Feynn Labs

A PROJECT REPORT ON

WASTE SEGREGATION USING DEEP LEARNING

Team Members

RAJEEV T M

SAMARTH RIKHE

VANUSHUL SHIIKKEWAL

CHARAN SAI GOTTIPATI

The problem statement:

Inadequate waste segregation practices lead to challenges in reducing general waste output, identifying reusable items, and recycling materials, ultimately causing a detrimental impact on the environment. Improperly sorted trade waste ends up in landfills, mimicking the same mixed state as it was in the bins. Consequently, the decomposition of waste items like food scraps, paper, and liquid waste releases harmful runoff into the soil and emits noxious gases into the atmosphere.

DATASET AND DATA COLLECTION

The process of acquiring data was performed manually since there were no publicly available datasets specifically focused on garbage materials. Initial attempts were made using the Flickr Material Database and images from Google Images. However, these sources did not accurately represent the actual state of recycled goods, as further research revealed discrepancies with recycling plants and the condition of recycled materials. For instance, the images from the Flickr Material Database depicted materials in pristine and undamaged conditions, which is unlikely for recycled materials treated as waste that are often dirty, ruffled, crumpled, etc.

To address this limitation, we took the initiative to collect our own dataset of images, which we intend to make publicly available. The dataset comprises approximately 2,400 images, distributed among six classes, with each class containing around 400-500 images (except for the "trash" class, which has approximately 100 images). The data acquisition process involved capturing photographs of trash and recycling in various locations, including Stanford, our homes, and our relatives' homes. The images were taken against a white posterboard background, resulting in variations in lighting and pose, thereby introducing diversity into the dataset. Example images from each of the six classes are depicted in the figures below.

To augment the dataset and compensate for the relatively small size of each class, we applied various data augmentation techniques to each image. These techniques included random rotation, random brightness control, random translation, random scaling, and random shearing of the images. These transformations were chosen to account for the different orientations of recycled materials and to maximize the dataset size. Additionally, mean subtraction and normalization were performed to further enhance the dataset's quality and consistency.

Convolutional neural networks

Convolutional neural networks (CNNs) are a crucial component of image recognition and classification tasks. They utilize image-specific algorithms to detect objects, differentiate faces, and perform various other visual recognition tasks. CNNs consist of interconnected neurons with learnable weights and biases. Each neuron receives multiple inputs, computes a weighted sum, applies an activation function, and produces an output.

CNNs are commonly used for tasks such as image classification, object detection, and clustering based on image similarity. They have proven to be effective in recognizing objects, faces, street signs, animals, and more.

The CNN architecture comprises three fundamental components: Convolutional Layer, Pooling Layer, and Fully-Connected Layer.

Convolutional Layer: This layer employs filters and a stride value. Filters are local connectivity networks that learn specific image features. Each filter slides across the input image with a customizable stride size. For example, a 10x10 filter with a stride of 1 overlays a 10x10 pixel window on the image, calculating the inner product between the pixel values and the filter's parameters. The filter then moves horizontally and vertically by the stride value, covering the entire image.

Pooling Layer: Pooling layers are typically of two types: Max-Pooling and Average-Pooling. In Max-Pooling, the maximum value within each pooling window is selected, while Average-Pooling calculates the average value. The purpose of the pooling layer is to increase the filters' robustness and reduce the number of parameters. It also provides regularization to prevent overfitting.

Fully-Connected Layer: The fully connected layer connects the neurons from the last convolutional layer to the output layer. It flattens the output of the previous layer and establishes complete connectivity between the neurons. In some cases, multiple fully connected layers may be present. It is important to note that the number of parameters in fully connected layers is typically significantly higher than in convolutional layers. The fully connected layers often account for around 60% of the total parameters in a standard CNN.

By leveraging these components and stacking multiple layers together, CNNs can learn hierarchical representations of visual features, enabling them to classify and recognize objects within images accurately. The ability of CNNs to automatically learn and extract relevant features from images has made them highly effective in various computer vision applications.

Data Pre-Processing

Load data

In [1]:
```
import tensorflow as tf
import keras
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


from keras.callbacks import ModelCheckpoint
```

In [2]:
```
IMAGE_SIZE = 256
FILE_PATH = r"C:\Users\Admin\Desktop\Waste Segregation using Deep learning TF\dataset-resized"
CHANNELS = 3
BATCH_SIZE =32
```

In [3]:
```
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    FILE_PATH,
    image_size = (IMAGE_SIZE ,IMAGE_SIZE),
    shuffle =True ,
    batch_size = BATCH_SIZE,
    seed = 125
)
```

