Waste classification using Computer Vision👀

Group 6
[Nipun Haldar, Rahul Hebbar, Hargovind Singh]

Mentor - Dr. M. Senthil Sivakumar

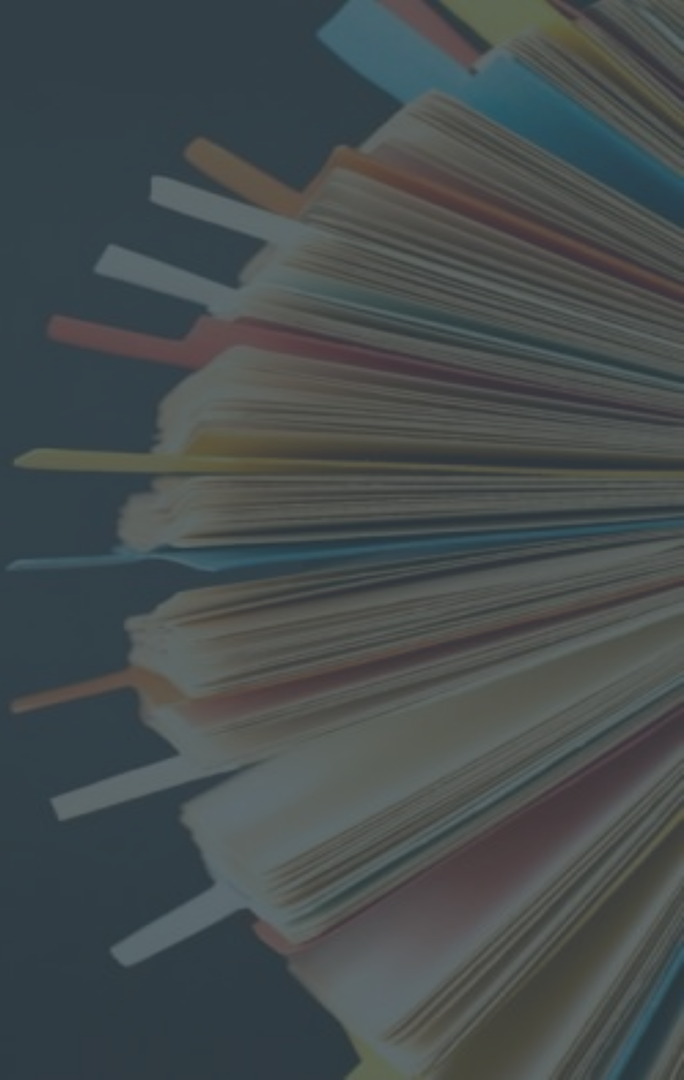
The project can be found at this GitHub link:
🔗 https://github.com/nipun24/waste-classification

ABSTRACT

- Waste management is one of the primary problem that the world faces irrespective of the case of developed or developing country.
- For eliminating or mitigating the garbage’s and maintains the cleanness, it requires smartness based waste management system.
- There are three main reasons for this:
  - In IoT applications,
  - Computer systems can learn to perform tasks, such as classification, clustering, predictions, and pattern recognition.
  - Feature extraction on the large amount of data generated

This project aims to understand the use of machine learning, artificial intelligence and IoT in the most potential areas and the ultimate need to completely replace the human interaction. Machine learning (ML) provides effective solutions, such as regression, classification, clustering, and correlation rules perception for IoT-based waste management.

Existing Work



Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

- Multilayer neural networks trained with the back-propagation algorithm constitute the best example of a successful gradient based learning technique.
- Given an appropriate network architecture, gradient-based learning algorithms can be used to synthesize a complex decision surface that can classify high-dimensional patterns, such as handwritten characters, with minimal preprocessing.
- Convolutional neural networks, which are specifically designed to deal with the variability of 2D shapes, are shown to outperform all other techniques.
- A new learning paradigm, called graph transformer networks (GTN), allows such multi module systems to be trained globally using gradient-based methods so as to minimize an overall performance measure.
- A graph transformer network for reading a bank cheque is also described. It uses convolutional neural network character recognizers combined with global training techniques to provide record accuracy on business and personal cheques. It is deployed commercially and reads several million cheques per day.

URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=726791&isnumber=15641

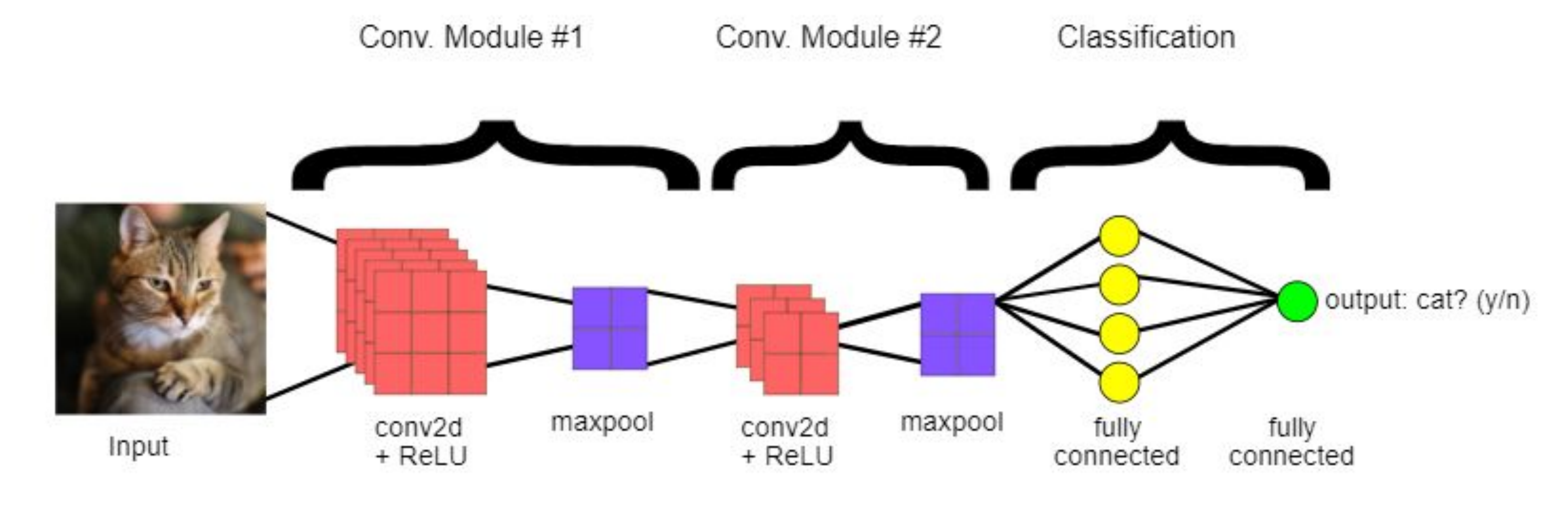
T. Guo, J. Dong, H. Li and Y. Gao, "Simple convolutional neural network on image classification," 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), Beijing, 2017, pp. 721-724, doi: 10.1109/ICBDA.2017.8078730.

