mini Project Report

on

"GARBAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK"

Submitted in partial fulfillment of the requirement for

Mini Project (20CSEP69) of VI semester

Bachelor of Engineering

in

Computer Science and Engineering

Submitted By

VIKAS K R	**1GA20CS168**
YASHWITH M CHAVAN	**1GA20CS172**

Under the Guidance of

Yashaswini K

Associate Professor

Dept. of CSE

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

(Accredited by NBA 2022-2025)

Rajarajeshwari Nagar, Bengaluru – 560 098

2022-2023

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

(Accredited by NBA 2022 - 2025)

Rajarajeshwari Nagar, Bengaluru – 560 098

CERTIFICATE

This is to Certify that VI Semester Mini Project Entitled **"GARBAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK"** carried out by **Mr. VIKAS K R and Mr. YASHWITH M CHAVAN** bearing USN **1GA20CS168 and 1GA20CS172 respectively** is submitted in partial fulfilment for the award of the **Bachelor of Engineering** in Computer Science and Engineering during the year **2022-2023**. The Mini Project report has been approved as it satisfies the academic requirements in respect of the mini project work prescribed for the said degree.

Yashaswini k
Associate Professor,
Dept of CSE,
GAT,Bengaluru.

Dr.Kumaraswamy S
Professor & HOD,
Dept of CSE,
GAT,Bengaluru.

External Exam

Name of the Examiners	Signature with date
1. _____	_____
2. _____	_____

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

(Accredited by NBA 2022 - 2025)

Rajarajeshwari Nagar, Bengaluru – 560 098

DECLARATION

We, **VIKAS K R** and **YASHWITH M CHAVAN** bearing USN **1GA20CS168** and **1GA20CS172**, students of sixth Semester B.E, Department of Computer Science and Engineering, Global Academy of Technology, Rajarajeshwari Nagar Bengaluru, declare that the Mini Project entitled **"GARBAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK"** has been carried out by us and submitted in partial fulfillment of the course requirements for the award of degree in Bachelor of Engineering in Computer Science and Engineering from Visvesvaraya Technological University, Belagavi during the academic year **2022-2023**.

Place: Bengaluru

Date**:**09-03-2023

VIKAS K R (1GA20CS168)

YASHWITH M CHAVAN(1GA20CS172)

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance crowned our efforts with success.

We consider ourself proud, to be part of **Global Academy of Technology** family, the institution which stood by the way in endeavors. We express our deep and sincere thanks to our Principal **Dr. N. Ranapratap Reddy** for his support.

We are appreciative of **Dr. Kumarswamy S, Professor and HOD**, Department of CSE, who serves as a source of motivation and provides priceless assistance in directing my efforts in the right path.

We are grateful to **Yashaswini K, Associate Professor**, Dept of CSE who is source of inspiration and of invaluable help in channelizing my efforts in right direction. And also, guiding and correcting various documents of mine with attention and care. He has taken lot ofpain to go through the document and make necessary corrections as and when needed.

We would like to thank the faculty members and supporting staff of the Department of CSE, GAT for providing all the support for completing the Project work.

Finally, we are grateful to our parents and friends for their unconditional support and help during the course of our Project work.

YASHWITH M CHAVAN(1GA20CS172)

VIKAS K R (1GA20CS168)

ABSTRACT

The buildup of solid waste in metropolitan areas is a major worry that, if not effectively handled, might lead to environmental contamination and be dangerous to human health.

To manage a range of waste products, it's crucial to have an advanced/intelligent waste management system. The act of separating garbage into its many components is one of the most crucial parts of waste management, and it is often carried out manually by hand-picking.

To simplify this, we suggest a customized architecture for multiclass Convolutional Neural Networks (CNNs). The design includes elements that are individual-specific, enabling the model to adjust to the traits of each class.

Our CNN architecture outperforms conventional CNN models like ResNet50 and VGG16 in multiclass classification problems by utilizing this personalized approach.

The effectiveness and resilience of our suggested architecture are demonstrated by experimental results on our dataset, opening the path for increased classification accuracy of 73.50%.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Garbage classification has emerged as a critical aspect of waste management and environmental sustainability. With the escalating global waste generation and its detrimental impact on the ecosystem, efficient garbage classification methods have become imperative to address the challenges of waste disposal and recycling. The improper management of waste not only contributes to pollution and habitat degradation but also depletes natural resources and exacerbates climate change.

Traditionally, waste disposal practices have been rudimentary, relying on manual sorting techniques that are time-consuming, labor-intensive, and prone to errors. In recent years, advancements in technology, particularly in the field of artificial intelligence and computer vision, have opened up new possibilities for automating waste sorting processes. Among these technological innovations, Convolutional Neural Networks (CNNs) have shown remarkable promise in revolutionizing garbage classification

1.1 MOTIVATION

According to the Central Pollution Control Board of Delhi's annual report for 2020–21 [2], the nation generates 160038.9TPD of solid waste, of which 95.4% is collected and only 50% is treated; the remaining 18.4% is dumped, and the remaining 31.7% of the total waste generated goes unnoticed.

In recent years some countries have begun to investigate recycling tactics in a new sort of cyclical economy for sustainable growth that enhances environmental quality.

The most common method for classifying garbage is manual sorting since it is currently the most precise one. Sadly, it takes a lot of time and requires experienced operators, which severely limits the rate of categorization of waste.

Therefore, an automated trash categorization solution is required to meet this expanding problem therefore we selected this problem to tackle

1.2 OBJECTIVES OF THE PROJECT

- Develop an Efficient CNN Model: Create a convolutional neural network (CNN) architecture tailored for garbage classification that can efficiently recognize and classify various waste materials, including

plastics, paper, glass, metals, and organic waste.

