Waste Segregation Using Artificial Intelligence

Sindhu Rajendran, Vidhya Shree, Rajat Keshri, Rohit S, Rachana S

Abstract: The current waste segregation methods include manual segregation of waste by the waste generators. This method though simple is not effective when it comes to accuracy and time interval of waste segregation. We present an effective system for the above purpose which segregates waste based on supervised machine learning algorithms and segregates the waste into cardboard, glass, metal, trash, paper and plastic. Initially, data is collected and augmentation is done. The algorithm converts the image sets available in different folders into gray scale converting it to a 2D matrix. The images are then converted and stored in a 1D array which are further used for labeling while testing. By using CNN the input images are sampled and then convolved to determine the edges in the images. Pooling is done iteratively to reduce the dimensions of the image. Further, on application of activation function the output image is obtained. The proposed system presents a simple and an effective method to segregate waste using machine learning, thus completely removing human intervention in the segregation stage. An efficiency of 80% has been achieved in the testing process.

1 INTRODUCTION

Waste management is a continuous process of the collection, transport, treatment and disposal of waste[1]. Monitoring and regulation are a very important part of the waste management process. With the increasing amount in the waste generation it is imperative that we make the process of waste disposal and treatment as efficient as possible[2]. When the domestic household waste is considered, the disposal method includes collection of waste by the government authorities at the door step of the people. But most of the time the waste is not segregated properly, which effects the further segregation stages in a negative manner. Our goal is to find an efficient and a cost-effective method to segregate waste at the lowest possible level. According to the Solid Waste Management Rules 2016, the generators of waste are supposed to segregate the waste into three parts which are biodegradable, non-biodegradable and domestic hazardous wastes and hand them over to the local authorities on time to time basis. Untreated waste become a breeding point for many harmful microorganisms. The gases released at the site of untreated waste is harmful and causes diseases like nausea and many other airborne diseases. Untreated waste mixes with water and causes many waterborne diseases like cholera and diarrhoea. We propose to solve the waste segregation problem using machine learning. This model consists of training a neural network based to classify the images into biodegradable and non-biodegradable wastes. Logistic regression and basic binary classification is applied[3]. The images are classified into glass, metal, paper, plastic, cardboard and trash. The neural network in consideration involves four layers, convolution, pooling, dropout, flatten and dense. This is followed by the activation function to give the output image. Using supervised learning, initially the data is collected and augmentation is done. The images are then stored in a 1D array and further labelled for testing the model. The advantage of this type of segregation is the ease of segregation and also the accuracy of the segregation of waste.

- *Sindhu Rajendran is currently working as the Assistant Professor in Electronics and Communication engineering in R.V College of Engineering, Karnataka, India. E-mail: sindhur@rvce.edu.in*
- *Vidhya Shree is currently pursuing Bachelor's degree program in electronics and Instrumentation engineering in R.V College of Engineering, Karnataka, India. E-mail: vidhya1697@gmail.com*
- *Rajat Keshri, Rohit S, Rachana S are pursuing Bachelor's degree program in Electronics and Communication engineering in R.V College of Engineering, Karnataka, India.*

2 METHODOLOGY

Deep learning neural networks are of prime importance when it comes to image recognition, classification and detection [5]. Different layers of the neural network work together in order to achieve the given output. Each layer can either reduce the dimensions or work on sampling the image. Convolution neural network (CNN) is a type of neural network where a multilayer perceptron is used. It is helpful in identifying a general 2D image and collect the information [6]. The network is made up of:

- input layer
- convolution layer
- sample layer
- output layer

To improve the accuracy and structure, the convolution layer and sample layer can be multi-layered. In CNN, instead of checking the whole image, we check the local area containing valuable information about the image which sometimes is called Region CNN or RCNN. Consider taking a small patch of the input image and run a small neural network on it with 'k' outputs and represent them vertically. With the help of a suitable kernel, we can reduce the dimensions of the image to be ready for pooling.
Network is moved across the whole image to get another image with different width, height, and depth. In the later stages, we use an activation function on the convolved image to get the output (RELU or SoftMax function) [7]. Matrix multiplication is used for convolution.

