A

Report

On

Reduction of Recyclable Waste ending up in a landfill

Submitted in partial fulfillment of the requirements of the course MF F485 Sustainable Manufacturing

By

Tejash Singh - 2019ABPS0775P

Anurag Singh - 2019ABPS1060P

Abhishek Malav - 2019ABPS0916P

B.E. Manufacturing (2019-23)

Under the supervision of

Prof. Kuldip Singh Sangwan

Birla Institute Technology and Science Pilani (Pilani Campus) Semester 1 2021-22

December 2021

Acknowledgement:

We would like to express our gratitude to Professor K.S. Sangwan of the Mechanical Engineering Department for providing us with the opportunity to work on this project, and compile our results in the form of this report, which helped us learn not only about this particular topic but also the entire process of conducting technical research.

We would also like to extend this thanks to Rajni ma'am and Deepika ma'am for guiding us through our journey of working on this project.

Abstract:

A Machine Learning model devised for waste segregation and classification was made using CNN (Convolutional Neural Network) algorithm with the main aim of reducing the load and unattended and unsegregated waste ending up in a landfill which should have been recycled and would have been used again this created a plenty of problems like soil pollution, leachating and air and water pollution which are the indirect results of the former.

Every machine learning model has to be trained with a ton of data, and then the accuracy of the model which is trained needs to be inspected by some other set of data so as to show what amount of accuracy our model leads. So we collected and divided our dataset into 2 groups namely train dataset and test dataset. Our dataset was basically images of waste we wanted to segregate, we limited our domain to household waste and trained and tested our model with respect to such a set of images.

Table Of Contents:

Abbreviations:

CNN: Convolution Neural Networks
ML: Machine Learning
R: Recyclable
O: Organic
Conv2D : 2D Convolution Layer
ReLU : rectified linear activation function

Introduction:

The majority of human activity produces waste, which is unavoidable. Economic development and rising living standards in Asia and the Pacific have resulted in an increase in the quantity and complexity of waste generated, while industrial diversification and the provision of expanded health-care facilities have added substantial quantities of industrial hazardous waste and biomedical waste to the waste stream, potentially posing serious environmental and human health risks.

With the advancement of the modern world has increased the use of a lot more use-and-throw material which mainly includes plastics, according to a survey it was recorded that there was more than 27 million tons of plastic waste that ended up in a landfill, however these plastics and other wastes can be recycled and may further be used. In our society, waste classification has become an important problem, and as such, it is important that landfills and other waste deposition sites are able to properly classify recyclable wastes such as plastics, from non-recyclable wastes such as organic materials, so that as much recyclable materials can be reused. When waste is sorted properly, a higher percentage of recyclable materials are correctly identified, which can lead to many benefits for the environment, as materials such as plastics can take decades to properly be broken down.

Unfortunately, in the present scenario, many recyclable materials are being contaminated with other non-recyclable materials, leading to them being simply discarded rather than broken down properly. The non-recyclable waste contaminating the recyclable waste gives us an example of 2016, when China received millions of tons of recyclable waste from the US but 30% percent of the waste was never recycled, reason being, contamination of it by non-recyclable waste. These situations are non-unique, and many un-recycled plastics contribute to patches of garbage that float around in the oceans, such as the Great Pacific Garbage patch, which is an accumulation zone of plastics that occupies the Pacific Ocean.

Many towns and states have used innovative technology to quantify and assess waste at waste sorting centers to mitigate the effects of recyclable waste being wasted and not recycled. Some

centers have begun to use computer vision and machine learning to distinguish and differentiate recyclable trash from nonrecyclable wastes in the early stages of the process, ensuring that recyclable wastes are not contaminated and are instead correctly sorted and recycled.
In this research, we employ a machine learning classification model to offer a method for classifying trash using computer vision. We trained our model to classify organic materials using a convolutional neural network (O)Classification of Waste materials that can be recycled (R). To avoid waste contamination, we use this method to distinguish between recyclable and non-recyclable wastes.