- Achieve High Classification Accuracy: Train the CNN model to achieve high accuracy in classifying different waste categories, ensuring reliable and precise waste sorting in real-world applications.
- Enhance Real-Time Performance: Optimize the CNN model to process garbage classification tasks in real-time, enabling quick and seamless waste sorting and management in practical scenarios.
- Address Imbalanced Datasets: Implement data augmentation techniques and strategies to handle imbalanced garbage datasets, ensuring fair representation and accurate classification of all waste categories.
- Reduce Hardware Requirements: Design the CNN model to be resource-efficient, reducing the hardware requirements and enabling deployment on low-power devices or edge computing platforms for cost-effective garbage classification systems.
- Customize for Local Waste Contexts: Develop a CNN architecture that can be easily customized and adapted to specific local waste compositions and disposal practices, enhancing the system's adaptability across different regions.
- Incorporate Transfer Learning: Utilize transfer learning techniques to leverage pre-trained CNN models, enabling faster convergence and better performance even with limited training data.
- Integrate with IoT and Sensing Technologies: Explore the integration of CNN-based garbage classification with Internet of Things (IoT) sensors and smart waste bins to enable real-time data collection, monitoring, and optimization of waste management processes.

By pursuing these objectives, the CNN-based garbage classification system can contribute significantly to more efficient waste management, promote recycling initiatives, and reduce environmental pollution, ultimately fostering a cleaner and more sustainable environment

1.3 PROJECT REPORT OUTLINE

The rest of this report is structured as follows: chapter 4 evaluates the work designs, highlighting their advantages and disadvantages. Chapter 2 presents the literature survey. All the experimental information, findings, and analyses are presented in chapter 5 and chapter 6 respectively. The paper is summed up in the final portion.

CHAPTER 2

LITERATURE SURVEY

Today, several automated methods for classifying waste have been suggested. These methods may be divided into three categories: mechanical methods (MAs), Internet of Things methods (ITAs), and artificial intelligence methods (AIAs).

To successfully replace human garbage sorting, an MA uses microprocessors, external sensors, and mechanical gearbox in an automatic garbage categorization system. However, because to low-accuracy recognition, this method does not produce the intended trash categorization result [3]– [5].

A unique ITA technique for automated trash categorization was put out to solve this issue. A more automated and accurate waste categorization system was created using the ITA and a cloud server, however owing to the high price and intricate design of the system, it is challenging to deploy and maintain [6]– [9].

Artificial intelligence (AI)-based AIAs for waste categorization are extremely accurate, adaptable, and durable. To perform trash detection and classification tasks, a prediction model trained on garbage data might be used. However, the current classification algorithms do not meet the requirements of trash classification systems and run on high-performance servers or PCs [10]– [13].

This study suggests a brand-new deep learning-based embedded Linux system for intelligent garbage categorization. The system combines an integrated Linux platform for real-time processing with a convolutional neural network (CNN) model for picture categorization. [14]
Gives An overview of garbage detection and categorization approaches utilising deep learning algorithms is given in this survey study. It explores several methods for classifying garbage, including CNNs, and shows both their benefits and drawbacks.[15].

The study mainly concentrated on garbage identification utilising aerial hyperspectral data, which may not be easily transferrable to other circumstances or image-based methods for waste categorization [16].
Due to the benchmark datasets and the proposed garbage image recognition method's focus on domestic waste, its application to other waste categorization domains may be constrained [17].

The deep learning framework CGBNet, created exclusively for compost categorization, is introduced in this research. It may not address other waste types or management issues because it only classifies compost [18]. An artificial neural network-based method for predicting the production of municipal solid trash is presented in this paper. It does not particularly address trash classification [19].

The research focuses on utilising machine learning approaches to categorise rubbish according to its suitability for recycling. It offers insightful information on recycling but skips over other waste management or deep learning topics [20].

The idea of very deep convolutional networks (VGGNet) for extensive image recognition is introduced in this fundamental paper. Despite not being specifically for garbage categorization, it offers a basis for deep learning techniques applied in this area [21].

The previously mentioned references have various limitations, even if they offer insightful contributions to the subject of intelligent waste categorization systems employing deep learning. By considering several data modalities and expanding the evaluation to more complex and varied circumstances, our study addresses these constraints.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

3.1 Hardware Requirements

- System : Intel i3 or above
- Hard Disk : 120 GB or above
- Monitor : 15'' LED or above
- Input Devices : Keyboard, Mouse
- Ram : 8 GB or above

3.2 Software Requirements

- Operating system : Windows 10 or above.
- Coding Language : PYTHON 3.6 or above
- Tools : Google Collab

CHAPTER 4

DESIGN

It is advisable to go with a new CNN as the current CNN is unable to control the difficulties in real-garbage detection analysis, this is due to the need to find a reliable and accurate model for classification.

4.1 HIGH LEVEL DESIGN

We developed the dataset, which contains 6 garbage categories and a total of 2521 images, of which 2019 images were used for training and 502 for testing, to promote the application of deep learning in the field of environmental protection and improve the performance of the object recognition algorithm for garbage images. The dataset's trash has the following characteristics: fixed shape, variable shape, constant texture, variable texture, and various scales. We collected each picture and gave it a labelled and the fig 1-6 are the sample images available in the dataset.

Fig. 1. Cardboard

Fig. 2. Glass

Fig. 3. Metal

Fig. 4. Paper

Fig. 5. Plastic

Fig. 6. Trash

Due to the lesser size of each class, data augmentation techniques were used on each image. Rescale, Shear_range of (0.2), Zoom_range (0.4), and Horizantal Flip were some of these methods.

Fig. 7 is a presentation of the precise information abotut dataset and Table. 1 is the amount of images in each category of waste. And number of images for training and testing.

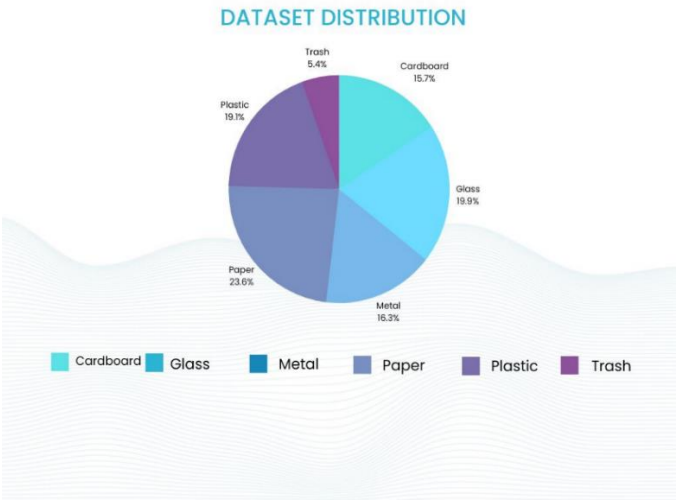


Fig.7.Dataset distribution

Table 1 . Number of images in the training and testing of each category of waste

Class	Training	Testing
Cardboard	322	75
Glass	400	101
Metal	328	82
Paper	475	119
Plastic	385	97
Trash	109	28

4.2 DETAILED DESIGN

In this part, in Fig. 8 we describe our model architecture in depth and go over the training process.