- In recent years, deep learning has been used in image classification, object tracking, pose estimation, text detection and recognition, visual saliency detection, action recognition and scene labeling.
- Among different type of models, Convolutional neural networks has been demonstrated high performance on image classification.
- In this paper we builded a simple Convolutional neural network on image classification.
- On the basis of the Convolutional neural network, we also analyzed different methods of learning rate set and different optimization algorithm of solving the optimal parameters of the influence on image classification.

URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8078730&isnumber=8078675

Introduction

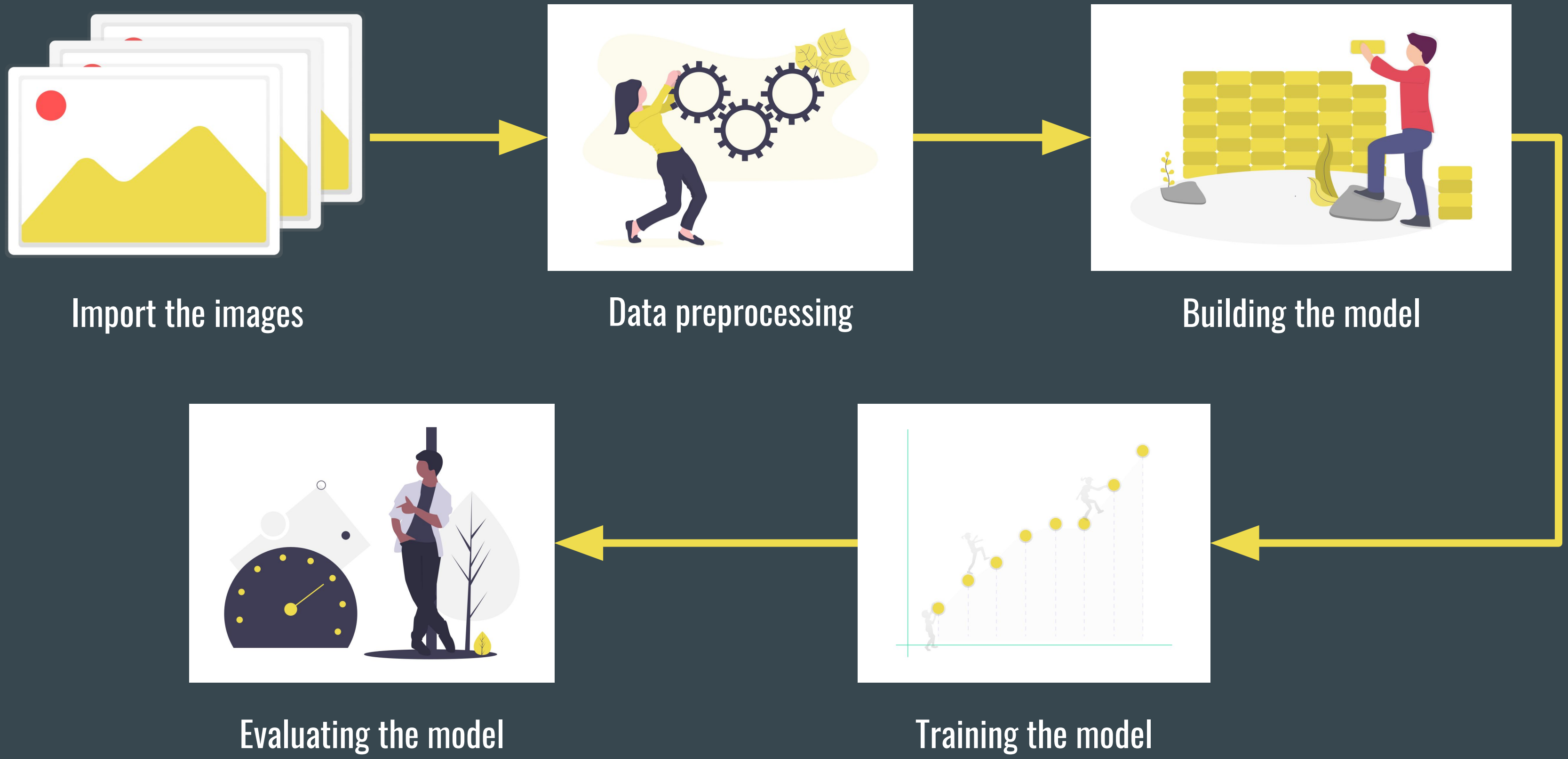
Designing a Convolutional neural network that will be a fully capable of classifying images of waste products as either organic or recyclable.
In final step would be to integrate this with an IoT platform.



Tools and Software used

- Python (language used)
- Kaggle (Training data - https://www.kaggle.com/techsash/waste-classification-data)
- Tensorflow (open source platform for machine learning)
- Tensorboard (visualization toolkit integrating tensorflow)
- Google Colab (google's cloud platform with high compute including GPUs)
- Visual Studio Code (text editor for local machine)
- GitHub (version control)

Software Working Flow



Code Walkthrough

Just 50 lines of code to achieve ~88% accuracy on an augmented dataset of 2,50,000 images!

Importing libraries

```python
from zipfile import ZipFile
import tensorflow as tf
```