$$\begin{array}{|c|c|c|}\hline C_{11} & C_{12} & C_{13} \\ \hline C_{21} & C_{21} & C_{22} \\ \hline C_{31} & C_{32} & C_{33} \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|}\hline A_{11} & A_{12} & A_{13} & A_{14} & A_{15} \\ \hline A_{21} & A_{22} & A_{23} & A_{24} & A_{25} \\ \hline A_{31} & A_{32} & A_{33} & A_{34} & A_{35} \\ \hline A_{41} & A_{42} & A_{43} & A_{44} & A_{45} \\ \hline A_{51} & A_{52} & A_{53} & A_{54} & A_{55} \\ \hline \end{array} \text{ X } \begin{array}{|c|c|c|}\hline K_{11} & K_{12} & K_{13} \\ \hline K_{21} & K_{22} & K_{23} \\ \hline K_{31} & K_{32} & K_{33} \\ \hline \end{array}$$

Fig 1. Depicts matrix operation of **C=A*K**

Thus, the output matrix has lesser weights used to express most of the important information of original matrix. Next step after sampling and convolving with kernel is Pooling

[8]. In pooling, we consider a pooling layer that is continuously inserted in the image matrix and reduces the size to reduce the computation speed and prevent overfitting.

Pooling can either be 'Average' pooling where the average values of the image region is taken or 'Max' pooling where maximum value is considered.

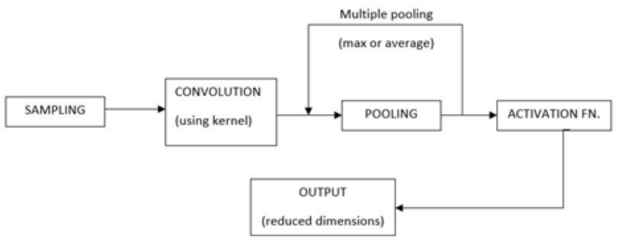
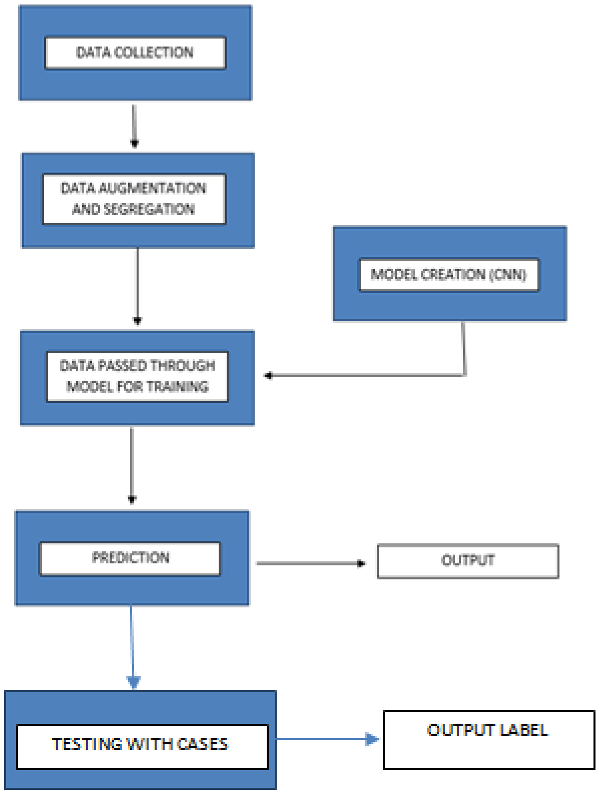