Deep learning models have significantly advanced picture classification tasks in recent years, revolutionizing the area of computer vision. Convolutional Neural Networks (CNNs) have become an effective method for removing significant characteristics from pictures and obtaining cutting-edge results in a variety of visual identification applications. In this study, we provide a brand-new CNN architecture for precise and effective picture categorization into multiple classes.

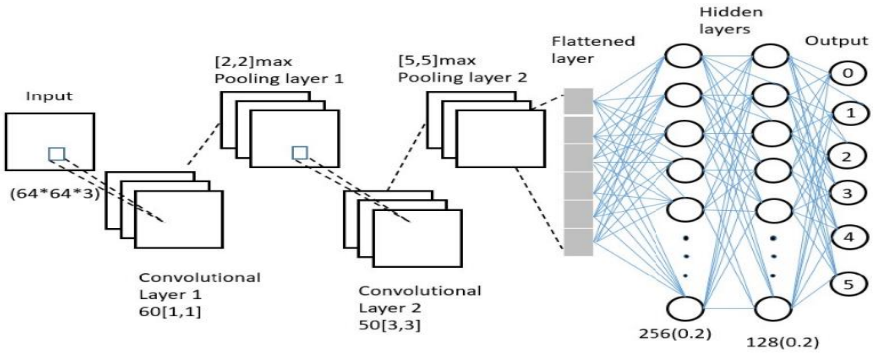


Fig 8. The architecture of the proposed model

Input Layer: pictures having the shape (64, 64, 3) are supplied to the input layer; these pictures have a width and height of 64 pixels and three RGB color channels. For jobs involving picture categorization, this input size is common.

Convolutional Layers: 60 filters with a size of (1, 1) are applied by the first convolutional layer (Conv2D). These filters assist in removing minute details from the input photos. Then Rectified Linear Unit (ReLU) activation function, which adds non-linearity to the network and enhances its capacity to recognize complex patterns. Following the first convolutional layer comes a MaxPooling2D layer with a pool size of (2, 2). It conducts a down sampling technique, cutting the feature maps' spatial dimensions in half while keeping the most important information.

In the second convolutional layer, 50 filters with a (3, 3) pixel size are applied. In comparison to the data recorded by the first layer, these filters aid in the extraction of higher-level features that are more complicated and abstract. After the second convolutional layer MaxPooling2D layer with a pool size of (5, 5). In doing so, the spatial dimensions are further reduced, and the most important data from the feature maps are captured.

Flattening Layer: The preceding layer's 2D feature maps are converted into 1D vectors using the flattening layer. To link the convolutional layers to the fully linked layers, this flattening process is required.

Fully Connected Layers: A first dense layer with 256 units is present. By collecting abstract representations of the input pictures, it performs the function of a high-level feature extractor.
After the initial dense layer, a Dropout layer with a rate of 0.2 is added to prevent overfitting. Dropout forces the model to acquire more robust and generalized characteristics by randomly deactivating certain neurons during training.
The learnt characteristics are further refined in the second dense layer, which has 128 units.
The second dense layer is followed by a Dropout layer with a rate of 0.2, which adds more regularization.

Output Layer: Six units make up the ultimate dense layer, the same number of waste classification groups. Each unit stands for a certain type of garbage.
The output layer is given the SoftMax activation function, which results in a probability distribution over the classes. As a result, the model can give each waste category a probability value, indicating how likely it is that the input image belongs to that class.

Overall, this architecture blends fully connected layers for high-level feature processing and classification with convolutional layers for feature extraction. The model can learn discriminative features, decrease overfitting, and enhance generalization with the use of ReLU activation functions, max pooling, and dropout layers. This model can be taught to categorize fresh garbage photographs reliably and automatically by training on a huge dataset of labelled waste images.

CHAPTER 5

IMPLEMENTATION

5.1 IMPLEMENTATION OF CNN

1. Importing Required Libraries:

from keras.models import Sequential

from keras.layers import Conv2D

from keras.layers import Flatten

from keras. layers import MaxPooling2D

from keras.layers import Dense

import numpy as np

from keras.layers.regularization.dropout import Dropout

The code begins by importing the necessary libraries from Keras, including the Sequential model, various layer types (Conv2D, Flatten, MaxPooling2D, Dense, and Dropout), and numpy for array operations.

2. Creating the Model Architecture:

```
model=Sequential()
model.add(Conv2D(60,(1,1),input_shape=(64,64,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(50,(3,3),input_shape=(64,64,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(5,5)))
model.add(Flatten())
model.add(Dense(units=256,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(units=128,activation='relu'))
model.add(Dropout(0.2))
```

```
model.add(Dense(units=6,activation='softmax'))
```

The first layer is a 2D convolutional layer (Conv2D) with 60 filters of size (1, 1) and a ReLU activation function. This layer processes the input images, which have a shape of (64, 64, 3) (64x64 pixels with 3 color channels: red, green, and blue). After the first convolutional layer, a max-pooling layer

(MaxPooling2D) is added with a pool size of (2, 2). This layer helps reduce the spatial dimensions of the output from the previous convolutional layer.

3. Model Compilation:

```
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

In this code, X_train and y_train are the training data and corresponding labels, X_val and y_val are the validation data and labels (if you have a validation set), num_epochs is the number of epochs you want to train the model for, and batch_size is the number of samples per gradient update.

4. Data Preprocessing:

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range = 0.4,horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
training_set =train_datagen.flow_from_directory('/content/drive/MyDrive/dataset/training_set',target_size = (64, 64),batch_size = 40,class_mode = 'sparse')
test_set = test_datagen.flow_from_directory('/content/drive/MyDrive/dataset/test_set',target_size = (64, 64),batch_size = 4,class_mode = 'sparse')
```

ImageDataGenerator to perform real-time data augmentation on your image dataset. This is a powerful technique that helps improve the performance of your model and prevents overfitting by creating variations of the images during training.