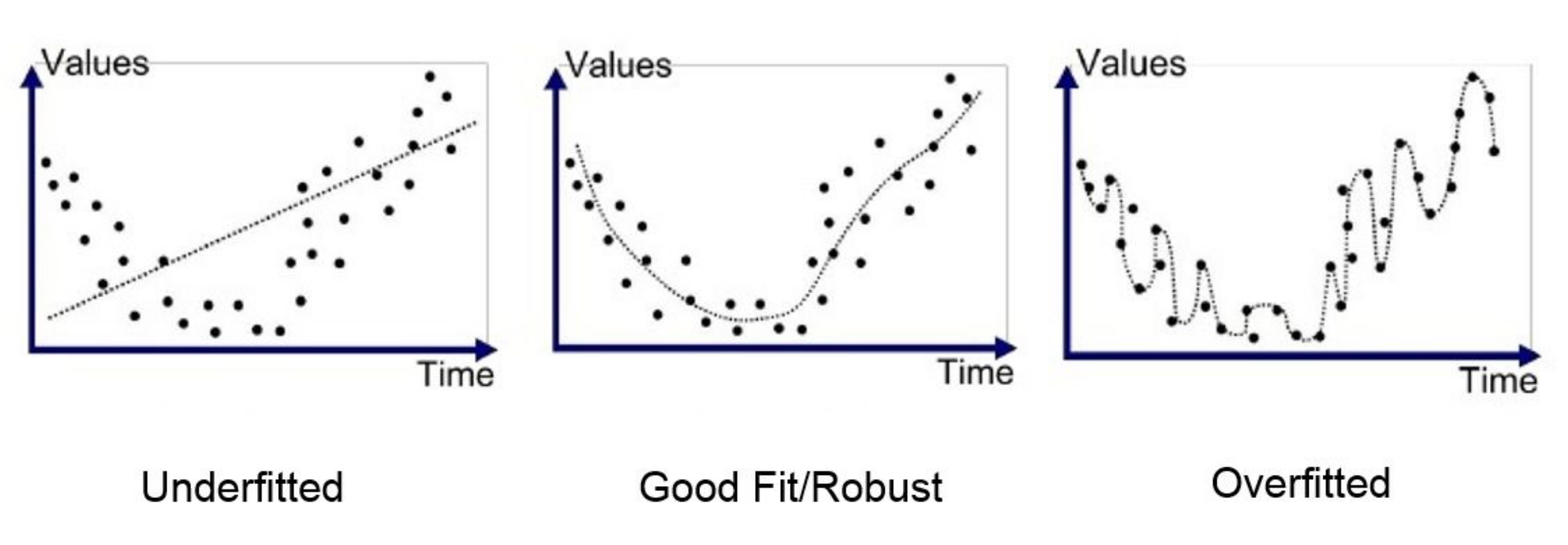
- ***zipfile*** : extract the dataset
- ***tensorflow*** : preprocessing, data augmentation, building the neural network

Extracting the dataset

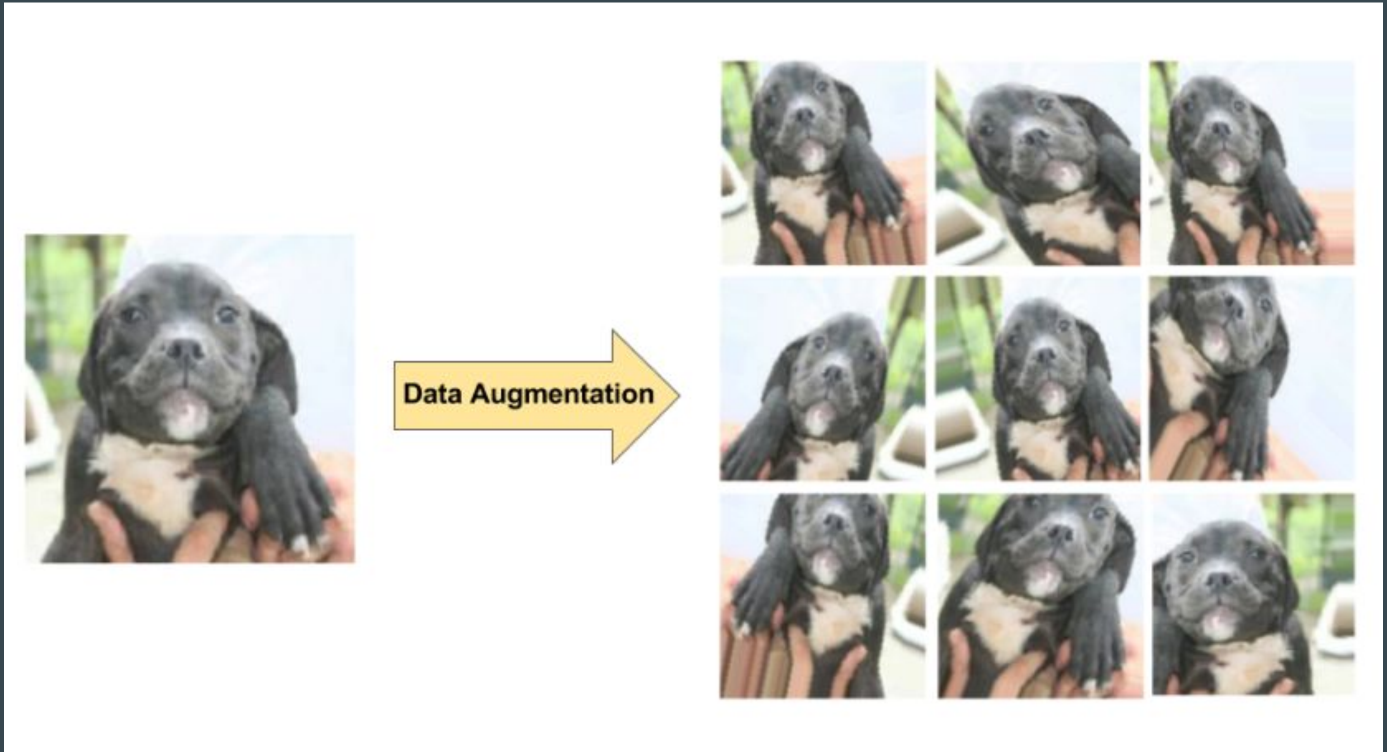
```python
f = ZipFile('DATASET.zip', 'r')
f.extractall()
```

Create a ***ZipFile*** object and extract the Dataset

Underfitting and Overfitting



Data Augmentation



Creating a image augmentor instance

```
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

test_datagen = tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
```

ImageDataGenerator generates batches of tensor image data with real-time data augmentation.

- ***train_datagen*** **:** generates batches of images with *shear, zoom and flips* randomly.
- ***test_datagen*** **:** generates batches of images and normalizes the pixel values.