Background:

Our project is based on **classification of waste using machine learning**.
Waste classification and recycling play a very important role in our daily lives. As people's living standards improve, more and more garbage is being generated. In the face of increased waste disposal and environmental degradation, Accurately classify waste, make the best use of waste resources, The quality of the living environment is an urgent issue that is a common concern in the world. Waste Classification techniques are used to classify and manage waste at its source and to contain waste. Resources by later classification and recycling. In the past, waste classification. It required a lot of human and physical resources.

With artificial development Intelligence, deep learning, and smart technology are widespread. Intellectual Waste classification has become an important technology in waste management. Intellectual Waste classification can be applied to mobile devices, smart recyclable trash cans, and more. It is environmentally friendly and improves the recycling of waste resources.
Describes the overall structure of the project. First, the discarded image is preprocessed. Second, some image features are extracted through a designed network model. Next, the features of the extracted image are normalized. Finally, use the Softmax classifier to classify the garbage images. This section details the designed network model

Methodology:

Waste management is a big issue in the world and most of the wastes end up in landfills. This leads to many issues such as Increase in landfills, eutrophication, consumption of some toxic waste by animals, and various types of pollution and many more things which are damaging our environment. So, basically our approach towards the problem is that we will first do an analysis of the components in household waste then segregation of that will be done based on two factors like if the waste is organic or recyclable. We will automate this process by using a machine learning model. Which will help to reach our end goal to reduce any unwanted waste ending in landfills.

Starting with the study of waste, its types, sources of origin and its classification.
Realizing the huge domain of waste comes is either organic or recyclable.
Collected data to train the machine learning model, divided it into 2
Sets. Now for the coding part, We devised a CNN (Convolutional Neural Network) Algorithm and trained it with the training data, and validated with the test data. Rectified the code several times to reach the desired accuracy.
A thorough study of wastes was done and knowledge was gained regarding different types of wastes and what are their consequences to the environment due to mismanagement of the waste.

Environmental pollution as a result of improper solid waste management is a worldwide problem. The major waste treatment and ultimate disposal procedures, which are mostly apparent in low-income nations, are open dumping and open burning. The primary consequences of garbage mishandling in developing nations are examined in this research, with an emphasis on environmental pollution and social difficulties. The informal sector's activities in developing cities were also examined, with an emphasis on the major health concerns associated with rubbish scavenging. The findings revealed that environmental repercussions are widespread over the world: marine litter, air, soil, and water pollution, and garbage pickers' direct contact with hazardous material are the most pressing concerns. Many assessments of various waste streams

have been published in the scientific literature in an attempt to estimate their environmental impact.

We also studied the various methods of classification of waste. Like organic waste, recyclable waste and hazardous waste.

Organic waste is a typical type of home garbage. Organic waste includes all food waste, garden trash, manure, and rotting meat. Microorganisms convert organic waste into manure over time. This does not, however, imply that you may dispose of them anywhere you like.

Organic trash in landfills produces methane, thus it should never be thrown away with regular garbage. Rather, contact your local government to obtain a green bin, or hire a green skin bin or garden bag for proper garbage disposal.

We're probably all acquainted with the term "**recyclable trash,**" which refers to any garbage that may be turned into products that can be used again. Paper, metals, furniture, and biological trash are all solid materials that may be recycled.

Hazardous waste is defined as trash that is combustible, poisonous, corrosive, or reactive in nature.

These objects have the potential to be severely damaging to both you and the environment, and must be properly disposed of. As a result, I propose that you hire a garbage removal firm to dispose of any hazardous material properly.

After gathering all the information we tried searching for a data set and wrote the basic code using CNN. Finalized the library and wrote the basic code. We also tried improving accuracy using hyper parameter tuning and made necessary changes wherever necessary.

