ABSTRACT

This report briefly describes about the proposed project on waste classification system titled "Classification of Garbage into Different Waste Classes". Garbage Management is a challenge for the city authorities in developing countries mainly due to increasing generation of waste. Garbage wastes are often discarded haphazardly and without proper distinction. Waste management and recycling is the fundamental part of a sustainable economy. For more efficient and safe recycling, it is necessary to organize the waste into proper classes for their suitable recycling and management. This can be achieved through an intelligent system that rightfully distinguishes garbage into its suitable waste class. This project attempts to build that intelligent system to classify garbage into recycling categories in an efficient and reliable way. This system feeds on images and recognize that image into any of the five classes as per the dataset used to train the model. This system works through neural networks and different algorithms for its implementation and also enforce artificial intelligence. This system architecture is based off of VGG-16 model approach which helped to achieve a better result in recognizing the image. With the dataset of 8018 images the system provides an accuracy of 86.61%.

Keywords: Waste classification, recycling, convolution, neural network, VGG-16

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	Full Form
NN	Neural Network
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
ReLu	Rectified Linear Unit
VGG	Visual Geometry Group
ILSVRC	ImageNet Large-Scale Visual Recognition Challenge
LRN	Local Response Normalization
FC	Fully Connected
YOLO	You Only Look Once

CHAPTER 1

INTRODUCTION

1.1 Background

Improper disposal of garbage wastes has become one of the major problems due to the growth in population and rapid industrialization as well as urbanization. Knowing that a good part of the generated garbage in large cities are recyclables, it is need to know and apply reuse methods that could bring benefits or, at least reduce environmental problems. The existence of techniques or models that help people to sort garbage has become essential in the correct dispose of those materials. Although there are different types of recycling categories, people still can be confused or do not properly recognize about how to determine the correct trash bin can to dispose of each garbage.

In order to minimize the impact caused by the incorrect dispose of garbage, more specifically domestic (i.e., paper, plastic, glass and trash), we proposed to use an automated system based on neural network techniques aiming at the correct separation of waste in recycling categories. Ways which humans have managed solid waste over the centuries still based on the original strategy of just eliminate them. Population growth has been the main factor for the increasing production of that garbage. Therefore, it should be reduced on a personal basis to maintain the balance at which the waste is managed [1].

Waste management and efficient sorting of them have been considered as an important role for ecologically sustainable development worldwide. It is essential for the society to reduce waste accumulation by recycling and re-using disposed of products. For more sustainable future, it is vital to maintaining an effective recycling system. Despite the seriousness of this field, currently recycling process is hugely depending on human abilities. In order to automatize recycling process, it is vital to propose intelligent systems to detect the waste components correctly.

For the development of the proposed system, we first need to understand the background of Artificial Neural Network (ANN) and Convolutional Neural Network (CNN). We should have a proper understanding on the concept of ANN and CNN, their methodology, functionalities and working mechanism. We also need the concept of VGG architecture, which we have decided to use as an architecture base for our CNN model.

Artificial Neural Networks Overview

An artificial neural network is a type of machine learning which models itself after the human brain, creating an artificial neural network that via an algorithm allows the computer to learn by incorporating new data [4]. Neural networks consist of individual units called neurons. Neurons are located in a series of group layers. Neurons in each layer are connected to neurons of the next layer. Data comes from the input layer to the output layer along these compounds. Each individual node performs a simple mathematical calculation. Then it transmits its data to all the nodes it is connected to. The last wave of neural networks came in connection with the increase in computing power and the accumulation of experience. That brought Deep learning, where technological structures of neural networks have become more complex and able to solve a wide range of tasks that could not be effectively solved before. Image classification is a prominent example.

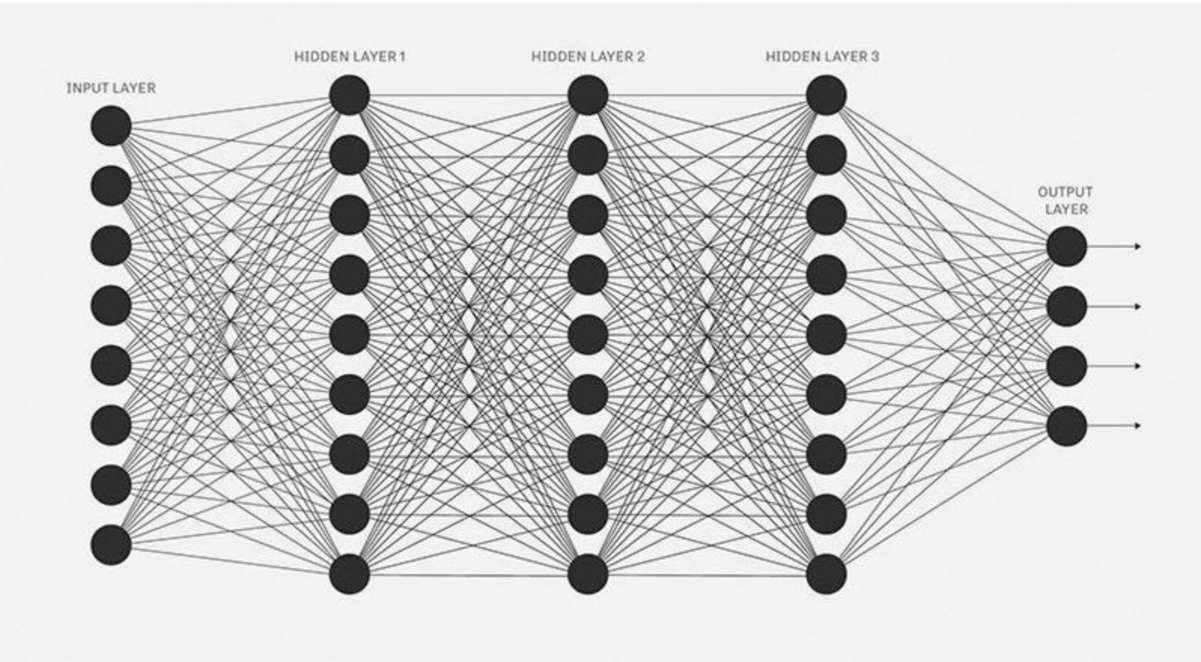


Fig 1.1: Artificial neural network

[Source: https://trendytechz.com/deep-neural-network-plays-major-role-finding-obstacles-near/]

Image Classification using CNN

A Convolutional Neural Network (CNN) is a multilayered neural network with a special architecture to detect complex features in data [6]. CNNs have been used in image recognition, powering vision in robots, and for self-driving vehicles.

An image classifier CNN can be used in myriad ways, to classify cats and dogs, for example, or to detect if pictures of the brain contain a tumor likewise garbage classification etc. Our project is to classify image of daily life garbage into degradable and non-degradable classes.

Once a CNN is built, it can be used to classify the contents of different images. All we have to do is feed those images into the model. Just like ANNs, CNNs are inspired by the workings of the human brain. CNNs are able to classify images by detecting features, similar to how the human brain detects features to identify objects[5].

Before we dive in and build the model, let's understand some concepts of CNNs and the steps of building one.

Images are made up of pixels. Each pixel is represented by a number between 0 and 255. Therefore each image has a digital representation which is how computers are able to work with images.

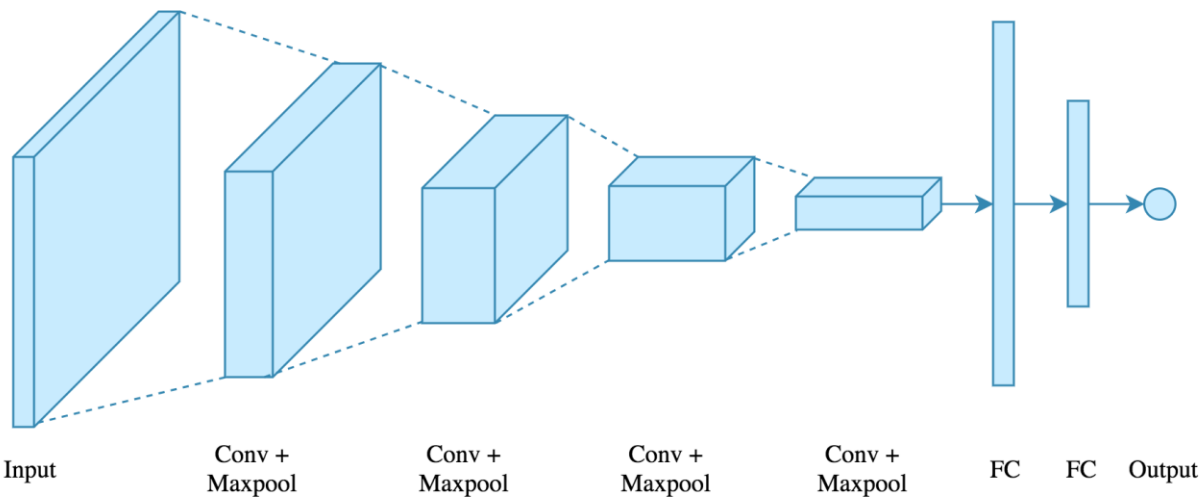


Fig 1.2. CNN Model

[Source:https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2?gi=c12a5a4d6f83]

Convolution: A convolution is a combined integration of two functions that shows you how one function modifies the other.