Getting images from dataset

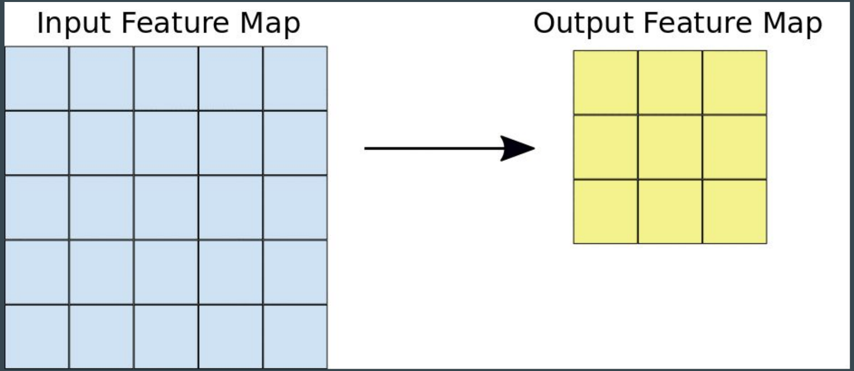
```python
train_generator = train_datagen.flow_from_directory(
    'DATASET/TRAIN',
    target_size=(150, 150),
    batch_size=batch_size,
    class_mode='binary')

validation_generator = test_datagen.flow_from_directory(
    'DATASET/TEST',
    target_size=(150, 150),
    batch_size=batch_size,
    class_mode='binary')
```

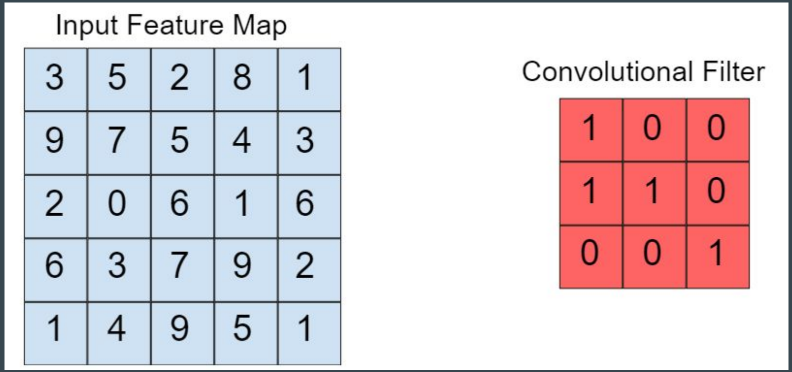
Generating batches of images using the ***flow_from_directory*** method provided by *keras* using the ***train_datagen*** and ***test_datagen*** object created previously. The images are resized to 150x150 and binary classes are assigned to them.