Discussion:

Our dataset consists of 22564 training images and 2513 testing images. All the images in the dataset can be classified into ‘Organic’ or ‘Recyclable’. The specific dataset used by us can be found at:

Waste Classification Data

4.1 **Initial Stages**

Having fulfilled our requirement of a bid dataset, we need special libraries to process this data. To handle this type of data we make use of numpy and pandas, these are the libraries which are capable of making fast calculations. We import the libraries required for our model like BatchNormalization, MaxPooling2D, Sequential, etc. from Keras in Tensorflow.
The dataset was then loaded into the file. We used the code given in the image below.

```
train_dir = "D:\Sus manu\DATASET\TRAIN"
test_dir = "D:\Sus manu\DATASET\TEST"

def load_dataset(path):
    data = load_files(path) #load all files from the path
    files = np.array(data['filenames']) #get the file
    targets = np.array(data['target'])#get the the classification labels as integer index
    target_labels = np.array(data['target_names'])#get the the classification labels
    return files,targets,target_labels
```

```
x_train, y_train,target_labels = load_dataset(train_dir)
x_test, y_test,_ = load_dataset(test_dir)
```

The target_labels is the variable that has the classification labels i.e. ‘O’ for organic and ‘R’ recyclable. The x_train, y_train, x_test, y_test are respective x and y values for train and test. Now, we will split the train data into train and validate. This validation data is used for testing the model after it has been trained on training data. The code we have used for this is:-

```
x_train,x_validate,y_train,y_validate = train_test_split(x_train,y_train,test_size = 0.2,random_state = 1)
```

x_validate and y_validate are analogous to x_train and y_test values. The difference is that they are used for different purposes. Testing data is used for determining final scoring metrics. Whereas validation data is used to see how our data is performing on unknown data and to see if it is underfitting or overfitting.

4.2 **Transforming the data**

Since our computer only understands the numbers and it does not understand images naturally, we have to convert the images in our dataset into arrays of numbers. For this purpose we will use the OpenCV library. The code is given below.

```
# Convert jpg file to numpy array to feed to the CNN.
#By using Opencv .

def convert_image_to_array(files):
    width, height, channels = 100, 100, 3
    images_as_array = np.empty((files.shape[0], width, height, channels), dtype=np.uint8) #define train and test data shape
    for idx,file in enumerate(files):
        img = cv2.imread(file)
        res = cv2.resize(img, dsize=(width, height), interpolation=cv2.INTER_CUBIC) #As images have different size, resizing all
        images_as_array[idx] = res                                                  #images to have same shape of image array
    return images_as_array
```

```
x_train = np.array(convert_image_to_array(x_train))
print(x_train.shape)

x_valid = np.array(convert_image_to_array(x_validate))
print(x_valid.shape)

x_test = np.array(convert_image_to_array(x_test))
print(x_test.shape)
```