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau$$

$$= \int_{-\infty}^{\infty} f(t-\tau)g(\tau)\,d\tau \ldots\ldots\ldots\ldots\ldots\ldots \text{(1.1)}$$

The equation 1.1 f and g are both function which operate on the same domain of the inputs. To calculate a convolution between f and g at a point t, we take the integral over all values tau between negative and positive infinity, and at each point, multiply the value of f(x) at position tau by the value of g(x) at t-tau, i.e., the difference between the point for which the convolution is being calculated and the tau at a given point in the integral.

There are three important items to mention in this process: the input image, the feature detector, and the feature map. The input image is the image being detected. The feature detector is a matrix, usually 3x3 (it could also be 7x7). A feature detector is also referred to as a kernel or a filter.

Intuitively, the matrix representation of the input image is multiplied element-wise with the feature detector to produce a feature map, also known as a convolved feature or an activation map. The aim of this step is to reduce the size of the image and make processing faster and easier. Some of the features of the image are lost in this step.

However, the main features of the image that are important in image detection are retained. These features are the ones that are unique to identifying that specific object. For example each animal has unique features that enable us to identify it. The way we prevent loss of image information is by having many feature maps. Each feature map detects the location of certain features in the image.

Apply the ReLu (Rectified Linear Unit): In this step we apply the rectifier function to increase non-linearity in the CNN. Images are made of different objects that are not linear to each other. Without applying this function the image classification will be treated as a linear problem while it is actually a non-linear one.

Pooling: Spatial invariance is a concept where the location of an object in an image doesn't affect the ability of the neural network to detect its specific features. Pooling enables the CNN to detect features in various images irrespective of the difference in lighting in the pictures and different angles of the images.

There are different types of pooling, for example, max pooling and min pooling. Max pooling works by placing a matrix of 2x2 on the feature map and picking the largest value in that box. The 2x2 matrix is moved from left to right through the entire feature map picking the largest value in each pass.

These values then form a new matrix called a pooled feature map. Max pooling works to preserve the main features while also reducing the size of the image. This helps reduce overfitting, which would occur if the CNN is given too much information, especially if that information is not relevant in classifying the image.

Flattening: Once the pooled featured map is obtained, the next step is to flatten it. Flattening involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing.

Full connection: After flattening, the flattened feature map is passed through a neural network. This step is made up of the input layer, the fully connected layer, and the output layer. The fully connected layer is similar to the hidden layer in ANNs but in this case it's fully connected. The output layer is where we get the predicted classes. The information is passed through the network and the error of prediction is calculated. The error is then back propagated through the system to improve the prediction.

The final figures produced by the neural network don't usually add up to one. However, it is important that these figures are brought down to numbers between zero and one, which represent the probability of each class. This is the role of the Softmax function.

$$\sigma: \mathbb{R}^K \longrightarrow (0,1)^K$$

$$\sigma(\boldsymbol{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} for\ j = 1 \ldots\ldots\ldots\ldots\ldots \text{(1.2)}$$

The equation 1.2 represents generalization of logistic function. It squashes a K-dimension $\sigma(z)$ bitrary reak value to a K-dimension vector of real values. Its range is from 0 to 1.

For loss function the Categorical crossentropy is used. Mathematically,

$$L(y, \hat{y}) = -\sum_{j=0}^{M} \sum_{i=0}^{N} (y_{ij} * \log(\widehat{y_{ij}})) \ldots\ldots\ldots\ldots \text{(1.3)}$$

where ŷ is the predicted value.

The equation 1.3 provides the expression of categorical crossentropy to find the loss function. Categorical crossentropy will compare the distribution of the predictions (the activations in the output layer, one for each class) with the true distribution, where the probability of the true class is set to 1 and 0 for the other classes [8]. To put it in a different way, the true class is represented as a one-hot encoded vector, and the closer the model's outputs are to that vector, the lower the loss.

VGG16 Architecture

VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR (ImageNet) competition in 2014 [11]. VGG-16 is a convolutional neural network architecture, its name VGG-16 comes from the fact that it has 16 layers. Its layers consists of Convolutional layers, Max Pooling layers, Activation layers, Full connected layers. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and MaxPool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC (fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx.) parameters.

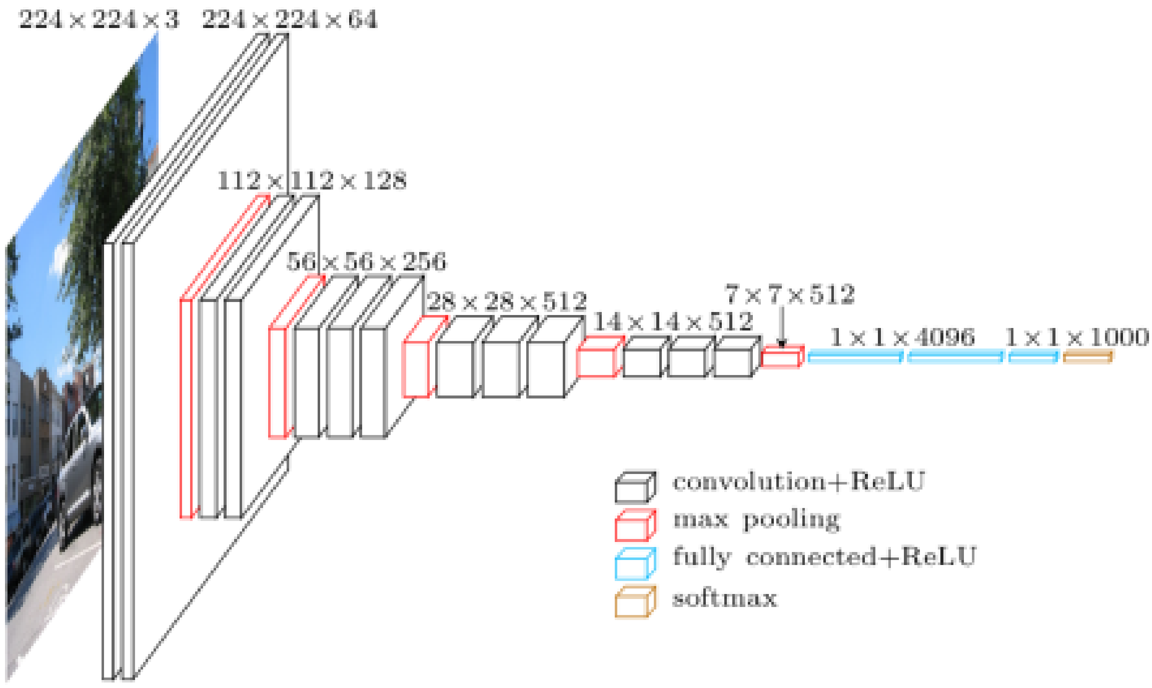


Fig 1.3: VGG16 Architecture

[Source: Published as conference paper at ICLR 2015]

The input to cov1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. Layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2.

Three Fully-Connected (FC) layers follow a stack of convolutional layers (which has a different depth in different architectures): the first two have 4096 channels each, the third performs 1000-way ImageNet Large-Scale Visual Recognition Challenge ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks.

All hidden layers are equipped with the rectification (ReLU) non-linearity. It is also noted that none of the networks (except for one) contain Local Response Normalization (LRN), such normalization does not improve the performance on the ILSVRC dataset, but leads to increased memory consumption and computation time.

All configurations follow the generic design present in architecture and differ only in the depth: from 11 weight layers in the network. A (8 conv. and 3 FC layers) to 19 weight layers in the network E (16 conv. and 3 FC layers). The width of conv. layers (the number of channels) is rather small, starting from 64 in the first layer and then increasing by a factor of 2 after each max-pooling layer, until it reaches 512 [11].

Our motivation is concerned to build an efficient and reliable system which systematically distinguishes the garbage image into different waste classes and provides assistance for proper recycling of that classified waste. So, we investigate the different types of neural networks (NN) to classify the garbage waste images into four classes: glass, paper, metal, cardboard and plastic.

1.2 Motivation

In regards to garbage waste management in Nepal, there is no proper methodology for the classification and management of waste materials. So, the main purpose of this work is to demonstrate an efficient intelligent system to classify selected classes of common waste materials.

1.3 Statement of Problems

The existing waste management system has no proper technical mechanism for the classification and management of garbage wastes. Following problems can be seen in present system:

1. No proper classification of wastes.
2. The disposal and management of wastes are manually performed.

1.4 Objectives

The main objective of this proposed project is to build an efficient system to classify common waste materials into different waste classes.

1.5 Scope and Limitations

The proposed classification of garbage waste system focuses on executing the proper classification of garbage into different waste classes using an efficient, intelligent and computerized system.

Some limitations are as follows:

It only distinguishes five waste classes.

It works on still images and not real time.

1.6 Structure of Project Report

This report includes following chapters:

- Chapter 1 includes the introduction of the project including background of the study, objectives, problem statement, motivation and scope of the project.
- Chapter 2 presents the literature reviews of the project work.
- Chapter 3 presents the methodology used for the development of the project.
- Chapter 4 shows the results and necessary discussions of the project work.
- Chapter 5 presents the conclusion and future recommendations for the project.

CHAPTER 2

LITERATURE REVIEW

Garbage has become a major problem worldwide due to uncontrolled disposal of household waste from citizen's home and industries without an effective and efficient waste management program that can result in health risks and a negative impact on the environment [1]. A waste management with efficient classification play an important role in ecologically sustainable development by ensuring that waste is properly disposed of. Efficient selective collection is often implemented to improve recycling and reduce environmental impact, especially in developing countries where waste management is a serious problem for economic development [2].