Fig 2. Execution of CNN algorithm

3 Implementation and Specifications:

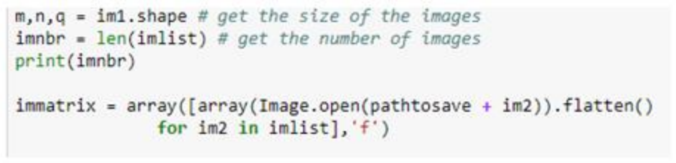


The segregation problem is solved using supervised learning approach. Firstly, images of cardboard, glass, paper, plastic, metal and random trash (soda cans) was collected. Each of these categories has around 500 images. Each of the images were reshaped to 64x64 dimensions, and converted to grayscale. This is essential for reshaping that would decrease the time and computational complexity of the neural network. Converting to gray scale allows the network to work with black and white images and RGB is eliminated. This is implemented considering that the colour of the trash is not an important factor. Once we have the data ready, we create output labels for each dataset class. This is done to predict the output category.

3.1 Training

For training the data, first, all the images of a particular category are converted to an array. So cardboard images will be stored in a numpy array. We use numpy library here as it is easy to use it with matrices and arrays.

An array of arrays of all images will be created with all the categories in that single array of array with its corresponding labels. This array of array is fed into the neural network and training is done.



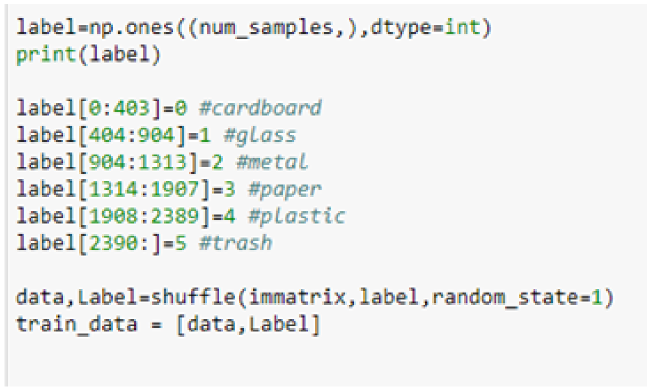
```
m,n,q = im1.shape # get the size of the images
imnbr = len(imlist) # get the number of images
print(imnbr)

immatrix = array([array(Image.open(pathtosave + im2)).flatten()
              for im2 in imlist],'f')
```

Snippet 1

In snippet 1, each image is converted into an array and stored in numpy format.

In snippet 2, immatrix is the array of arrays of images and flatten is used to convert it into a 1 dimension vector.



```
label=np.ones((num_samples,),dtype=int)
print(label)

label[0:403]=0 #cardboard
label[404:904]=1 #glass
label[904:1313]=2 #metal
label[1314:1907]=3 #paper
label[1908:2389]=4 #plastic
label[2390:]=5 #trash

data,Label=shuffle(immatrix,label,random_state=1)
train_data = [data,Label]
```

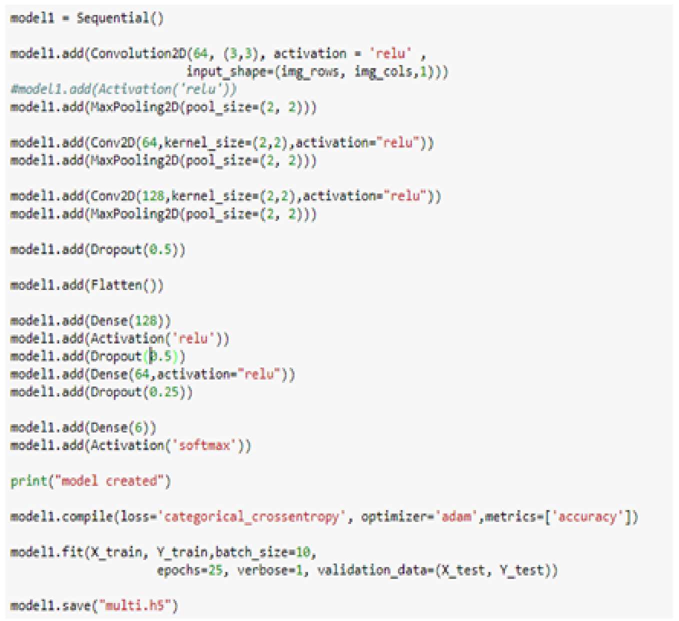
Snippet 2

Output labels are created and then added to the tuple 'train_data'.