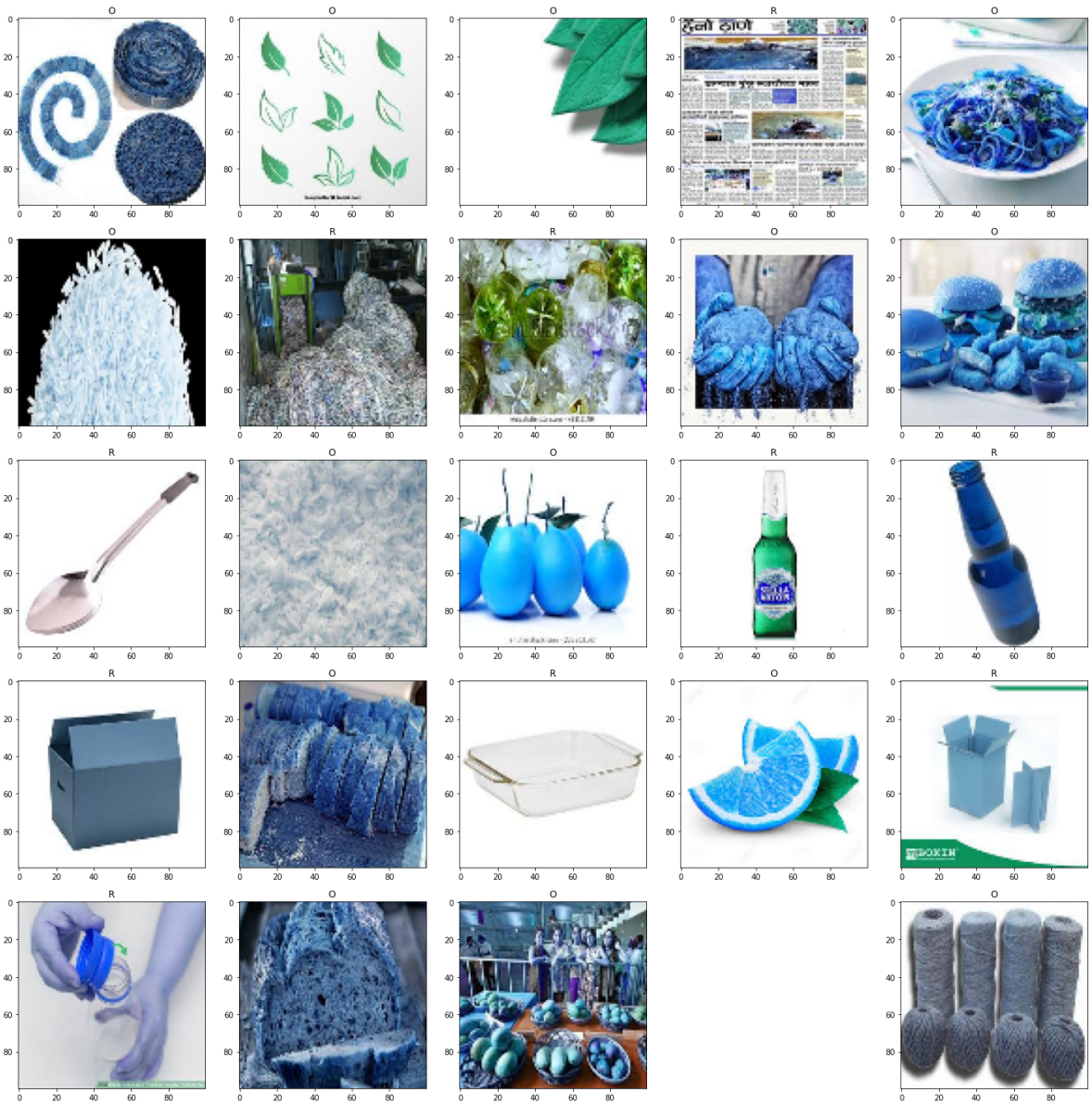
After writing the function for converting the image to an array, we use the function on images in train. Validate and test dataset.

Now each image is transformed into an array of numbers. The number of elements in the array depends on the number of pixels. Each pixel is again an array of three elements depicting the three values of RGB. Each value ranges from 0 - 255. And then we divide each number in the array by 255 because numbers from 0-1 can be easily processed by our computer.

```
x_train = x_train.astype('float32')/255
x_valid = x_valid.astype('float32')/255
x_test = x_test.astype('float32')/255
```

Now that we have loaded and transformed our data, we will want to see if our images are being inputted correctly. Hence we write the code for randomly printing 25 images.

```
plt.figure(figsize=(20,20))
classes = ['O','R']
for i in range(1,26):
    index = np.random.randint(x_train.shape[0])
    plt.subplot(5, 5, i)
    plt.imshow(np.squeeze(x_train[index]), cmap='cool')
    plt.title(classes[int(y_train[index])])
    plt.tight_layout()
plt.show()
```