Over the years, many works have been implemented with the aim of minimizing the impact of the waste uncontrolled disposal of wastes. Technologies such as Radio Frequency Identification (RFID) and Sensor Network (SN) have been used to provide a new way to optimize waste management systems [3]. Various scholars from around the world have conducted different projects related to proper management and classification using distinct algorithms and technologies as mentioned above.

Among various research paper, one such paper contributed to classification of trash is a thesis report by Mindy Yang and Gary Thung. They initiate their paper by introducing about their work and describing their methodology. The models they used are support vector machines (SVM) with scale-invariant feature transform (SIFT) features and a convolutional neural network (CNN) [12]. The objective of their project was to take images of a single piece of recycling or garbage and classify it into six classes consisting of glass, paper, metal, plastic, cardboard, and trash. They created a dataset that contains around 400-500 images for each class, which was hand collected.

They used the Torch7 framework for Lua to construct CNN. They implemented an eleven layer CNN that is very similar to AlexNet [12]. Their network is smaller than AlexNet (using 3 4 of the amount of filters for some convolutional layers) because of computational constraints. Their project concluded of five convolutional layer of 288, 288, 192, 192 and 96 filters and 3 max-pooling layers of size 3x3 and 3 fully connected layers with 4096 neurons and 5 neurons. They also used support vector machines (SVM) with scale-invariant feature transform (SIFT) features. The CNN was trained with a train/val/test split of 70/13/17, an image size of 256x256, 60 epochs, a batch size of 32, a learning rate of 5e-8, 5e-1 weight decay every 5 epochs, an L2 regularization strength of 7.5e-2 [12]. Their experiment results provided that the SVM achieved better results than the CNN. It achieved a test accuracy of 63% while CNN achieved a test accuracy of only 22%.

Our project is also based on the same concept of proper waste management through a computerized system. The system we intended to build uses CNN architecture and is divide into 5 categories of waste classes. Our project mainly focuses on providing a platform for the assortment of garbage wastes in selected classes which in turn offer for good management of that wastes. This project aims to take garbage waste images and classify them into five classes: glass, paper, metal, cardboard and plastic. Our system is contrived to achieve its objective by exercising artificial intelligence and its tools and utilization of neural networks. The justification of this project is to provide an intelligent and efficient system for rightful categorization of garbage waste resulting in proper recycling cycle of wastes.

CHAPTER 3

METHODOLOGY

3.1 Background

For developing the system for classification of garbage waste into different classes various phase and methods has to be proceeded with the help of various software, tools and languages. In our project we initially collected related data in form of images and perform analysis of the proposed system, designing and development of complete system. We trained the images and later utilized it to categorize the image into different classes. We studied the concept of ANN and CNN and employed VGG-16 architecture as the base for our model.

3.2 System Block Diagram

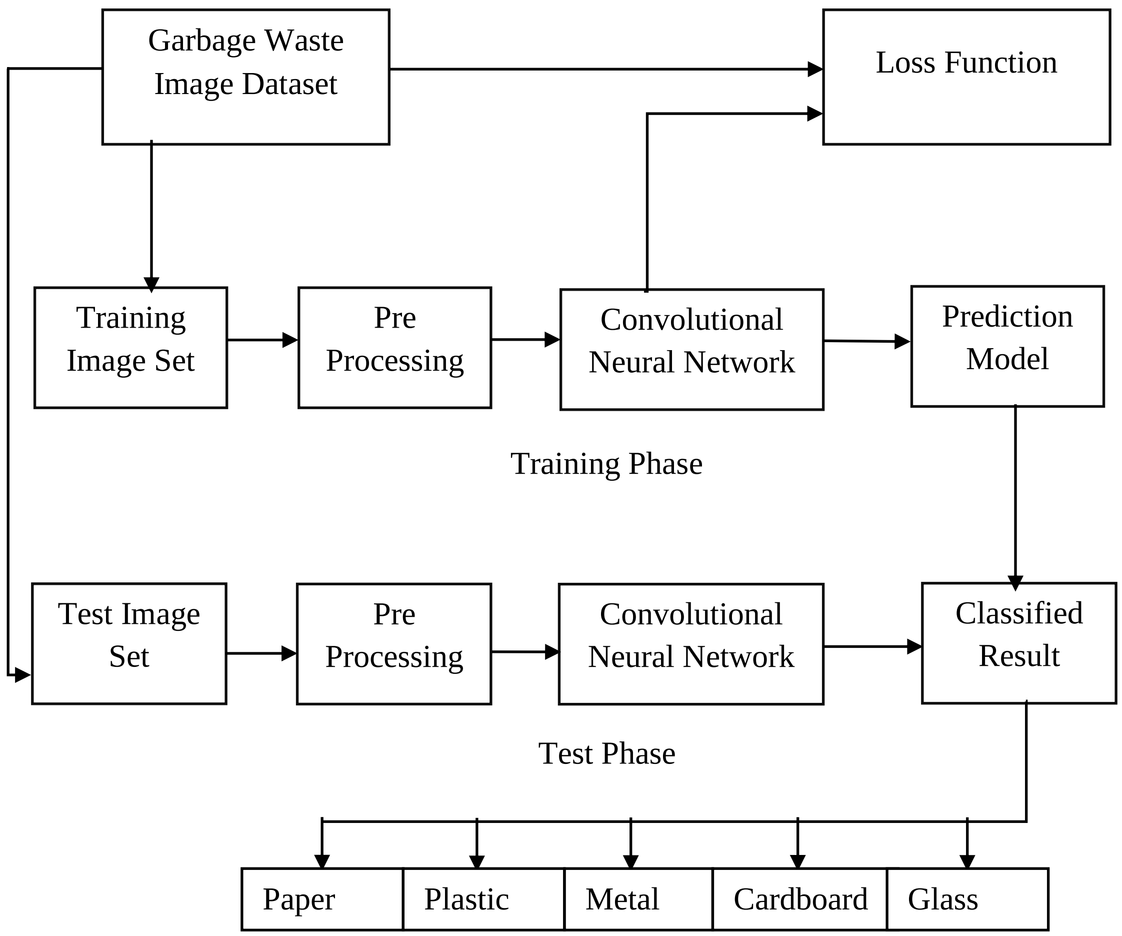


Fig 3.1: System Block Diagram for Garbage Classification into Different Waste Material Classes System

Our system is based on the architecture of neural networks and its components functionalities. Our system block diagram major entities are image dataset which is divided into train set images and test set images. They both are pre-processed and work through a convolutional neural network with various layers which calculates loss function and predicts the image into classified categorization. We have five classes namely: glass, metal , plastic, paper and cardboard.

3.3 Data Flow Diagram

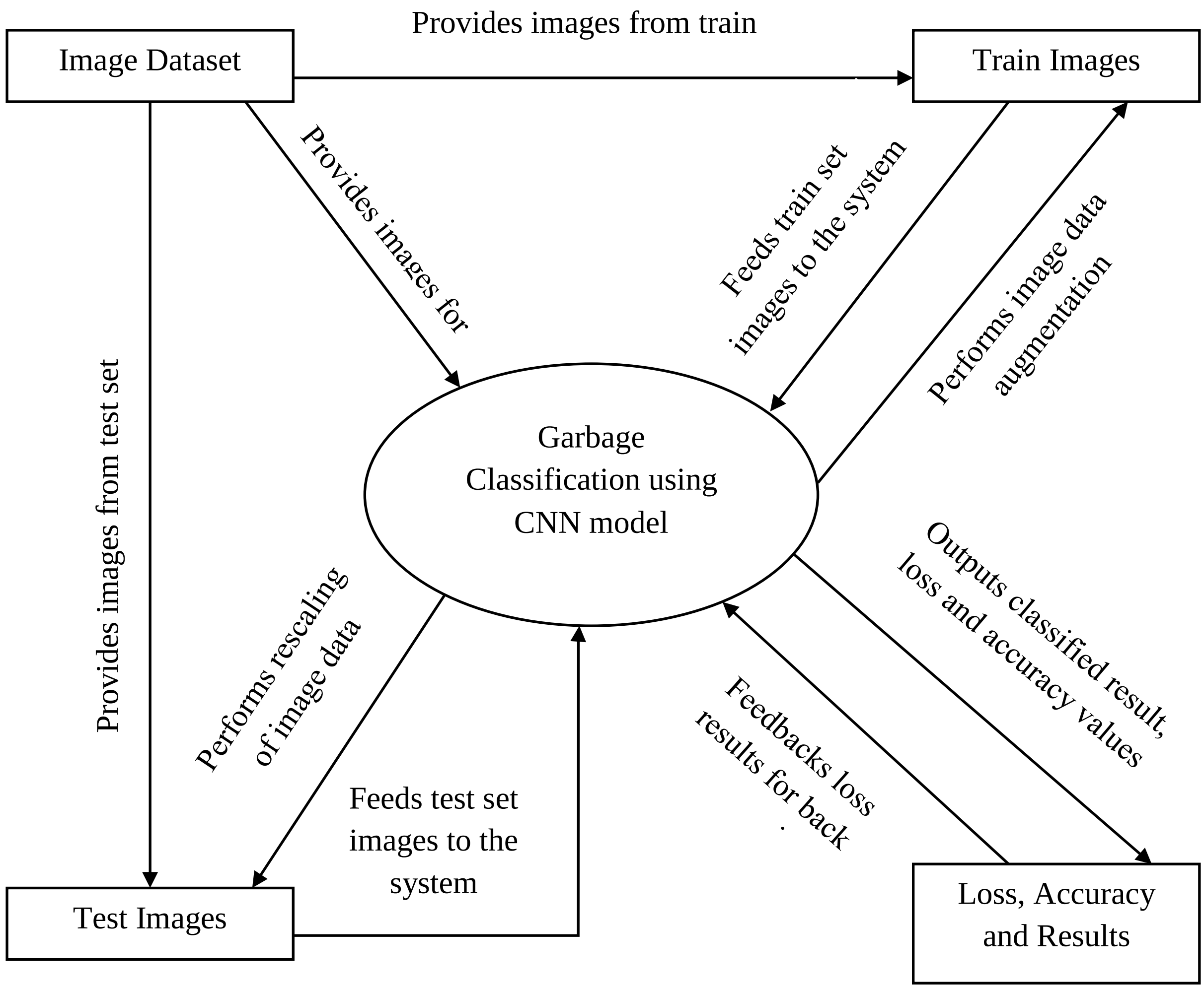


Fig 3.2: Context Diagram for Garbage Classification into Different Waste Material Classes System

3.4 Algorithm

Step 1: Start

Step 2: Feed the images of the garbage material object

Step 3: Identify by the image using CNN model

Step 4: Calculate the prediction percentage.