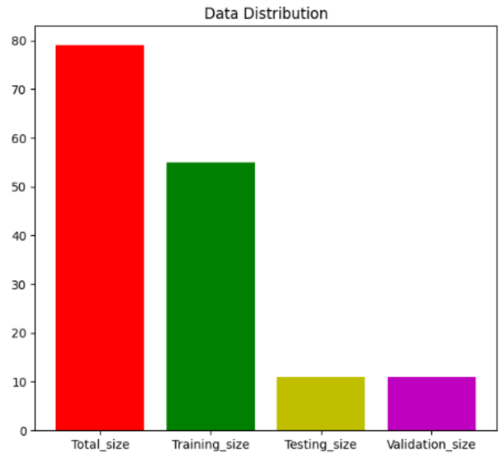
Viewing labels

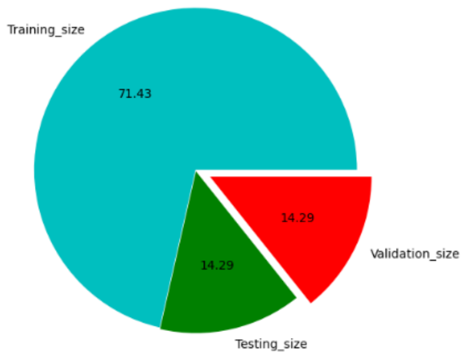
```
In [5]: #viewing labels and class of 1st batch  32 images

        for image_batch , labels in dataset.take(1):
          print(image_batch.shape)
          print(labels.numpy())

(32, 256, 256, 3)
[1 3 2 4 4 4 1 0 1 0 3 4 1 4 2 3 2 0 3 4 4 2 3 0 5 1 2 3 3 3 4 0]
```

```
In [6]:
        plt.figure(figsize=(15, 15))
        for img ,labels in dataset.take(1):
          for i in range(20):
            plt.subplot(4 ,5, i+1)
            plt.imshow(img[i].numpy().astype('int32'))
            plt.title(class_names[labels[i]])

            plt.axis('off')
```

Data Distribution

```
In [8]: train_data , val_data , test_data = split_data(dataset)

{'Total_size': 79, 'Training_size': 55, 'Testing_size': 11, 'Validation_size': 11}
```





Model Building and Training

The VGG (Visual Geometry Group) model is a deep convolutional neural network architecture that was proposed by Karen Simonyan and Andrew Zisserman in their paper "Very Deep Convolutional Networks for Large-Scale Image Recognition" in 2014. The VGG model achieved outstanding performance on the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) dataset, demonstrating the power of deep convolutional networks.

The VGG model architecture is characterized by its simplicity and uniformity. It consists of multiple layers stacked on top of each other, primarily comprising 3x3 convolutional layers and max-pooling layers. The main idea behind VGG is to stack multiple smaller convolutional layers instead of using a few large ones to deepen the network.

In [11]:
```
input_shape = (IMAGE_SIZE , IMAGE_SIZE , CHANNELS )


base_model = tf.keras.applications.VGG16(weights ='imagenet' ,include_top = False , input_shape =input_shape)

base_model.trainable =False
inputs = tf.keras.Input(shape = input_shape)

#layers
x = data_augmentation(inputs)
x = resize_rescale(x)

x= base_model(x , training =False)
x= keras.layers.GlobalAveragePooling2D()(x)
x= keras.layers.Dropout(0.2)(x)
x= tf.keras.layers.Dense(1024, activation='relu')(x)
x= tf.keras.layers.Dense(1024, activation='relu')(x)

outputs = tf.keras.layers.Dense(num_classes, activation = 'softmax')(x)

vgg16_model = tf.keras.Model(inputs ,outputs)


lr = 0.0001
vgg16_model.compile(
    optimizer =tf.keras.optimizers.Adam(learning_rate =0.0001),
    loss = 'sparse_categorical_crossentropy' ,
    metrics =['accuracy']
)
```

Model Selection

Out[116]: Text(0.5, 0, 'EPOCHS')

Conclusion

The advancements in modern libraries like TensorFlow have provided powerful tools for developing complex machine learning algorithms, enabling us to assess the true value of models in addressing environmental risks. However, building and scaling a large dataset over time requires a sophisticated data storage architecture. Additionally, human involvement is crucial for data collection and interpretation, necessitating collaboration among multiple individuals.

Deep learning, being the forefront of computational advancements, has made significant contributions. Tasks like garbage segregation, which can be inconvenient and laborious, can now be automated using Artificial Intelligence. Although we are still far from having sentient robots that fully support humans, the progress made in this direction is undeniable.

We have achieved notable improvements in the accuracy of categorizing recyclables and rubbish using CNN. Through our model, either independently or in combination with other models, we have consistently achieved test accuracies exceeding 70%. These advancements provide promising prospects for effectively addressing waste management challenges and promoting sustainable practices.