5. Loading Training and Test Data:

```
history=model.fit(training_set,steps_per_epoch = 50,epochs = 80,validation_data=test_set, validation_steps=5)
```

training_set: This is the ImageDataGenerator object you created for the training data. It will generate batches of augmented data from the training image files.

steps_per_epoch: The number of steps (batches) to process from the training data generator for each epoch. In this case, you've set it to 50, meaning the model will process 50 batches from the training data during each epoch

6. Model Training:

```
test_loss, test_accuracy = model.evaluate(test_set )
print(f'Test Loss: {test_loss}')
print(f'Test Accuracy: {test_accuracy}')
```

It appears that the code you provided is using a machine learning model with the fit function. The fit function is commonly used in various machine learning libraries like TensorFlow or Keras to train a model on a dataset.

7. Making Predictions:

```
image_path = '/content/drive/MyDrive/dataset/test_set/paper_test/paper487.jpg'
test_image =image.load_img(image_path,target_size = (64, 64))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis = 0)
result = model.predict(test_image)
print(result)
```

This code snippet demonstrates how to load a single test image, prepare it for input to the model, and use the model to predict the class probabilities for that image. The probabilities represent the model's confidence in its predictions for each class.

8. Displaying Results:

```
import matplotlib.pyplot as plt
from PIL import Image
image_data = plt.imread(image_path)
plt.imshow(image_data)
plt.axis('off')  # Turn off axis labels and ticks
plt.show()
res=result[0]
val=max(res)
ind = np.where(res == val)[0]
class_names = ['Cardboard', 'glass', 'metal', 'paper','plastic','trash']  # Replace with your class names
predicted_class_name = class_names[int(ind)]
print(predicted_class_name)
```

Overall, this code snippet is a common way to convert the model's output probabilities into a human-readable class prediction by finding the class with the highest probability and matching it to its corresponding class name. Keep in mind that this code assumes that the classes in the class_names list are in the same order as the model's output.

9.Evaluating model performance:

```
test_loss, test_accuracy = model.evaluate(test_set )
print(f'Test Loss: {test_loss}')
print(f'Test Accuracy: {test_accuracy}')
```

When you run this code snippet, it will output the test loss and test accuracy of the model on the provided test dataset. This information is crucial for assessing how well the model generalizes to new, unseen data.

5.2 IMPLEMENTATION OF TRANSFER LEARNING

1. Importing Required Libraries:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, Flatten, MaxPooling2D, Dense, Dropout
from tensorflow.keras.applications import ResNet50, VGG16
```

The code imports the necessary modules from TensorFlow Keras, including the Sequential model and various layers (Conv2D, Flatten, MaxPooling2D, Dense, Dropout) for building custom CNNs, and also the pre-trained models ResNet50 and VGG16 from the Keras Applications library.

2. Building ResNet50:

```
model=Sequential()
pre_model=tf.keras.applications.ResNet50(
    include_top=False,
    weights="imagenet",
    input_shape=(64,64,3),
    pooling=max,
    classes=6,
)
for layer in pre_model.layers :
  layer.trainable=False
model.add(pre_model)
model.add(Flatten())
model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=6,activation='softmax'))
model.summary()
```

The code you provided creates a new Sequential model and adds layers from a pre-trained ResNet50 model, followed by additional layers for fine-tuning. Let's go through the code step by step:

model=Sequential(): This line creates a new sequential model

pre_model=tf.keras.applications.ResNet50(...): This section creates a ResNet50 model using tf.keras.applications.ResNet50. The include_top parameter is set to False to exclude the final fully connected layers of the ResNet50 model, weights is set to "imagenet" to load pre-trained weights from the ImageNet dataset, input_shape is set to (64, 64, 3) to specify the expected input shape for the model, pooling is set to "max" to use global max pooling, and classes is set to 6 to specify the number of classes in the new classification task.

3. Building VGG16:

```
model=Sequential()
pre_model=tf.keras.applications.VGG16(
include_top=False,
weights="imagenet",
input_shape=(64,64,3),
pooling=max,
classes=6,
)
for layer in pre_model.layers :
 layer.trainable=False
model.add(pre_model)
model.add(Flatten())
model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=6,activation='softmax'))
model.summary()
```

summary of the model's architecture, which should include the VGG16 layers (with frozen weights), followed by the additional layers you added for fine-tuning and classification. This architecture can be used

for transfer learning on a new classification task with 6 classes.

CHAPTER 6

TESTING

The experiments involve the subsequent steps:

1) Evaluation of Optimizers in Comparison

This experiment compares the effectiveness of various optimizers for CNN-based helmet identification. Several optimizers, including Adam, SGD, RMSprop, Adadelta, and Adagrad, are used to train the CNN model. Each optimizer has its own set of hyperparameters, including learning rate, batch size, and 50 epochs. Following that, the models are assessed using the testing dataset to gauge their effectiveness in terms of testing accuracy and testing loss. To choose the best optimizer for the task of headgear detection, the results are noted and examined.

2) Dropout Research

The performance of the headgear detection model is being examined in this experiment in order to determine the effect of dropout regularisation. The CNN model is trained both with and without dropout regularisation using the optimizer found to be the most efficient in Comparative Analysis of Optimizers. The accuracy of the models' performances is contrasted. This investigation sheds light on the significance of this regularisation method for headgear identification by evaluating the impact of dropout regularisation on the model's capacity to generalise and avoid overfitting.

3) Analysis of Transfer Learning

In this experiment, pre-trained CNN models for headgear detection are used in conjunction with transfer learning approaches. Particularly, pre-trained models are applied to well-known architectures as AlexNet, ResNet50, and VGG16. These models are polished using the headgear detection dataset, and the testing dataset is used to gauge how well they work. Accuracy is one of the parameters for evaluation. The effectiveness of various pre-trained architectures for headgear identification can be understood by contrasting the transfer learning model outcomes.

4) Integrated Analysis

This experiment attempts to assess the performance of a combined strategy by building on the findings from Comparative Analysis of Optimizers and Transfer Learning Analysis. The best-performing architecture from the transfer learning analysis is used to implement transfer learning once the CNN model has been trained using the best optimizer found in the comparative analysis of optimizers. The testing dataset is then used to evaluate the combined model, and the findings are contrasted with those from earlier tests. This investigation sheds light on how the optimal optimizer and transfer learning affect the accuracy of headgear identification and other evaluation criteria.

5)Robustness Evaluation