> 4.1 If the prediction percentage is more than 50% than show the class name of highest score of percentage.
>
> 4.2 Else the show invalid input.

Step 5: Stop

3.5 Flowchart

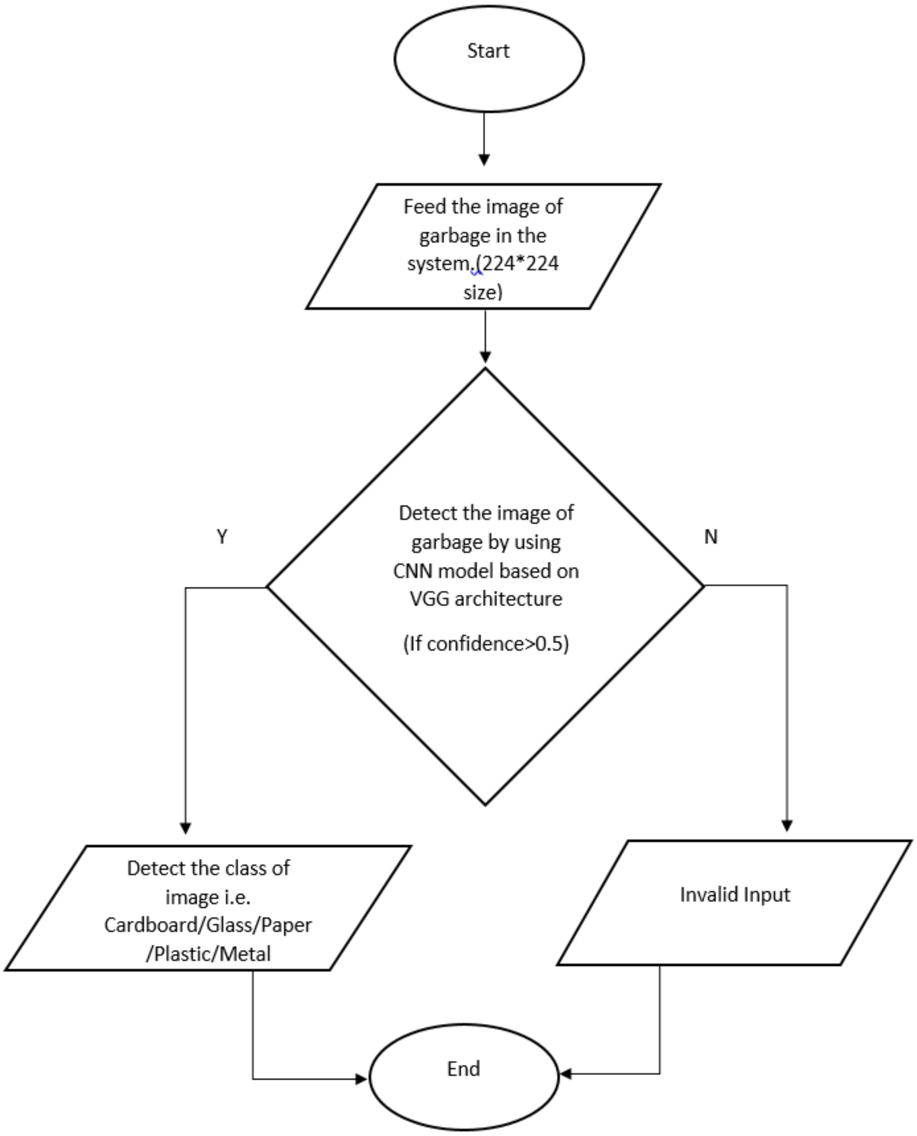


Fig 3.3: Flowchart for Garbage Classification into Different Waste Material Classes

3.6 Model and Methods

A Convolutional Neural Network (CNN) is a multilayered neural network with a special architecture to detect complex features in data. CNNs have been used in image recognition, powering vision in robots, and for self-driving vehicles.

CNN is basically composed of convolutional layer, applying ReLu, pooling layer, flattening and fully connected.

- Convolutional: Convolution is an element-wise multiplication. The purpose of the convolution is to extract the features of the object on the image locally.
- Applying ReLu (Rectified Linear Unit): The output is subject to an activation function to allow non-linearity. The usual activation function for convnet is the Relu. All the pixel with a negative value will be replaced by zero.
- Pooling: The pooling layer in CNN progressively reduces the spatial size of the representation to lower the number of parameters in the convolutional neural network. There are basically 2 types of pooling:

    Max pooling: Extract the maximum value of the patch from the feature map.

    Average pooling: Extract the average of all patches in the feature map.

- Flattening: It involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing.
- Fully connected: Here we connect all neurons from the previous layer to the next layer. We use a softmax activation function to classify the number on the input image.

Once a CNN is built, it can be used to classify the contents of different images. All we have to do is feed those images into the model. Just like ANNs, CNNs are inspired by the workings of the human brain. CNNs are able to classify images by detecting features, similar to how the human brain detects features to identify objects.

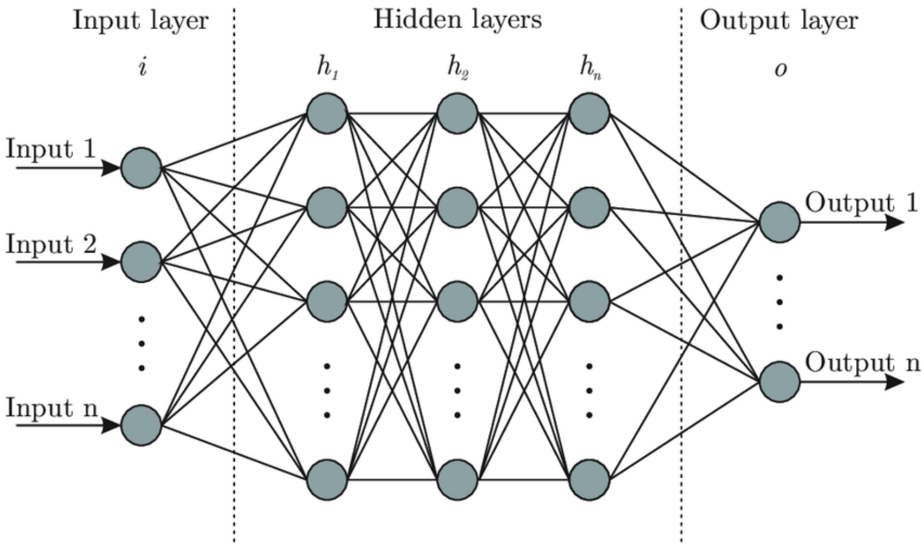


Fig 3.4: Convolutional neural network

[Source: https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051]

VGG16 Model

One of the more popular Convolutional Network architectures is called VGG-16, named such because it was created by the Visual Geometry Group and contains 16 hidden layers. Its layers consists of Convolutional layers, Max Pooling layers, Activation layers, Full connected layers. It is considered to be one of the excellent vision model architecture till date. Right now it had models with 16 to 19 layers variant of VGGNet. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and MaxPool layer of 2x2 filter of stride 2. VGG used small filters because of fewer parameters and stack more of them instead of having larger filters. VGG has smaller filters with more depth instead of having large filters. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC (fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx) parameters.

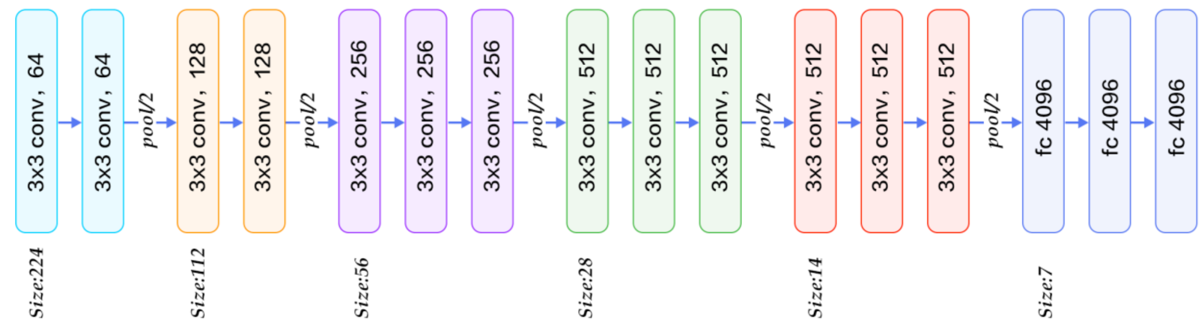


Fig 3.5.: VGG16 Architecture

[Source: chttps://www.quora.com/What-is-the-VGG-neural-network]

The input to cov1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. Layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2.