Further, the splitting of the train and test data is done and the model is trained on the training set and validated on the test set.

3.2 Validation

Validation of the model is very important as it tells us how good our model is performing on the random test data set. We aim to make our validation accuracy high. In snippet 3, we have used a sequential model, with one input convolution layer, with 64x64 pixels as the input and a kernel size of 3x3. Kernel size basically gives the size of the matrix which moves over the image matrix and compares all the pixels in the grid of 3x3 pixels, and reduces it to one pixel. A maxpooling layer is added to pad with zeroes and not reduce the size of each image below 64x64, as due to convolution, the image size reduces. Here, 3 convolution layers are added, with 64, 64 and 128 as the convolution size, respectively. A dropout layer is also added with a dropout of 0.5. This layer is used to increase the computation even further, by not using some neurons during the training. One hidden layer, with 128 neurons, is added to the model, with activation function as rectified linear function (relu). The output layer has 6 neurons as there are 6 classes and softmax function is used. Softmax function is used for multiclass classification.



```
model1 = Sequential()

model1.add(Convolution2D(64, (3,3), activation = 'relu' ,
                         input_shape=(img_rows, img_cols,1)))
#model1.add(Activation('relu'))
model1.add(MaxPooling2D(pool_size=(2, 2)))

model1.add(Conv2D(64,kernel_size=(2,2),activation="relu"))
model1.add(MaxPooling2D(pool_size=(2, 2)))

model1.add(Conv2D(128,kernel_size=(2,2),activation="relu"))
model1.add(MaxPooling2D(pool_size=(2, 2)))

model1.add(Dropout(0.5))

model1.add(Flatten())

model1.add(Dense(128))
model1.add(Activation('relu'))
model1.add(Dropout(0.5))
model1.add(Dense(64,activation="relu"))
model1.add(Dropout(0.25))

model1.add(Dense(6))
model1.add(Activation('softmax'))

print("model created")

model1.compile(loss='categorical_crossentropy', optimizer='adam',metrics=['accuracy'])

model1.fit(X_train, Y_train,batch_size=10,
                    epochs=25, verbose=1, validation_data=(X_test, Y_test))

model1.save("multi.h5")
```

Snippet3

With the given small dataset, and the basic convolution model, we got an accuracy of approx 75% on test case and 65% on validation case, for 25 epochs. If run for a higher number of epochs, the accuracy would increase even more. Another way to increase the accuracy is to use the already defined models like the VGG16 or the ImageNet.

3.3 Sample output

Notice that two out of the 10 samples, the model predicts it wrong, but the rest of the time, the model predicts it right.

4 CONCLUSION:

Solid Waste Management Rules, 2000 which only required the generators of waste to hand over the waste to the local authorities on time to time basis. Waste management is necessary for the protection of our environment and the health and hygiene of the human population. The proposed system presents a simple and an effective method to segregate waste using machine learning, thus completely removing human intervention in the segregation stage. An efficiency of 80% has been achieved in the testing process.

REFERENCES:

[1] B.Dev, "Automatic Waste Segregation using Image Processing and Machine Learning", May 2018.

[2] Adhithya .P.M, S.V Kaushal, P. Mahalakshmi, " Survey on Identification and Classification of Waste for efficient disposal and recycling", International Journal of Engineering and Technology, 2018.

[3] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on, pages 253–256. IEEE, 2010.

[4] “Content-based Image Retrieval using Deep Convolution Neural Network”: D.D.Pukale ; S.G. Bhirud ; V.D. Katkar,2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA),2017.