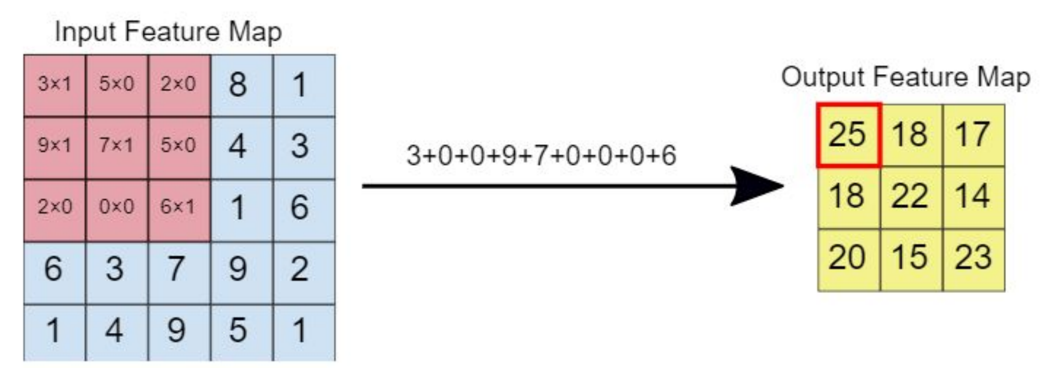
Building Convolutional Neural Network

- Convolutional Layers
- Dense Layers

Working of Convolutional layer

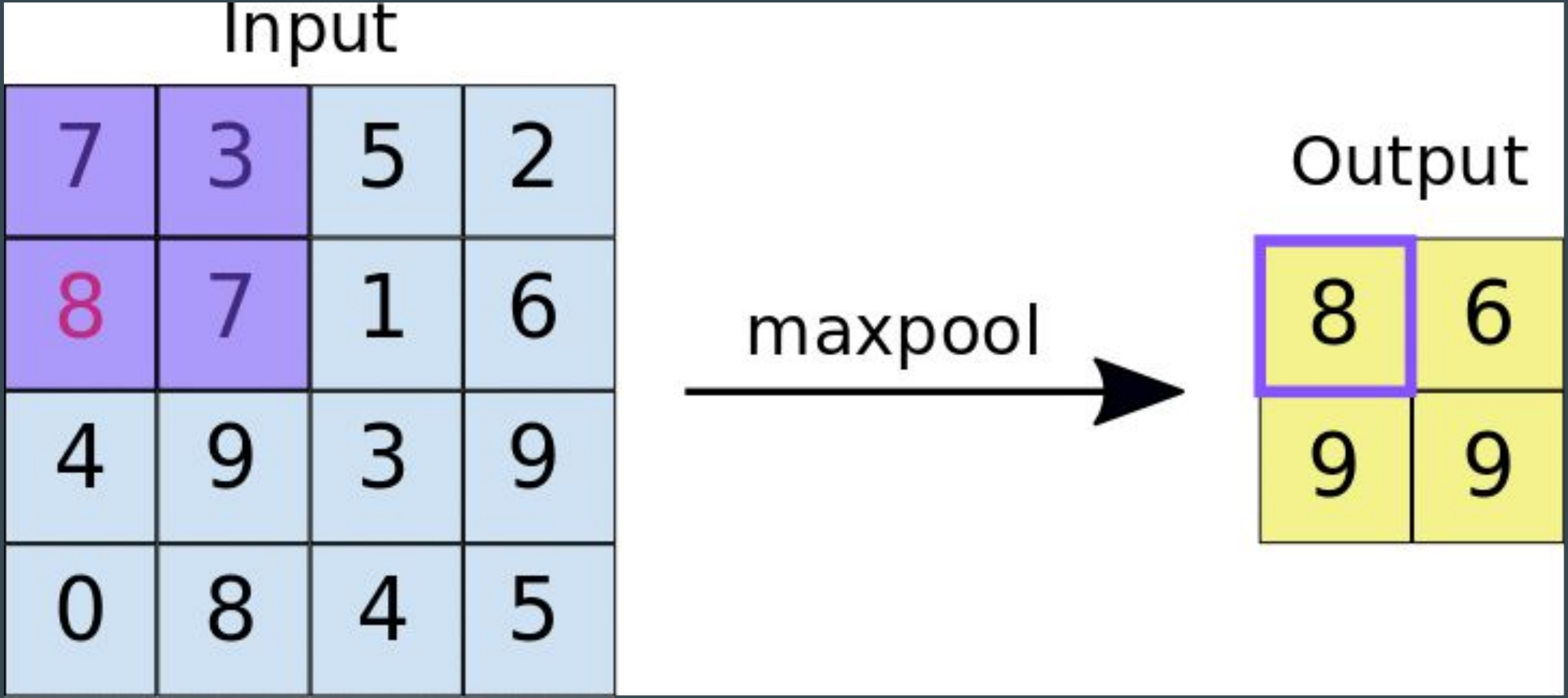




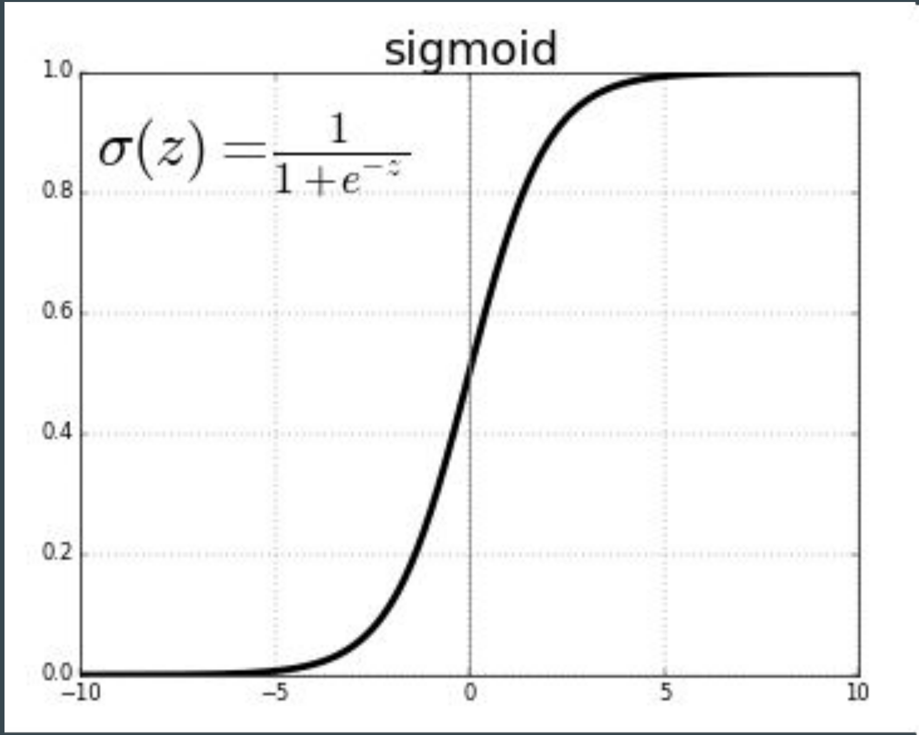


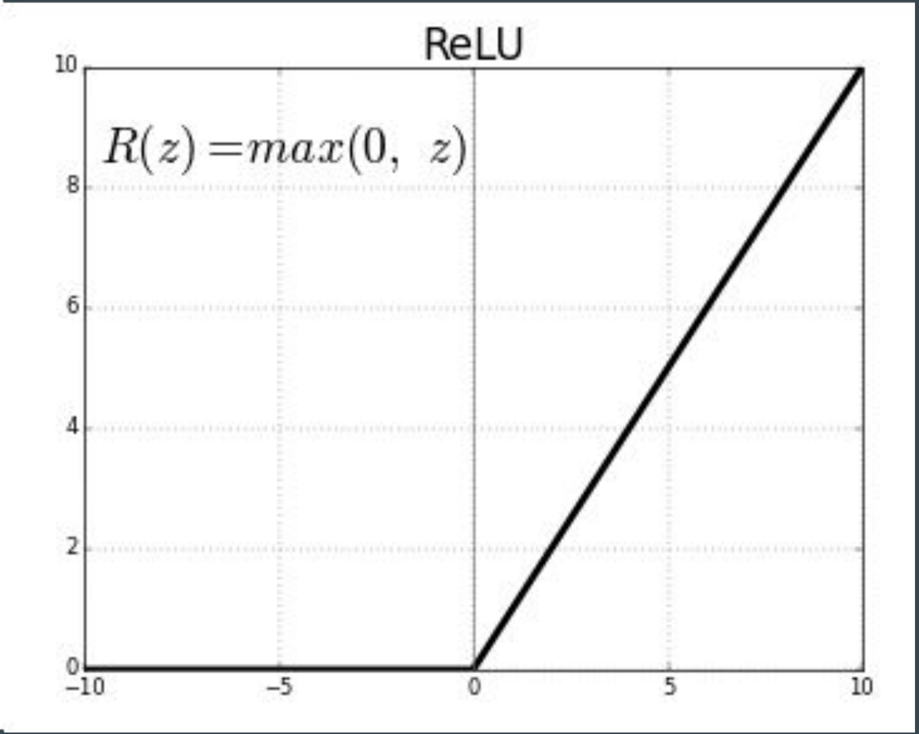
Convolutional Kernel Example (Edge Detection)