In this step, additional tests are carried out to evaluate the resilience of the models. There are a variety of hyperparameters in the CNN design, including learning rate, batch size, and number of layers. On many dataset modifications, including various lighting conditions, angles, and resolutions, the models' performance is assessed. The most reliable method for headgear identification can be found by evaluating the models' performance in various settings.

The findings of these tests offer a thorough grasp of the comparison of optimizers and transfer learning techniques for headgear detection while employing a CNN. They give light on various pre-trained architectures' performances, the impact of dropout regularisation, the efficiency of various optimizers, and the possible advantages of combining optimisation with transfer learning for headgear identification.

CHAPTER 7

RESULTS

This section compares the results of our custom-built model employing convolutional neural network (CNN) on our specific problem domain and dataset with different transfer learning techniques, namely ResNet50 and VGG16.

The training loss and accuracy graphs for three models—ResNet50, VGG16, and our own proposed model are shown in Fig. 9-11 in this section. These graphs show the models' development as learners and can be used to evaluate how well they Train.

The training loss graph shows how the loss function changes during training. It demonstrates how the models get better at reducing mistakes over time. Reduced loss and enhanced convergence are shown by lower values on the y-axis.

The training accuracy graph displays the accuracy of the models on the training data over time. The capacity of the models to produce accurate predictions as they learn from the training data is shown by the higher numbers on the y-axis.

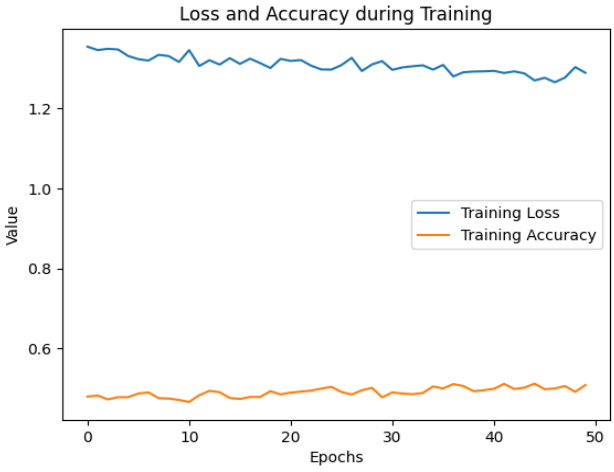


Fig. 9. Loss and accuracy during Training of ResNet50

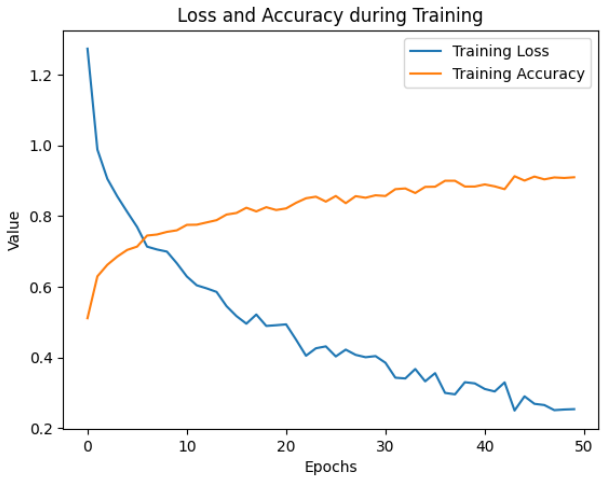


Fig. 10. Loss and accuracy during training of VGG16

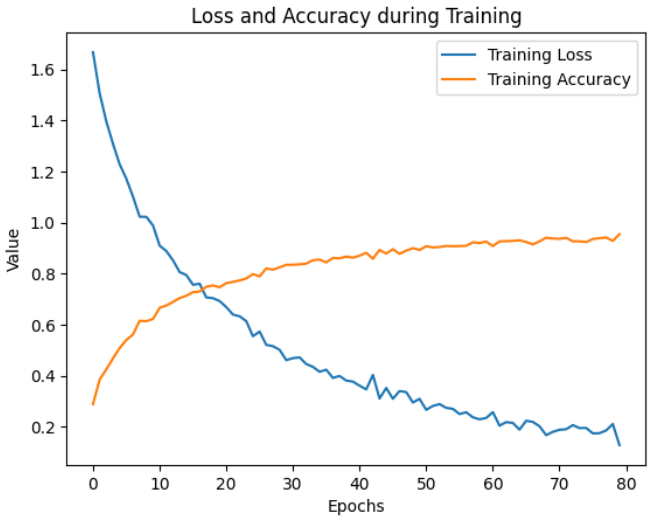


Fig. 11. Loss and accuracy during Training proposed model`

Three models—ResNet50, VGG16, and our own proposed model—training loss and accuracy graphs are shown in above Figures respectively. The loss and accuracy values are represented on the y-axis, while the number of training iterations or epochs is represented on the x-axis.

The training loss is represented by the blue line, which demonstrates how the loss values of the models decrease during the training cycles. The objective is to spot a declining trend, which denotes enhanced convergence and model performance.

The training accuracy is shown by the orange line, which throughout the process shows the accuracy values of the models on the training data. Our goal is to spot a rising trend that shows how the models get better at making predictions as the training goes on.

We can learn more about the training behaviours of models by contrasting the training loss and accuracy graphs of those models. These graphs give a complete picture of the learning ability of the models and emphasise variances in convergence, overfitting, or underfitting tendencies.

The accuracy and loss values of three distinct image classification models—ResNet50, VGG16, and a proposed model—are shown in Table 2. The same dataset is used to measure these models' accuracy in classifying images.