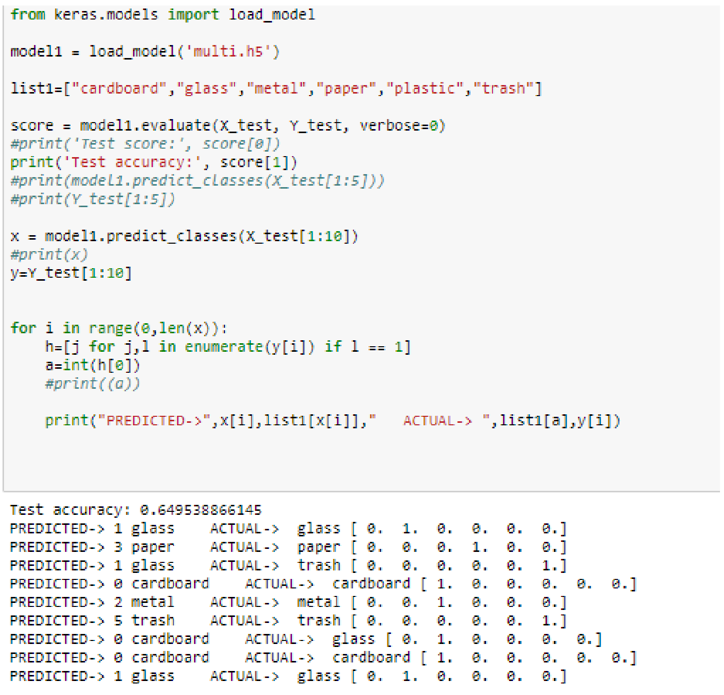
[5] “An efficient license plate recognition system using convolution neural networks”:Cheng-Hung Lin ; Yong-Sin Lin ; Wei-Chen Liu, 2018 IEEE International Conference on Applied System Invention (ICASI),2018.

[6] “Image convolution optimization using sparse matrix vector multiplication technique”:B Bipin ; Jyothisha J Nair,International Conference on Advances in Computing, Communications and Informatics (ICACCI),2016.

[7] “Fast algorithm using summed area tables with unified layer performing convolution and average pooling”: Akihiko Kasagi ; Tsuguchika Tabaru ; Hirotaka Tamura, IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP),2017.

[8] “Deep Learning using Linear Support Vector Machines”, Yichuan Tang,

[9] Department of Computer Science, University of Toronto. Toronto, Ontario, Canada.



```
from keras.models import load_model

model1 = load_model('multi.h5')

list1=["cardboard","glass","metal","paper","plastic","trash"]

score = model1.evaluate(X_test, Y_test, verbose=0)
#print('Test score:', score[0])
print('Test accuracy:', score[1])
#print(model1.predict_classes(X_test[1:5]))
#print(Y_test[1:5])

x = model1.predict_classes(X_test[1:10])
#print(x)
y=Y_test[1:10]


for i in range(0,len(x)):
    h=[j for j,l in enumerate(y[i]) if l == 1]
    a=int(h[0])
    #print((a))

    print("PREDICTED->",x[i],list1[x[i]],"   ACTUAL-> ",list1[a],y[i])
```

```
Test accuracy: 0.649538866145
PREDICTED-> 1 glass    ACTUAL->  glass [ 0.  1.  0.  0.  0.  0.]
PREDICTED-> 3 paper    ACTUAL->  paper [ 0.  0.  0.  1.  0.  0.]
PREDICTED-> 1 glass    ACTUAL->  trash [ 0.  0.  0.  0.  0.  1.]
PREDICTED-> 0 cardboard    ACTUAL->  cardboard [ 1.  0.  0.  0.  0.  0.]
PREDICTED-> 2 metal    ACTUAL->  metal [ 0.  0.  1.  0.  0.  0.]
PREDICTED-> 5 trash    ACTUAL->  trash [ 0.  0.  0.  0.  0.  1.]
PREDICTED-> 0 cardboard    ACTUAL->  glass [ 0.  1.  0.  0.  0.  0.]
PREDICTED-> 0 cardboard    ACTUAL->  cardboard [ 1.  0.  0.  0.  0.  0.]
PREDICTED-> 1 glass    ACTUAL->  glass [ 0.  1.  0.  0.  0.  0.]
```