Working of Max Pooling Layer



Activation Functions





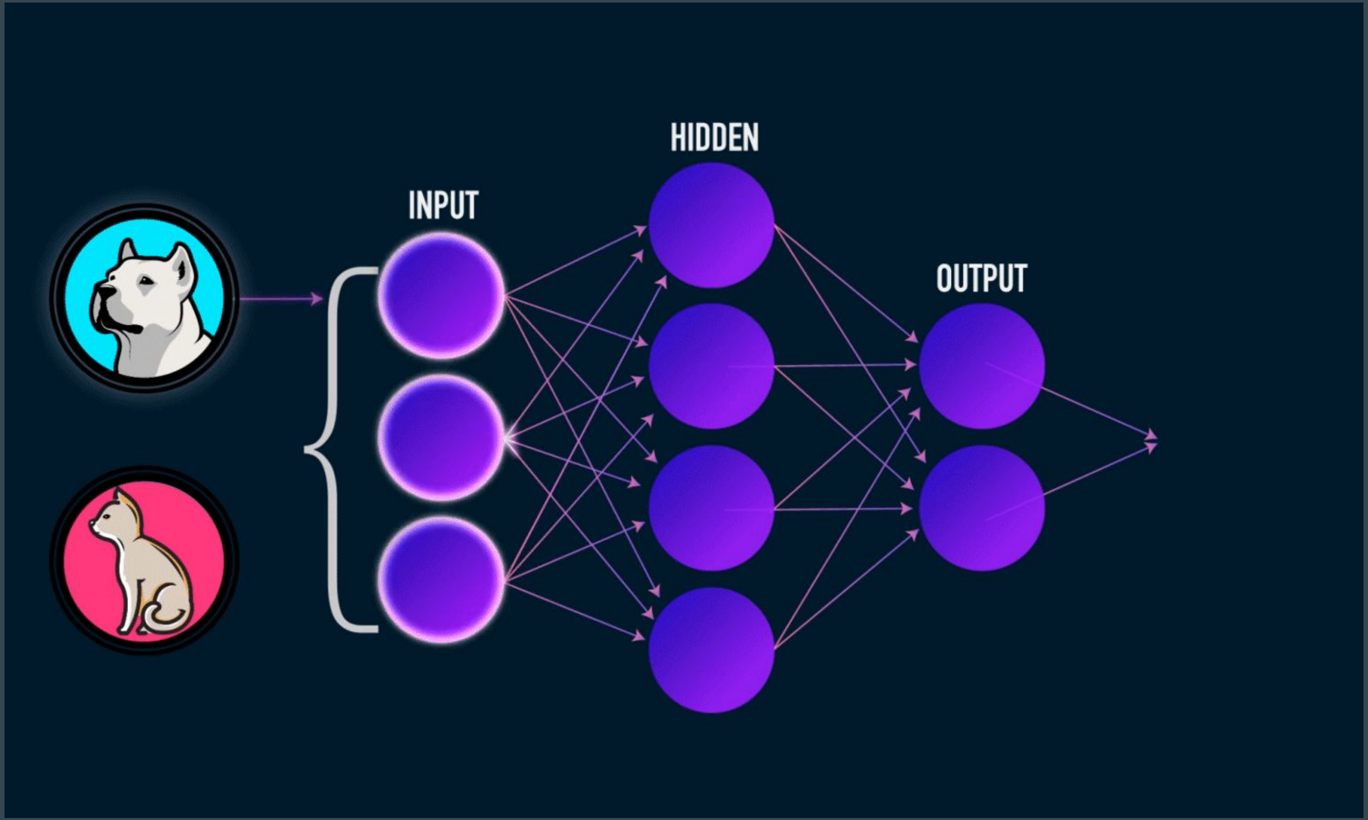
Convolutional layers

Convolutional Layers are great at extracting features from images.

- ***Conv2D*** **:** creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs.
- ***Maxpooling2D*** : Downsamples the input representation by taking the maximum value over the window
- ***Activation*** : *ReLU* (Rectified Linear Unit) activation function is used to activate the convolutional layers

```
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Conv2D(32, (3, 3), input_shape=(150, 150, 3)))
model.add(tf.keras.layers.Activation('relu'))
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))

model.add(tf.keras.layers.Conv2D(32, (3, 3)))
model.add(tf.keras.layers.Activation('relu'))
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))

model.add(tf.keras.layers.Conv2D(64, (3, 3)))
model.add(tf.keras.layers.Activation('relu'))
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
```

Working of Dense Layers



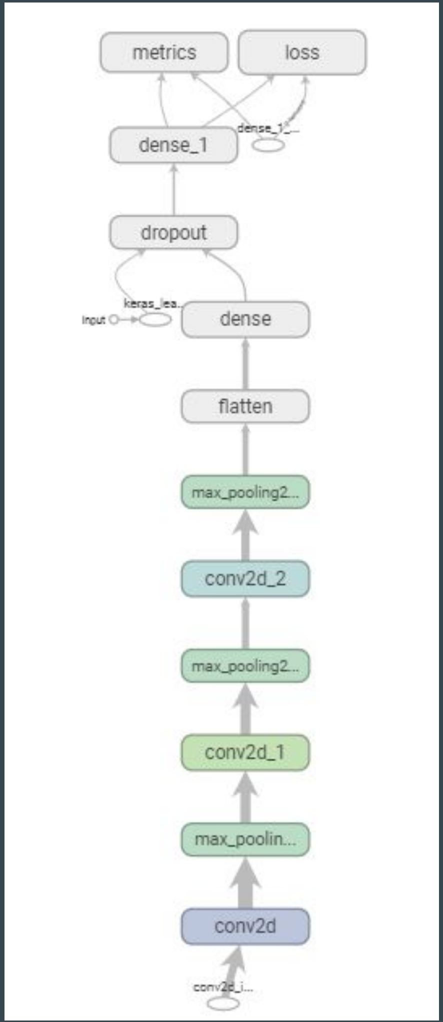
Dense layers

The convolutional layers are good for extracting features from images but they cannot recognising objects. For this traditional neural networks works best. So, dense layers are used after convolutional layers in a CNN.

- ***Flatten*** **:** flattens the 3D matrix from the convolution layers to 1D matrix.
- ***Dense*** : fully connected layers
- ***Dropout*** : reduce overfitting
- ***Activation*** : *ReLU* activation is used for dense layers. *Sigmoid* activation is used for the output layer.

```
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(64))
model.add(tf.keras.layers.Activation('relu'))
model.add(tf.keras.layers.Dropout(0.5))
model.add(tf.keras.layers.Dense(1))
model.add(tf.keras.layers.Activation('sigmoid'))
```

Structure of the Neural Network (Top-Down)



Compiling the model

```
model.compile(loss='binary_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

Configuring the model with loss and metrics for training. The weights of neurons are randomly initialized.

- **Loss Function** : binary cross entropy, as the number of outputs is two.
- **Optimizer Function** : RMSprop, task is not complex enough

Training model and saving layer and weights

```
model.fit(
    train_generator,
    batch_size=2000,
    steps_per_epoch=125,
    epochs=50)
model.save('waste-classifier.h5')
```

fit method is used to train the model on the training set. The model is trained for 50 epochs.

The model is saved for future use.

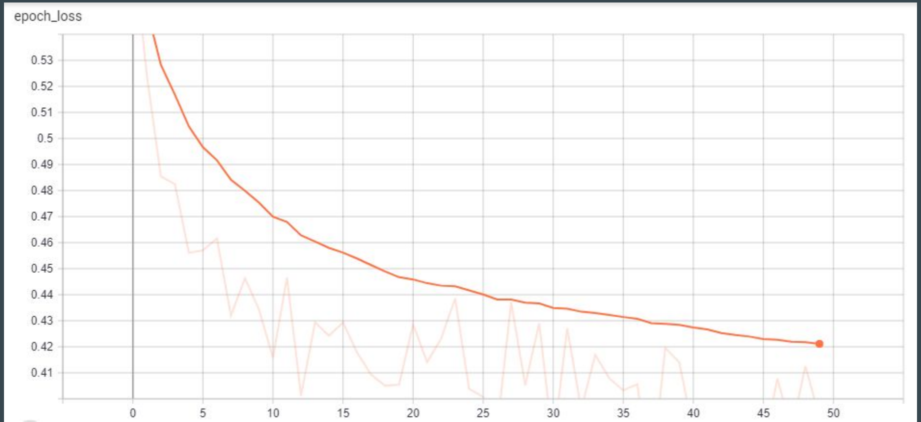
Evaluating our model