Three Fully-Connected (FC) layers follow a stack of convolutional layers (which has a different depth in different architectures): the first two have 4096 channels each, the third performs 1000-way ImageNet Large-Scale Visual Recognition Challenge ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks.

All hidden layers are equipped with the rectification (ReLU) non-linearity.

The width of conv. layers (the number of channels) is rather small, starting from 64 in the first layer and then increasing by a factor of 2 after each max-pooling layer, until it reaches 512.

Modification on the VGG16 Model

Due to lots of problem in VGG 16 architecture model such as limited hardware resources, datasets, overfitting underfitting, non-improving accuracy and loss etc. The model could not give the good result for the project so, we have modified the VGG16 model into simpler model which has following architecture and results.

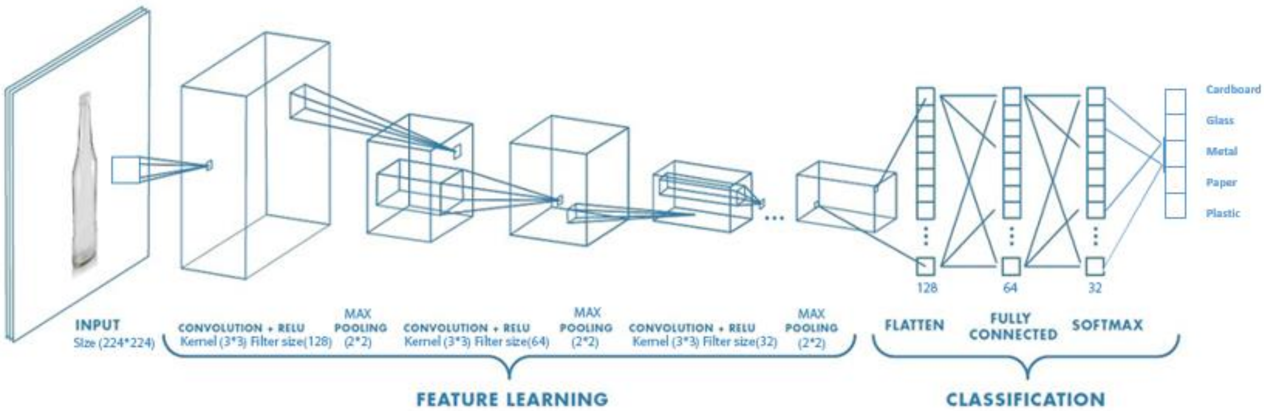


Fig 3.6: Modified VGG166 Architecture of Garbage Classification

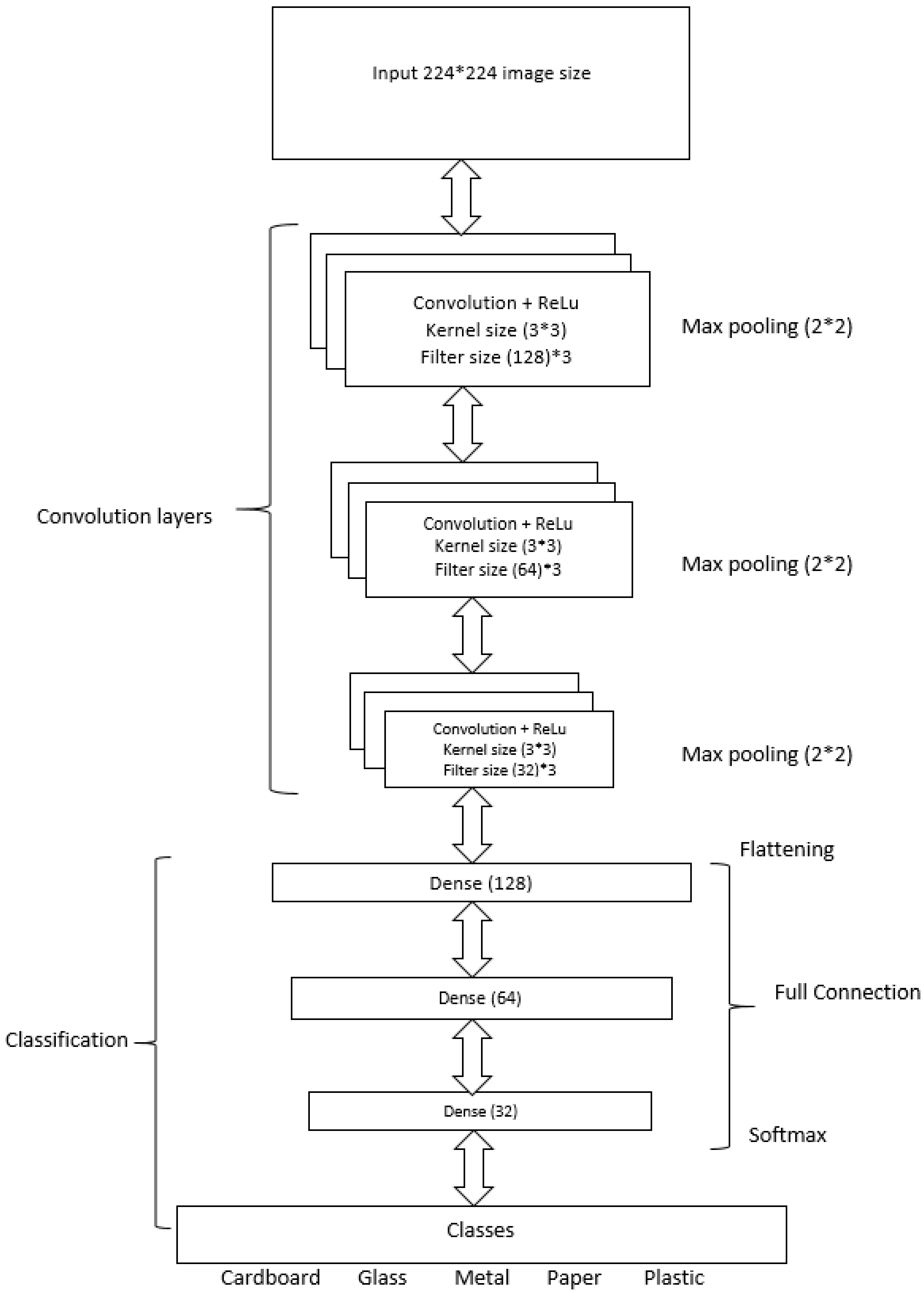


Fig 3.7: System architecture of modified VGG16 model for Garbage Classification

Our project works through a modified VGG-16 architecture. The architecture consists of:

- Layer 0: input image of size 224x224
- Layer 1: Convolution with 128 filters, size 3x3, activation=ReLu
- Layer 2: Convolution with 128 filters, size 3x3, activation=ReLu
- Layer 3: Convolution with 128 filters, size 3x3, activation=ReLu
- Layer 4: Max pooling with size 2x2
- Layer 5: Convolution with 64 filters, size 3x3, activation=ReLu
- Layer 6: Convolution with 64 filters, size 3x3, activation=ReLu
- Layer 7: Convolution with 64 filters, size 3x3, activation=ReLu
- Layer 8: Max pooling with size 2x2
- Layer 9: Convolution with 32 filters, size 3x3, activation=ReLu
- Layer 10: Convolution with 32 filters, size 3x3, activation=ReLu
- Layer 11: Convolution with 32 filters, size 3x3, activation=ReLu
- Layer 12: Max pooling with size 2x2
- Layer 13: Flattening layer
- Layer 14: Fully connected with 128 neurons
- Layer 15: Fully connected with 64 neurons
- Layer 16: Fully connected with 32 neurons
- Result: Softmax score, 5 classes

In this architecture we used 3*3 convolutional layers, the first layer with 128 output filters and 3*3 kernel size, which determinate the width and height of the 2D convolution window. The input to cov1 layer is of size 224x224 RGB image. An important component of the first convolution layer is an input shape, which is the input array of pixels. Further convolution layers are constructed in the same way of 64 and 32 output filters with same kernel size, but do not include the input shape.

The activation function of this model is Relu. This function setts the zero threshold and looks like: $f(x) = \max(0, x)$. If $x > 0$, the volume of the array of pixels remains the same, and if $x < 0$, it cuts off unnecessary details in the channel. We used Max Pooling 2D layer that is pooling operation for spatial data. Numbers 2, 2 denote the pool size, which halves the input in both spatial dimension.