4.3 **Our CNN model's layers**

We have such a big dataset that we cannot process whole data in one iteration. So we have to supply a small number of images in each iteration. We will process 10 such iterations. An average number of time taken per iteration is around 890 - 1000 seconds. To supply random images in each iteration we use the ImageDatagenerator function. For our model we have set the batch size to be 256.

```
datagen = ImageDataGenerator(
        featurewise_center=False,  # set input mean to 0 over the dataset
        samplewise_center=False,  # set each sample mean to 0
        featurewise_std_normalization=False,  # divide inputs by std of the dataset
        samplewise_std_normalization=False,  # divide each input by its std
        zca_whitening=False,  # apply ZCA whitening
        rotation_range=0,  # randomly rotate images in the range (degrees, 0 to 180)
        zoom_range = 0.1, # Randomly zoom image
        width_shift_range=0.2,  # randomly shift images horizontally (fraction of total width)
        height_shift_range=0.2,  # randomly shift images vertically (fraction of total height)
        horizontal_flip=False,  # randomly flip images
        vertical_flip=False)  # randomly flip images
datagen.fit(x_train)
```

There are a number of arguments that this function takes like height_shift_range for a vertical shift of image and width_shift_range for a horizontal shift of image.

It is time for us to build our CNN layer model. These are the layers that our data will pass though. The sequential command is the easiest way to build the model. Then we use the add() function to add layers to our existing model. Like this we have built a model with 24 layers. Our model includes the following layers: Conv2d, ReLU, MaxPool2d, and activation. For our neural network, we employed the ReLU linear activation function. Our MaxPool2d layers selected the highest value in our matrix. It decreases the size of the Conv2d layer's outputs.

```
# Convolutional Neural Network - CNN

model = Sequential()
model.add(Conv2D(32,(3,3),input_shape = (224,224,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D())

model.add(Conv2D(64,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D())

model.add(Conv2D(128,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D())

model.add(Conv2D(128,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D())

model.add(Conv2D(128,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D())

model.add(Flatten())
model.add(Dense(256))
model.add(Activation("relu"))
model.add(Dropout(0.5))
model.add(Dense(164))
model.add(Activation("relu"))
model.add(Dropout(0.2))
model.add(Dense(numberOfClass)) # output
model.add(Activation("sigmoid"))

model.compile(loss = "binary_crossentropy",
              optimizer = "adam",
              metrics = ["accuracy"])
batch_size = 256
```

After building our layers, we write the code to allow our model to read images from folders.

```
train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size= (224,224),
        batch_size = batch_size,
        color_mode= "rgb",
        class_mode= "categorical")

test_generator = test_datagen.flow_from_directory(
        test_dir,
        target_size= (224,224),
        batch_size = batch_size,
        color_mode= "rgb",
        class_mode= "categorical")
```

4.4 **Fitting the data**

Now this is where we will start fitting our data in the model. We take our data and fit it into our model. This step takes around 3 hrs to execute properly. This step also runs at 100% memory. If your computer is not compatible enough for such kinds of executions, it is recommended that you use google collab.

```
hist = model.fit_generator(
        generator = train_generator,
        epochs=10,
        validation_data = test_generator)
```

In our case we have taken 10 epochs. One epoch is when our dataset is being passed through our model once.

```
Epoch 1/10
89/89 [==============================] - 1045s 12s/step - loss: 0.4942 - accuracy: 0.7839 - val_loss: 0.3258 - val_accuracy: 0.
8822
Epoch 2/10
89/89 [==============================] - 1051s 12s/step - loss: 0.3898 - accuracy: 0.8386 - val_loss: 0.3257 - val_accuracy: 0.
8504
Epoch 3/10
89/89 [==============================] - 1134s 13s/step - loss: 0.3572 - accuracy: 0.8554 - val_loss: 0.2575 - val_accuracy: 0.
8981
Epoch 4/10
89/89 [==============================] - 916s 10s/step - loss: 0.3280 - accuracy: 0.8678 - val_loss: 0.3132 - val_accuracy: 0.8
739
Epoch 5/10
89/89 [==============================] - 832s 9s/step - loss: 0.2959 - accuracy: 0.8803 - val_loss: 0.2626 - val_accuracy: 0.89
30
Epoch 6/10
89/89 [==============================] - 15412s 173s/step - loss: 0.2682 - accuracy: 0.8942 - val_loss: 0.3047 - val_accuracy:
0.8854
Epoch 7/10
89/89 [==============================] - 854s 10s/step - loss: 0.2322 - accuracy: 0.9100 - val_loss: 0.2659 - val_accuracy: 0.9
061
Epoch 8/10
89/89 [==============================] - 1394s 16s/step - loss: 0.2010 - accuracy: 0.9230 - val_loss: 0.3262 - val_accuracy: 0.
8731
Epoch 9/10
89/89 [==============================] - 1349s 15s/step - loss: 0.1563 - accuracy: 0.9433 - val_loss: 0.3572 - val_accuracy: 0.
8818
Epoch 10/10
89/89 [==============================] - 1277s 14s/step - loss: 0.1285 - accuracy: 0.9549 - val_loss: 0.3938 - val_accuracy: 0.
8731
```