```
1 model.evaluate(validation_generator,steps=125)
```

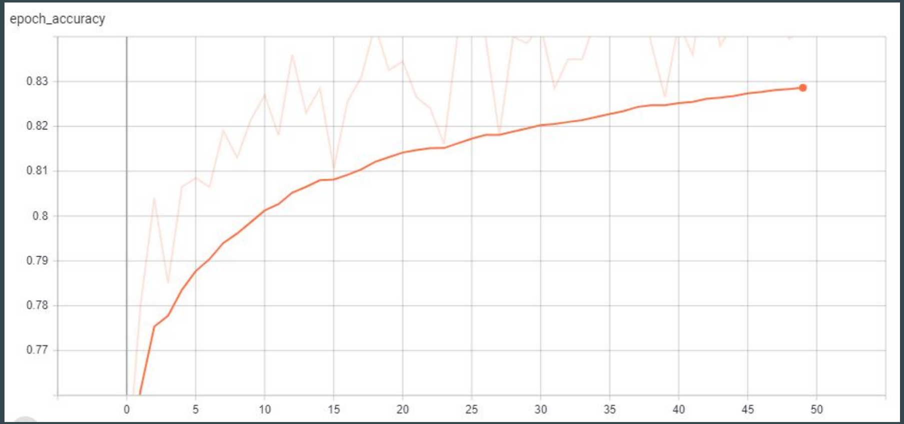
```
125/125 [==============================] - 3s 24ms/step - loss: 0.4069 - accuracy: 0.8800
[0.4068816304206848, 0.8799999952316284]
```

evaluate method passes all the test images through the model created and compares the predicted label with actual known label of the image and then calculates the accuracy of the model. (*about 88% accuracy*)