Flattening involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing. Next is Dense — densely connected layer with the value of the output space 128, 64 32 and ‘relu’ activation function. It follows Dropout, which is preventing overfitting. Overfitting is the phenomenon when the constructed model recognizes the examples from the training sample, but works relatively poorly on the examples of the test sample. Dropout takes value between 0 and 1. The last fully connected layer has 1 output and ‘softmax’ activation function. Next step is model compiling, it has a binary cross entropy loss function, which will show the sum of all individual losses. The optimizer algorithm is ‘adam’ is an adaptive learning rate optimization algorithm that's been designed specifically for training deep neural networks. ‘Adam’ is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

Batch size the number of training examples in one forward/backward pass (or for 1 epoch, which is expected).Then the already described Image Data Generator is added for training and testing samples. But it has a new transformation, which is called rescale. It multiplies the data by the given value. The flow from directory method is added for training and testing data. First, the path to the folders is specified. Further, the target size follows. It shows width and height to which images will be resized. Next, the batch size is added. Finally binary class mode is set.

When the preparation is complete, the code fragment of the training follows:

Training is possible with the help of the fit generator**.** Here it is important to indicate a number of epochs, which defines for how many times the training will repeat. 1 epoch is 1 forward pass and 1 backward pass over all the training examples.

Also, in this section steps per epoch and validation steps are set. Steps per epoch (or number of iterations) shows total number of steps, which is used to declare one epoch finished and begin the next.

The samples of images in the dataset is about 8018 in number. The dataset contains images of recycled objects across five classes with about 1400-1900 images each totaling about 8018 train images and about 350-500 images each totaling about 2012 test images. This was done to maintain the ratio of 6:4 or simply 60% for training purpose and 40 % for validation. The Figure below shows the ratio of images in training and test set. We trained the model through modified VGG-16 structure which feed on images of size 224x224, epochs 100, and learning rate of 0.0001.

3.7 Data Set

The data acquisition process was done by collecting images from Google images. Originally we were using the Flickr Material Database and images from Google Images. The dataset we created consists of 8018 images out of which 8018 images. The dataset contains images of recycled objects across five classes with about 1400-1900 images each totaling about 8018 train images and about 350-500 images each totaling about 2012 test images. This was done to maintain the ratio of 8:2 or simply 80% for training purpose and 20 % for validation. Data augmentation techniques were performed on each image because of the small size of each class. These techniques included random rotation of the image, random brightness control of the image, random translation of the image, random scaling of the image, and random shearing of the image. These image transformations were chosen to account for the different orientations of recycled material and to maximize the dataset size.

```
1461 1440 1625 1600 1892
8018
```

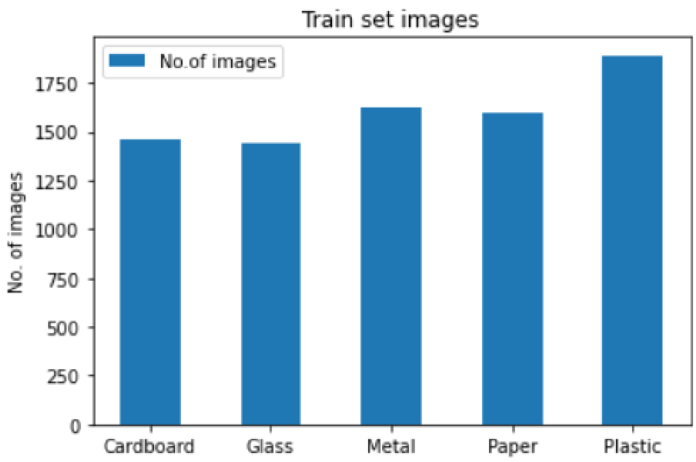


Fig 3.8: Bar Chart for Train Images

```
366 363 407 402 474
2012
```

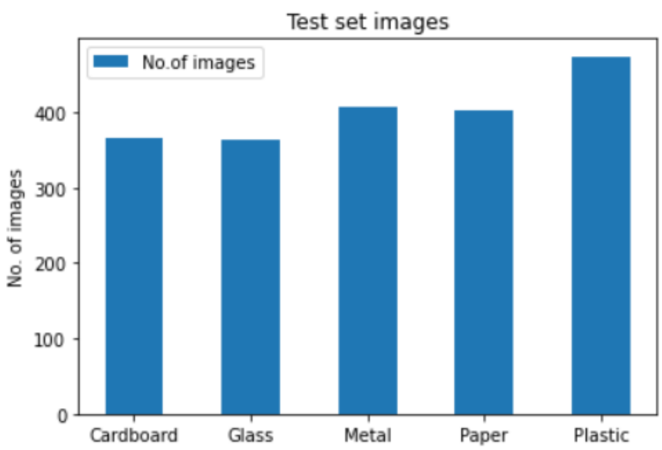


Fig 3.9: Bar Chart for Test Images

3.8 Tools and Platforms

1. Python Programming Language
2. Anaconda
3. Keras with TensorFlow backend
4. Python Packages : Matplotlib – allows to create graphs
   Skimage – image manipulation
   PIL – adds support for opening, manipulating, and saving images.

5. OS Environment – Windows
6. Hardware: CPU: Intel core i7 7^{th} Gen 7700HQ @2.8GHZ
   GPU: Nvidia GTX 1050
   RAM: 8GB
   OS: Windows 10
7. Documentation Tools
   - Microsoft Word
   - Microsoft PowerPoint

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Overview

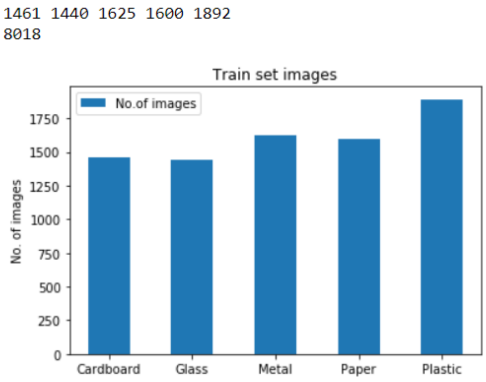
We have thoroughly completed the works related to our project as per our proposal. Using the concept ANN, CNN and its functionalities such as activation function, loss function, back propagation, VGG16 architecture and programming we have been able to get the expected result. However, some errors can be seen in prediction model. We implemented modified VGG 16 architecture for better accuracy and less loss. The model was fed a dataset of 8018 images and trained and tested for 100 epochs, with a batch size 1, a learning rate of 0.0001 which provided a validation accuracy of 89.86% with a validation loss of 11.71%.

4.2 Results and Discussions

Experimental Analysis

Our program is now able to differentiate various materials into degradable and non-degradable categories. We were able to achieve this by using the concept of ANN and CNN. We collected multiple images and created a dataset for training as well as for testing and prediction. We created five set of convolutional layers with activation function ReLu. Then we trained the dataset for 50 epochs and early stop function was also added to stop the training if the loss is not improved. Later we tested the dataset and predicted that whether the material is degradable or non-degradable for a number of images.

We trained an image dataset with VGG-16 in Keras. We followed the VGG16 architecture model with number of train images 8018 and test images 2012. The results are shown below:



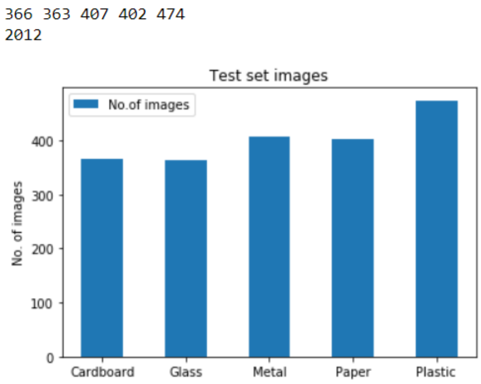


Fig 4.1: Bar Chart For Train Images for VGG16 Model

Fig 4.2: Bar Chart For Test Images for VGG16 Model

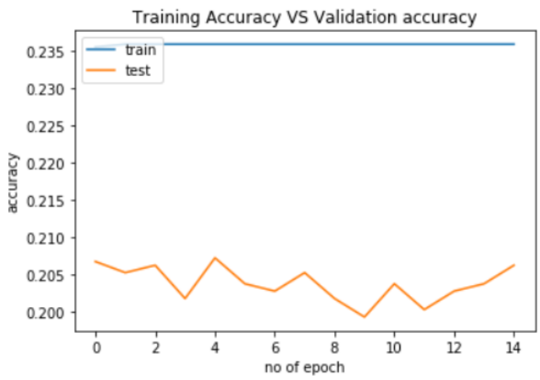


Fig 4.3: Training Vs Validation Accuracy for VGG16 Model

Fig 4.4: Training Vs Validation Loss for VGG16 Model

The number of samples of images for training and testing is 8018 and 2012 respectively photos of cardboard, plastic, paper, metal and glass combined. The model was trained with train/test split of 60/40, an image size of 224x224, 50 epochs and a learning rate of 0.001. After finishing epochs, we found the accuracy: 0.3236 with loss: 1.6052 and validation accuracy: 0.2360 with validation loss: 0.3236, it shows the ability of the model to generalize to new data. The following graph shows the training Vs validation accuracy and loss. The graph of accuracy clearly shows the huge difference in training and validation accuracy which is very poor for our system. The validation loss is greater than validation accuracy. VGG 16 architecture introduced overfitting for our system so, we modified the VGG16 architecture.

The samples of images in the dataset is about 8018 in number. The dataset contains images of recycled objects across five classes with about 1400-1900 images each totaling about 8018 train images and about 350-500 images each totaling about 2012 test images. This was done to maintain the ratio of 6:4 or simply 60% for training purpose and 40 % for validation. The Figure below shows the ratio of images in training and test set. We trained the model through modified VGG-16 structure which feed on images of size 224x224, epochs 100, and learning rate of 0.0001.