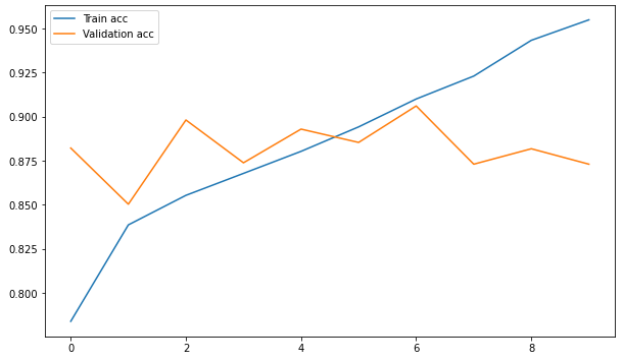
Results / conclusion

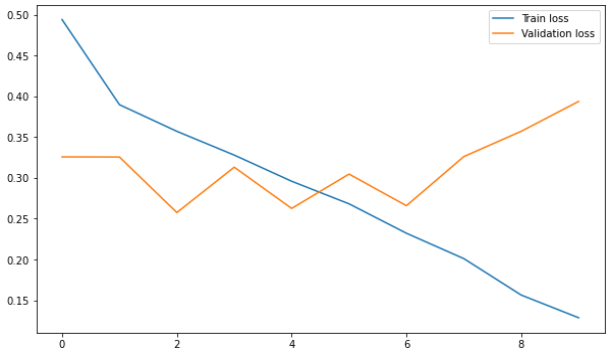
We consider the accuracy of our last epoch to be the accuracy of our model. This comes out to be 87.31%.
We have plotted the train and validation accuracy against our epochs.

```
plt.figure(figsize=[10,6])
plt.plot(hist.history["accuracy"], label = "Train acc")
plt.plot(hist.history["val_accuracy"], label = "Validation acc")
plt.legend()
plt.show()
```

```
plt.figure(figsize=(10,6))
plt.plot(hist.history['loss'], label = "Train loss")
plt.plot(hist.history['val_loss'], label = "Validation loss")
plt.legend()
plt.show()
```

Here we can see that our validation accuracy is highest for the 6th epoch.
Similarly we have plotted the training and validation loss against each epoch. Loss is a scalar variable that we try to reduce during model training. The lesser the loss, the more accurate our projections are.





Now based on our model, we will try to predict some categories for some of our images.

```
test_x, test_y = test_generator.__getitem__(1)

labels = (test_generator.class_indices)
labels = dict((v,k) for k,v in labels.items())

preds = model.predict(test_x)

plt.figure(figsize=(16, 16))
for i in range(15):
    plt.subplot(4, 4, i+1)
    plt.title('pred:%s / truth:%s' % (labels[np.argmax(preds[i])], labels[np.argmax(test_y[i])]))
    plt.imshow(test_x[i])
```

References:

- Neha Naveen, Michael Rodrigues and Srjana Srivatsa, *WASTE CLASSIFICATION.*
- Dataset:- Waste Classification data | Kaggle
- Stackoverflow for our doubts and debugging.