Table.2. Accuracy and Loss Table

MODEL	ACCURACY	LOSS
ResNet50	0.4462	1.4873
VGG16	0.6992	1.3818

The ResNet50 model obtained a greater test loss of 1.4873 and a significantly lower test accuracy of 44.62%. This suggests that, in contrast to the other models included in this study, the model had difficulty correctly classifying pictures. To increase its performance on a particular dataset The ResNet50 architecture may need more fine-tuning or changes because of its well-known deep and complicated structure.

The VGG16 model demonstrated a higher test accuracy of 69.92% and a test loss of 1.3818. This indicates that VGG16 performed better than ResNet50, but there is still room for improvement. VGG16 is a well-known and widely used architecture, but its performance may vary depending on the dataset and problem domain. Fine-tuning or hyperparameter optimization techniques can potentially enhance its accuracy further.

The custom-built model showcased the test accuracy of 73.50% highest among the evaluated models. It achieved a test loss of 1.0192, indicating a relatively lower error rate compared to ResNet50 and VGG16. The custom-built model's superior performance suggests that it was able to capture and leverage the dataset's specific characteristics effectively. This result highlights the potential advantages of designing a model tailored to the problem domain.

Confusion Matrix Representation:

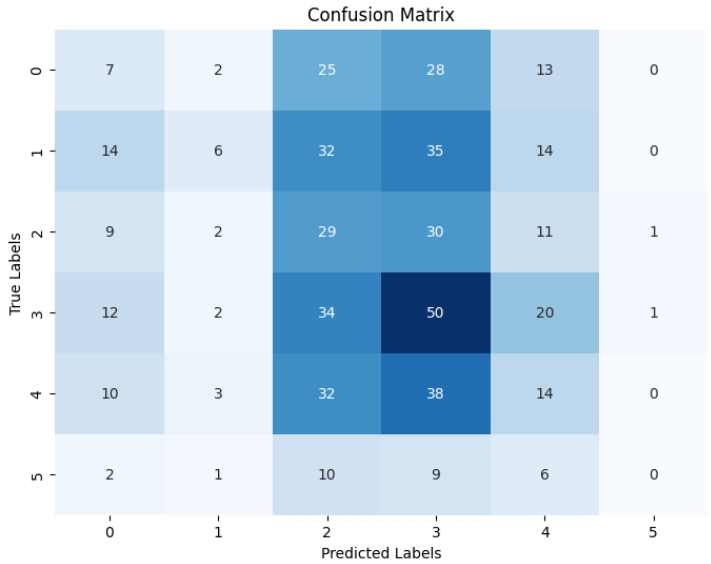


Fig. 12. Confusion matrix of ResNet50

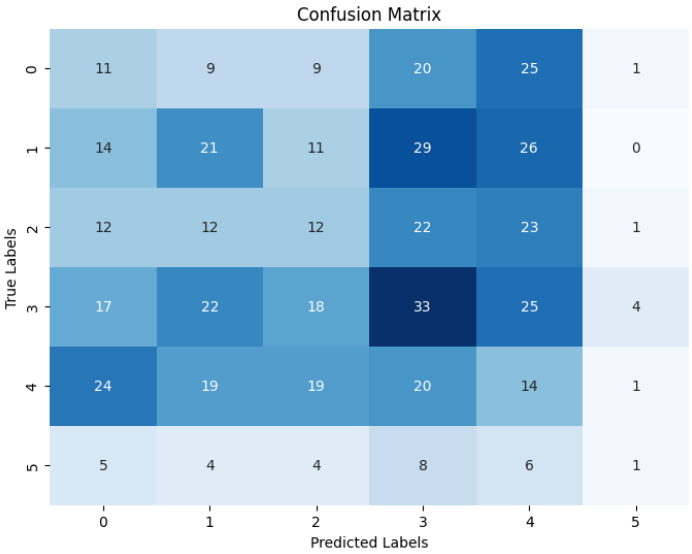


Fig. 13. Confusion matrix of VGG16

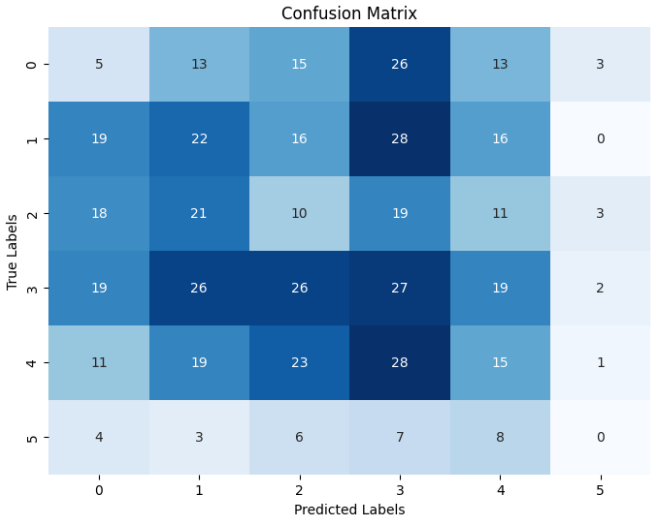


Fig. 14. Confusion matrix of proposed Model

In Figures 12, 13, and 14, we present the confusion matrices for ResNet50, VGG16, and our custom-built model, respectively. Each matrix provides a visual representation of the model's performance in classifying the test data. The matrix elements correspond to the number of instances classified correctly (true positives and true negatives) or incorrectly (false positives and false negatives) for each class.

To improve the performance of our proposed model, we conducted a series of experiments involving different optimizers and varying dropout ratios. This was done as our model consistently outperformed established architectures such as ResNet and VGG16. We wanted to explore if fine-tuning the optimization process and adjusting the dropout ratio could further enhance our model's accuracy. By employing optimizers like Adam, SGD, ADAGRAD, ADADELTA and RMSprop with varying dropout ratio, we were able to observe their impact on training convergence and overall performance. Additionally, we adjusted the dropout ratio, which controls the regularization strength and prevents overfitting, to find an optimal value. To visualize the results,

we plotted the corresponding accuracy readings on a bar graph in Fig. 15-16, allowing us to compare the different optimizer-dropout combinations and identify the most effective configuration for our proposed model.

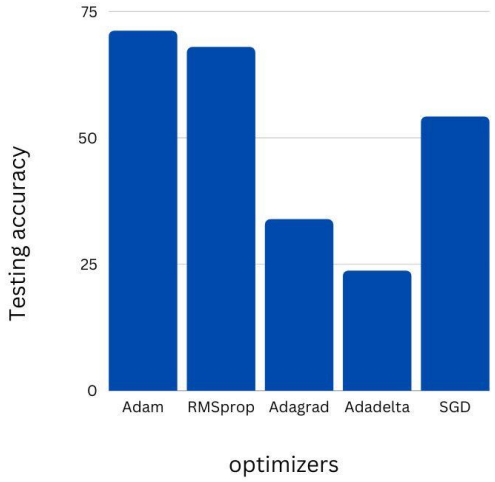


Fig.15.accuracy of proposed-model for different optimizer

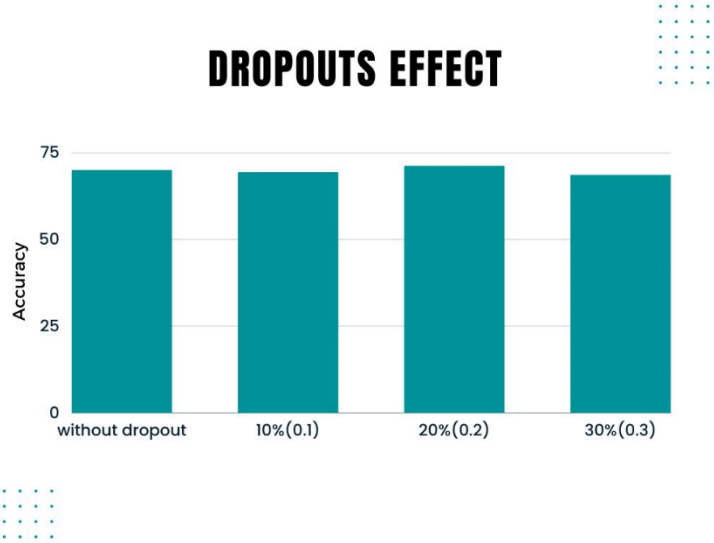


Fig.16. effect of optimizers on accuracy of model

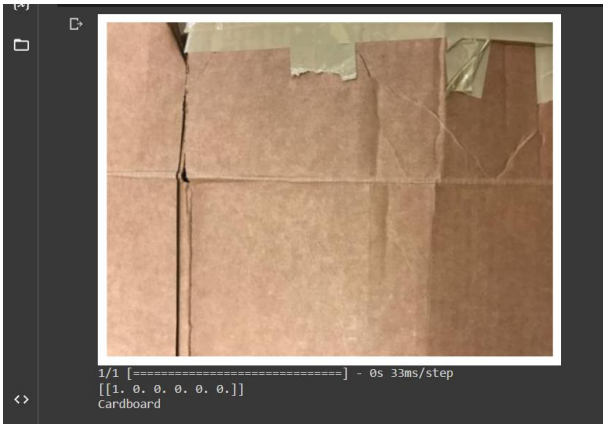
7.1 SNAPSHOTS