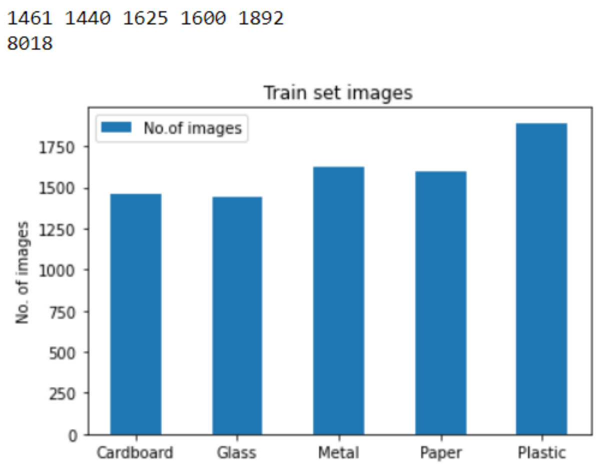


Fig 4.5: Bar Chart for Train Set Images For Modified VGG16 Model

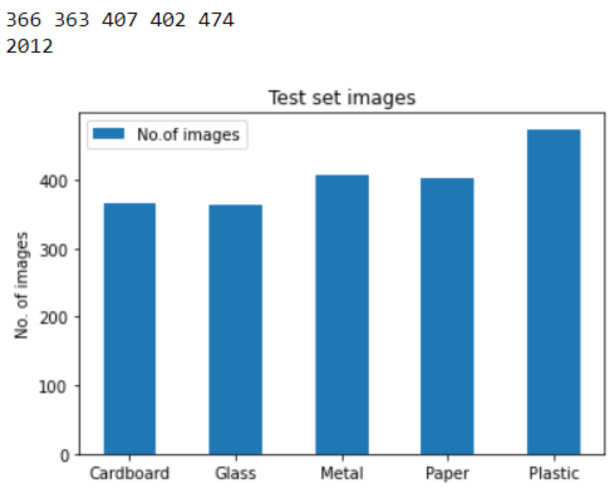


Fig 4.6: Bar Chart for Test Set Images For Modified VGG16 Model

When the model is trained, it should be saved with save weights. After finishing epochs, the accuracy: 0.8669 with loss: 0.3814 and validation accuracy: 0.9006 with validation loss: 0.7400, it shows the ability of the model to generalize to new data. The following graph shows the training Vs validation accuracy and loss.

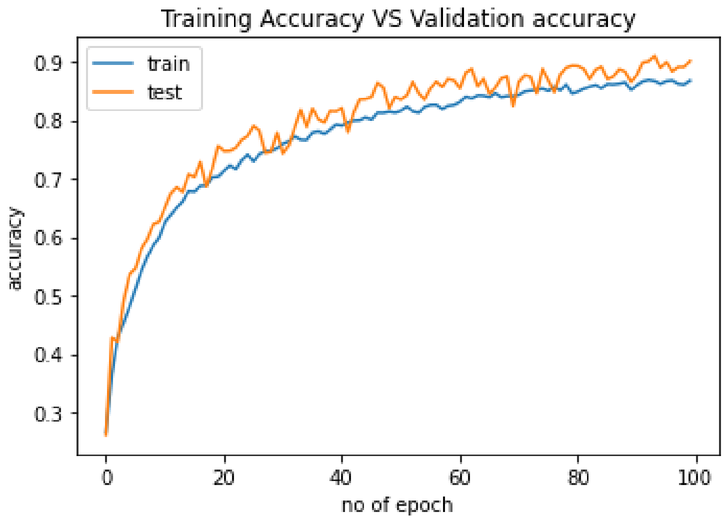


Fig 4.7: Training Vs Validation Accuracy for Modified VGG16 Model

Fig 4.8: Training Vs Validation Loss for Modified VGG16 Model

We can observe in fig 4.7, the training accuracy and validation accuracy and increasing almost simultaneously, which is taken as good training. However, there is a sharp fluctuation in validation loss in fig 4.8.

After finishing the epochs it is found that the validation loss does not improve from 0.1171. The validation accuracy at loss 0.1171 is 0.8986. The accuracy: 0.8661 with loss: 0.3839. One of the reasons for the sharp fluctuation in validation loss may be due to difference in image configuration of garbage images. It is hard for the model to extract features since almost every image is different from one another. There is no general shape to extract. They are different in shape, size, shining, texture. It may also be due to limited hardware resources and limited dataset.

Results

The modified VGG16 architecture provide better validation accuracy and less validation loss than the VGG16 model but there is fluctuation in the validation loss. In the previous model of VGG16, the validation loss is greater than validation accuracy, which was due to overfitting so we added a dropout layer in the modified model.

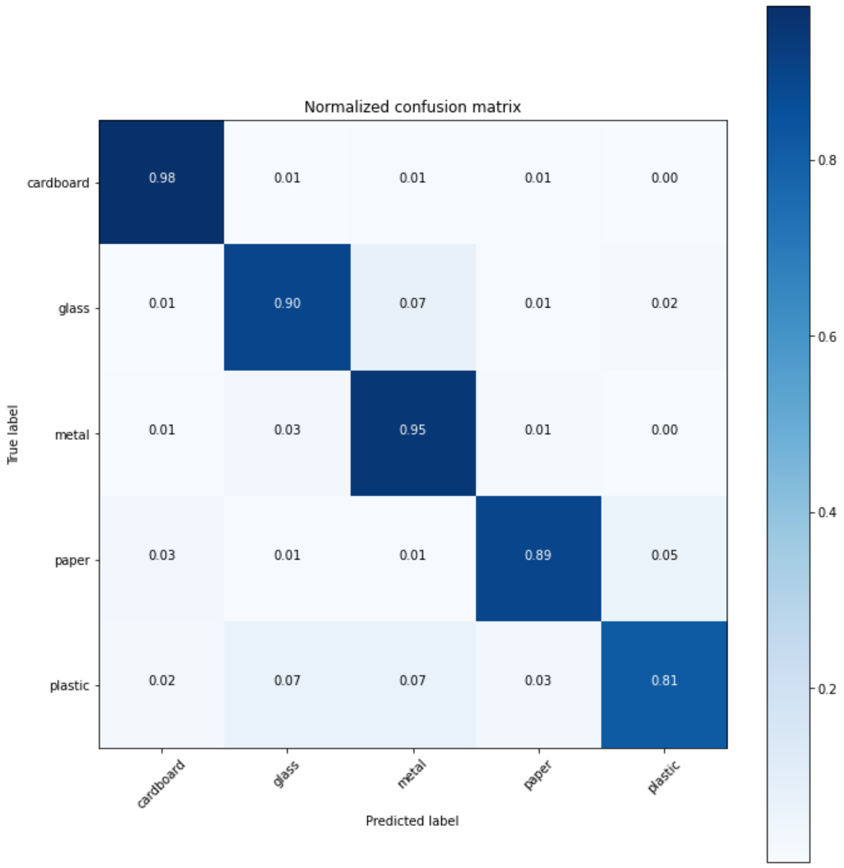


Fig 4.9: Normalized Confusion Matrix of Modified VGG16 Model

A confusion matrix is a technique for summarizing the performance of a classification algorithm.

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The general idea is to count the number of times instances of class A are classified as class B. For example, to know the number of times the classifier confused images of 5s with 3s, you would look in the 5th row and 3rd column of the confusion matrix. Each row in a confusion matrix represents an actual class, while each column represents a predicted class.

Let’s take an example of a confusion matrix for a binary classifier: A 2x2 matrix denoting the right and wrong predictions might help us analyze the rate of success.

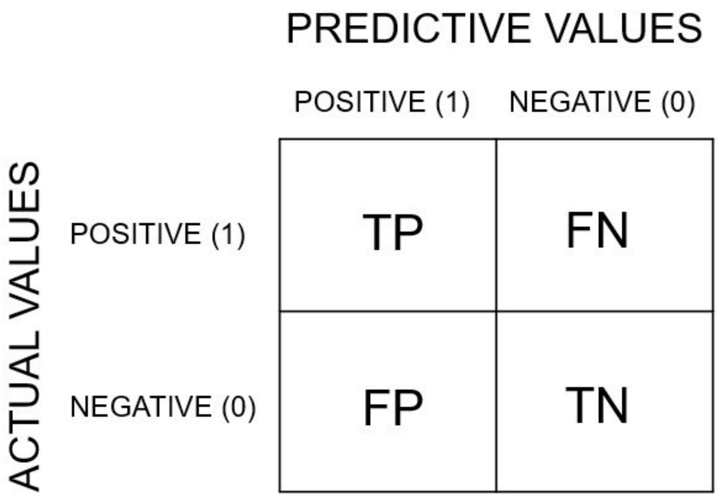


Fig 4.10: A Confusion Matrix for Binary Classifier

To understand confusion matrix we first must be familiar with the four parameters i.e. TP, TN, FP, FN. If C represents a confusion matrix and i and j represents the rows and columns of matrix then,

- **TN (True Negative)[1][1]:** represents the values which are predicted to be false and are actually false.
- **FP (False Positive) [1][0]:** represents the values which are predicted to be true, but are false.
- **FN (False Negative) [0][1]:** represents the values which are predicted to be false, but are true.
- **TP (True Positive) [0][0]:** represents the values which are predicted to be true and are actually true.

Most performance measures such as precision, recall are calculated from the confusion matrix**.**

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

Precision = TP/TP+FP

Recall is the ratio of correctly predicted observations to all observations in actual class.

Recall = TP/TP+FN

F1 score is the harmonic mean between precision and recall. It is a measure of test accuracy.

F1 = 2*((precision*recall)/(precision + recall))

Material	Precision	Recall	F1 score
Glass	0.89	0.88	0.88
Metal	0.95	0.86	0.90
Paper	0.90	0.94	0.92
Plastic	0.81	0.92	0.86
Cardboard	0.97	0.93	0.95

Table 4.1: Precision, Recall and F1 Score of Each Class

The above table depicts the precision value, recall value and f1 score of every class which was calculated with the help of confusion matrix.

Prediction Test

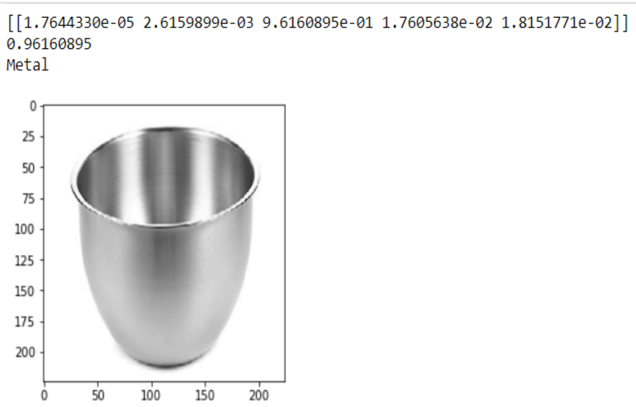


Fig 4.11: Prediction of Image 1

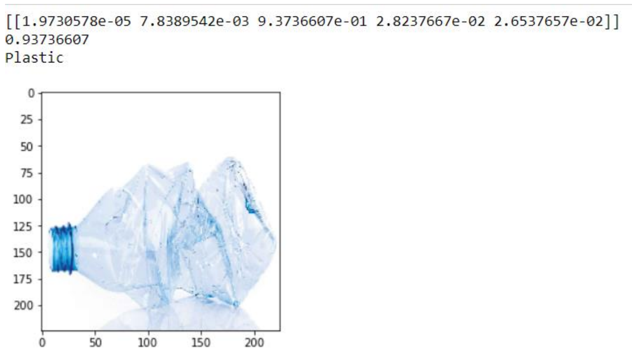


Fig 4.12: Prediction of Image 2

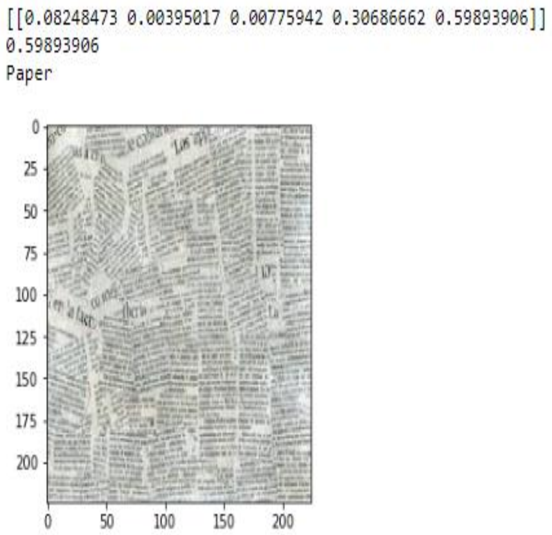


Fig 4.13: Prediction of Image 3

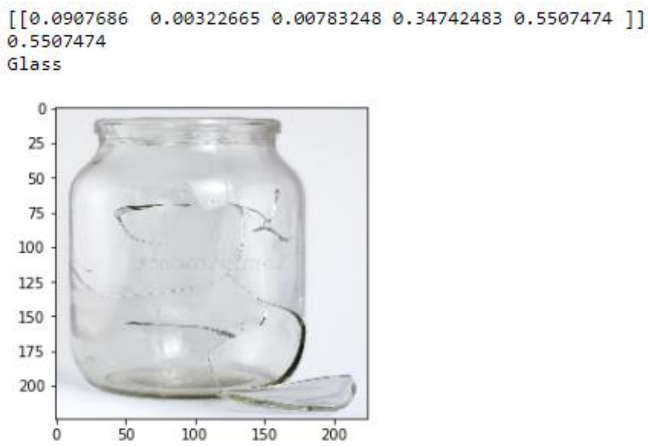


Fig 4.14: Prediction of Image 4

Fig 4.15: Prediction of image 5

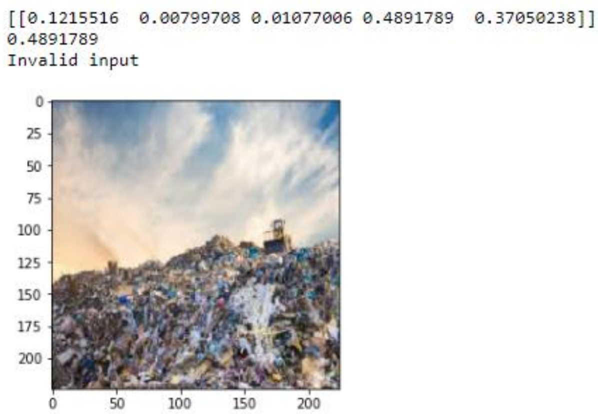


Fig 4.16: Prediction of Image 6

The prediction test predicts whether the image belongs in n of the classes. It is by comparing the prediction percentage of each class with the threshold prediction percentage (0.3). If any class has greater than 50% prediction rate then that image belongs to that class.

Error in Prediction

```
[[7.5626336e-02 1.6742999e-04 9.2420435e-01 1.0815184e-08 1.9257054e-06]]
0.92420435
Metal
```

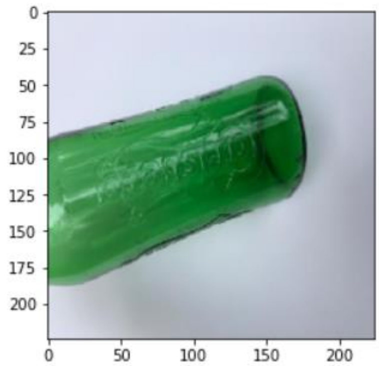


Fig 4.17: Prediction of Glass

```
[[1.21960334e-01 8.33623199e-05 8.77956212e-01 9.35998293e-11
  1.38185428e-07]]
0.8779562
Glass
```

Fig 4.18: Prediction of Plastic

Discussion

It can be seen that the modified VGG16 architecture performs better than the VGG 16 model. In VGG16 model, there was a huge gap between training and validation accuracy as well as in validation and training loss. The learning accuracy was greater than validation accuracy which was caused due to overfitting. After adding a dropout layer and modifying the VGG16 architecture we found that the learning and validation accuracy increases simultaneously and there is sharp fluctuation in validation loss at many points. This may have occurred due to the sharp variance in the image properties of the dataset which contributed difficulties in extracting the similar features which helps to identify and predict the image. This can be improved by feeding large datasets which require high powered hardware resources.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

The system is able to distinguish the garbage image into different waste classes. The system was able to achieve its goal objective. However, there have occurred some errors which can be diminished through high hardware resources and large datasets. This project was provided us with the knowledge of convolutional neural networks and its various architectures along with its functionalities and working mechanisms. We were able to get familiarized with python programming and artificial intelligence and their impact in human lives. Different performance measures, their computations and their understanding.

5.2 Future Recommendations

Some future recommendations for the system are as follows:

- This system can be implemented in real time using models like YOLO (You Only Look Once).
- The waste classes can be more distinguishable.
- The system can be implemented on a sensor robot for practical application.

CHAPTER 6

REFERENCES

1. https://www.semanticscholar.org/paper/Overview-for-solid-waste-bin-monitoring-and-system-Islam-Arebey/1c4288d41f9041168038273ef2e50931beed246c[visited on November 19,2019]

2. https://hal.inria.fr › hal-00924270 › document [visited on November19,2019]

3. https://www.researchgate.net/publication/45630693_Integrated_technologies_for_solid_waste_bin_monitoring_system [visited on November 18, 2019]

4. https://www.techradar.com/news/what-is-a-neural-network [visited on March 14, 2020]

5. https://trendytechz.com/deep-neural-network-plays-major-role-finding-obstacles-near [visited on March 14, 2020]

6. https://searchenterpriseai.techtarget.com/definition/neural-network [visited on March 14, 2020]

7. https://becominghuman.ai/simple image classification using convolutional neural network-Deep learning in python [visited on March 14, 2020]

8. https://towardsdatascience.com/The complete beginner's guide to deep learning-convolutional neural network and image classification [visited on March 14, 2020]

9. https://medium.com/image classify with convolution neural network and python [visited on March 14, 2020]

10. https://becominghuman.ai/ understanding the basics of cnn with image classification [visited on March 15, 2020]

11. Classification of trash for recyclability status, Stanford University [Mindy, Gary] [published as thesis paper in 2016] [visited on April 20, 2020]

12. Visual Geometry Group, Department of Engineering Science, University of Oxford [Karen, Andrew] [published as conference paper at ICLR 2015] [visited on March 15, 2020]

13. https://www.geeksforgeeks.org/confusion-matrix-machine-learning [visited on May 26,2020]

14. https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/ [visited on May 26,2020]

15. https://towardsdatascience.com/the-5-classification-evaluation-metrics-you-must-know-aa97784ff226 [visited on May 26,2020