```
Model Evaluation

[28]  1  test_loss, test_accuracy = model.evaluate(test_set )
      2  print(f'Test Loss: {test_loss}')
      3  print(f'Test Accuracy: {test_accuracy}')

    126/126 [==============================] - 3s 23ms/step - loss: 1.4020 - accuracy: 0.7351
    Test Loss: 1.401991844177246
    Test Accuracy: 0.7350597381591797
```

Fig.17: Accuracy of the model



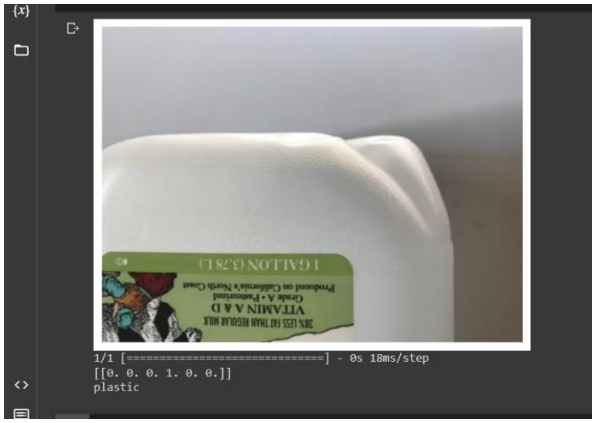


Fig.18:Model classifying each class of waste

CHAPTER 8

CONCLUSION

In conclusion, the custom-built model outperformed ResNet50 and VGG16 in terms of test accuracy. The variation in performance underscores the importance of selecting the appropriate model architecture and it for the specific dataset and problem domain.

We suggest a waste categorization system that uses deep learning to distinguish various waste components. This technology may be used to categorise garbage automatically, lowering the need for human involvement and decreasing pollution and illness.

We obtained an accuracy of 73.50% from the results when tested against the garbage dataset. Using our approach, the garbage will be separated more quickly and intelligently, maybe even with less input from humans.

The accuracy of the system may be increased by including additional images in the dataset. By changing some of the current settings, we will eventually enhance our system so that it can classify more trash items.

CHAPTER 9

LIMITATIONS

The model for garbage detection using CNN in the provided code has some limitations. Let's explore these limitations:

1. Accuracy and False Positives/Negatives: Garbage detection models may not always be perfectly accurate. They might misclassify certain objects as garbage or fail to identify some pieces of litter, leading to false positives or false negatives.

2. Dependence on Quality of Training Data: The accuracy of garbage detection models heavily relies on the quality and diversity of the training data. If the training data does not encompass a wide range of garbage types and environmental conditions, the model may not generalize well to new and unseen scenarios.

3. Generalization to Different Environments: Models trained on garbage data from specific locations may not perform as effectively when deployed in different environments with different types of waste or varying lighting conditions.

4. Speed and Resource Constraints: Some garbage detection models can be computationally intensive and require significant processing power, making real-time detection challenging on certain devices or in remote areas with limited resources.

5. Limited Detection Range: Computer vision models have limitations in the range of objects they can detect. Very small or heavily occluded pieces of garbage might be challenging to identify.

6. Detection of Hazardous or Uncommon Waste: Garbage detection models may struggle to identify hazardous waste or uncommon objects that do not resemble typical trash items.

CHAPTER 10

FUTURE ENHANCEMENTS

To enhance the garbage detection model using CNN, we can consider several improvements and future enhancements. Here are some suggestions:

1. Advanced Sensor Technologies: Incorporate advanced sensor technologies, such as hyperspectral imaging, LIDAR, or multi-spectral cameras, to improve the accuracy and efficiency of garbage detection. These sensors can provide more detailed and comprehensive data, enabling better identification of different types of waste.

2. Artificial Intelligence and Machine Learning Improvements: Continue to refine and enhance the machine learning algorithms used for garbage detection. Explore newer models and techniques to improve the accuracy, speed, and adaptability of the system, allowing it to recognize and classify garbage more effectively.

3. Integration with Waste Management Systems: Integrate garbage detection technology with existing waste management systems to streamline the entire waste collection and disposal process. This could involve automatic notification to waste collection agencies or generating waste pickup schedules based on the data collected.

4. Automated Sorting and Recycling: Expand the application of garbage detection to include automated sorting and recycling processes. Develop robotic systems that can efficiently sort different types of waste materials for recycling purposes.

Remember that technological advancements are continuously evolving, so these ideas may already be in progress or have seen further developments beyond my last update. Nevertheless, these suggestions can serve as a starting point for enhancing garbage detection projects in the future.

REFERENCES

[1] Kaza, Silpa; Yao, Lisa C.; Bhada-Tata, Perinaz; Van Woerden, Frank. 2018. What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050. Urban Development;. © Washington, DC: World Bank. http://hdl.handle.net/10986/30317 License: CC BY 3.0 IGO.

[2] CENTRAL POLLUTION CONTROL BOARD DELHI, ANNUAL REPORT 2020-21, Website: www.cpcb.nic.in.

[3] J. Hui, ‘‘Design and implementation of intelligent garbage sorting system based on RFID,’’ J. Anhui Inst. Electron. Inf. Technol., vol. 17, no. 4, pp. 10–13, 2018.

[4] T. Zhang, M. Li, L. Li, and W. Luo, ‘‘Design of garbage classification system based on RFID,’’ J. Phys., Conf. Ser., vol. 1744, no. 2, Feb. 2021, Art. no. 022111.

[5] C. Zhou, ‘‘Design of intelligent sorting trash dustbin based on STM32,’’ in Proc. E3S Web Conf., 2020, Art. no. 04032\

[6] [6] P. Pan, J. Lai, G. Chen, J. Li, M. Zhou, and H. Ren, ‘‘An intelligent garbage bin based on NB-IOT research mode,’’ in Proc. IEEE Int. Conf.Saf. Produce Informatization (IICSPI), Dec. 2018, pp. 113–117.

[7] Y. Wang, Y. Xu, B. Zhang, J. Zhang, and X. Su, ‘‘The design and imple mentation of the smart trash can based on the Internet of Things,’’ J. Phys., Conf. Ser., vol. 1550, May 2020, Art. no. 022003.

[8] Oralhan, Z., Oralhan, B., & Yiğit, Y. (2017). Smart city application: Internet of things (IoT) technologies based smart waste collection using data mining approach and ant colony optimization. Internet Things, 14(4), 5.

[9] Lu, Zhongzhi, and Na Xu, ‘‘Application strategies of waste sorting facil ities based on Internet of Things,’’ in Innovative Computing. Singapore:Springer, 2020, pp. 1291–1296.

[10] Z. Kang, J. Yang, G. Li, and Z. Zhang, ‘‘An automatic garbage classification system based on deep learning,’’ IEEE Access, vol. 8, pp. 140019–140029, 2020.

[11] O. Adedeji and Z. Wang, ‘‘Intelligent waste classification system using deep learning convolutional neural network,’’ Proc. Manuf., vol. 35,pp. 607–612, Jan. 2019.

[12] H. Wang, ‘‘Garbage recognition and classification system based on con volutional neural network VGG16,’’ in Proc. 3rd Int. Conf. Adv. Electron.Mater., Comput. Softw. Eng. (AEMCSE), Apr. 2020, pp. 252–255.

[13] K. Yan, W. Si, J. Hang, H. Zhou, and Q. Zhu, ‘‘Multi-label garbage image classification based on deep learning,’’ in Proc. 19th Int. Symp. Distrib. Comput. Appl. Bus. Eng. Sci. (DCABES), Oct. 2020, pp. 150–153.

[14] B. Fu, S. Li, J. Wei, Q. Li, Q. Wang and J. Tu, "A Novel Intelligent Garbage Classification System Based on Deep Learning and an Embedded Linux System," in IEEE Access, vol. 9, pp. 131134-131146, 2021, doi: 10.1109/ACCESS.2021.3114496.

[15] H. Abdu and M. H. Mohd Noor, "A Survey on Waste Detection and Classification Using Deep Learning," in IEEE Access, vol. 10, pp. 128151-128165, 2022, doi: 10.1109/ACCESS.2022.3226682.

[16] D. Zeng, S. Zhang, F. Chen and Y. Wang, "Multi-Scale CNN Based Garbage Detection of Airborne Hyperspectral Data," in IEEE Access, vol. 7, pp. 104514-104527, 2019, doi: 10.1109/ACCESS.2019.2932117.

[17] Z. Wu et al., "New Benchmark for Household Garbage Image Recognition," in Tsinghua Science and Technology, vol. 27, no. 5, pp. 793-803, October 2022, doi: 10.26599/TST.2021.9010072.

[18] S. Gangopadhyay and A. Zhai, "CGBNet: A Deep Learning Framework for Compost Classification," in IEEE Access, vol. 10, pp. 90068-90078, 2022, doi: 10.1109/ACCESS.2022.3201099.

[19] Kumar, S., Gaur, A., Kamal, N., Pathak, M., Shrinivas, K., & Singh, P. (2020, April). Artificial Neural Network Based Optimum Scheduling and Management of Forecasting Municipal Solid Waste Generation–Case Study: Greater Noida in Uttar Pradesh (India). In *Journal of Physics: Conference Series* (Vol. 1478, No. 1, p. 012033). IOP Publishing.

[20] Yang, M. and Thung, G., 2016. Classification of trash for recyclability status. CS229 project report, 2016(1), p.3.

[21] TY - JOUR, AU - Simonyan, Karen, AU - Zisserman, Andrew, PY - 2014/09/04, SP - ,T1 - Very Deep Convolutional Networks for Large-Scale Image Recognition,JO - arXiv 1409.1556,ER-