Loss and Accuracy graph during training

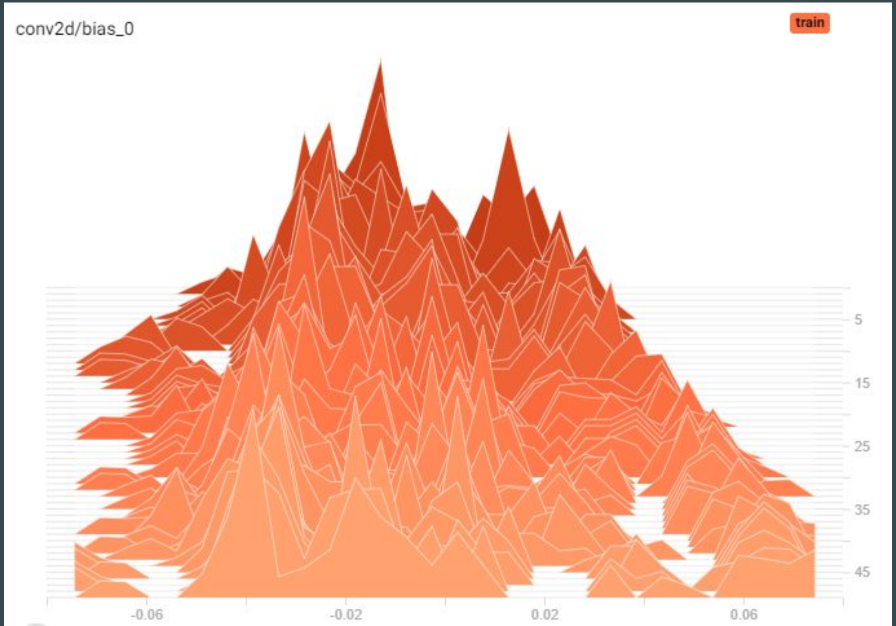


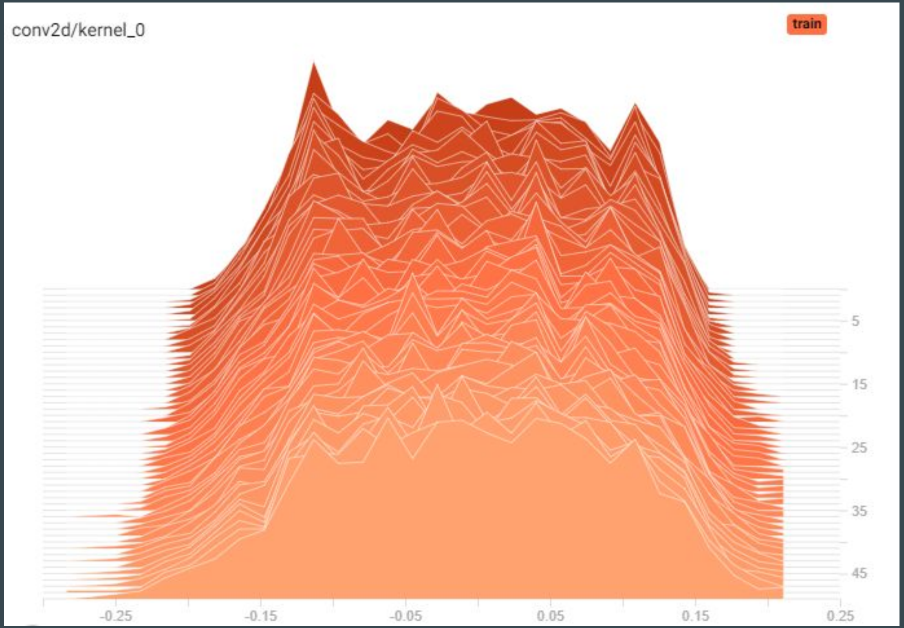
Loss at each epoch



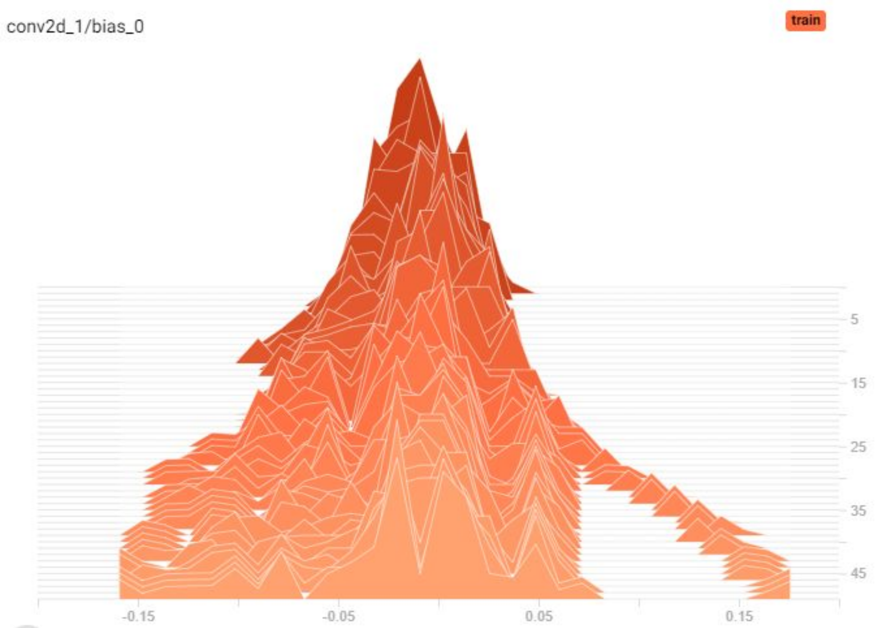
Accuracy at each epoch

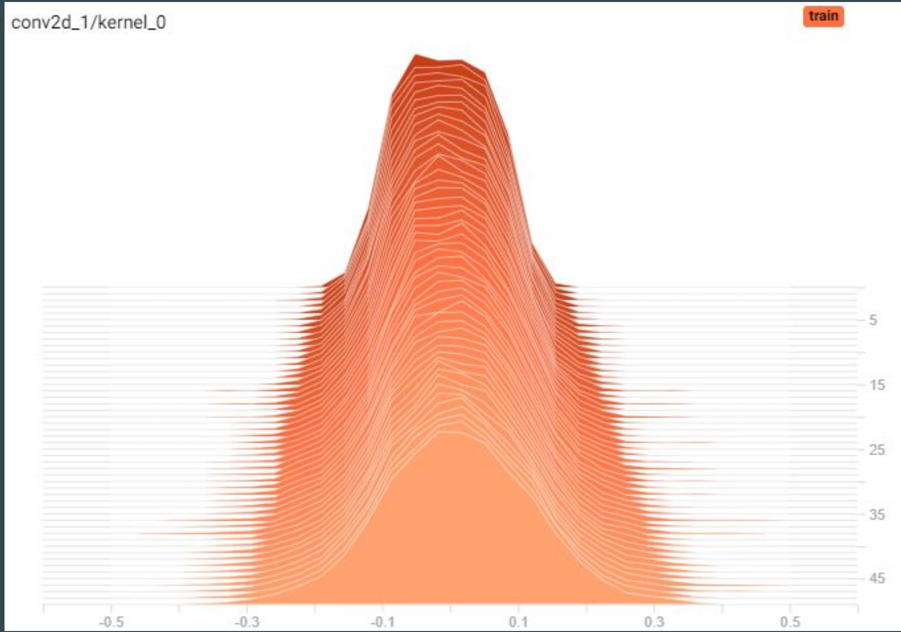
Input Convolutional Layer Histogram



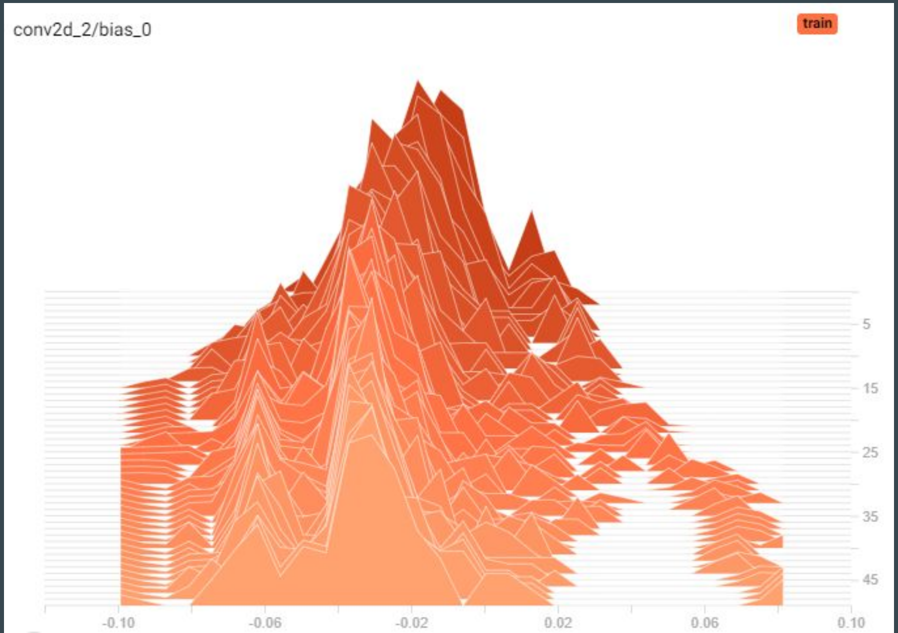


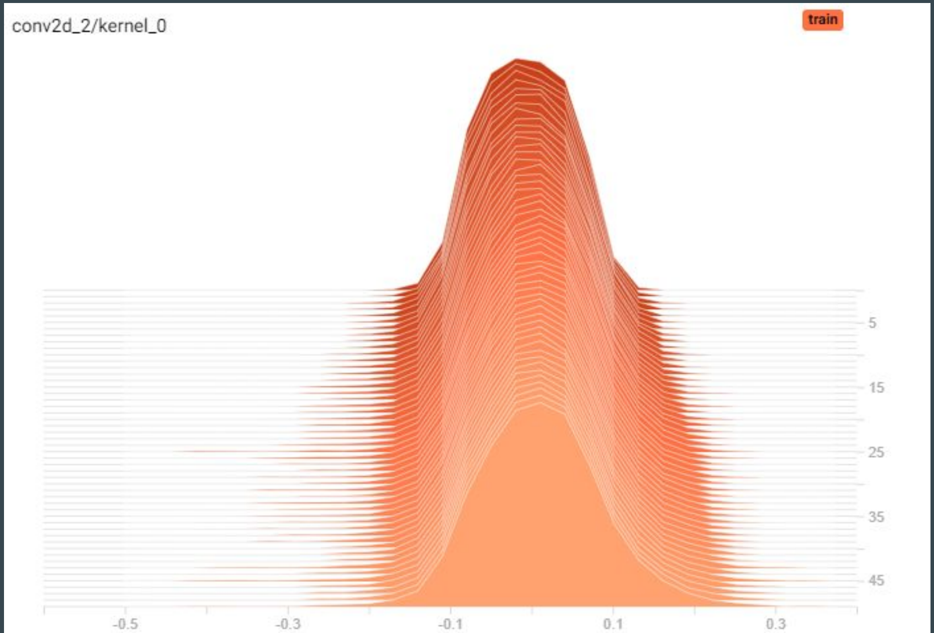
Hidden Convolutional Layer #1 Histogram



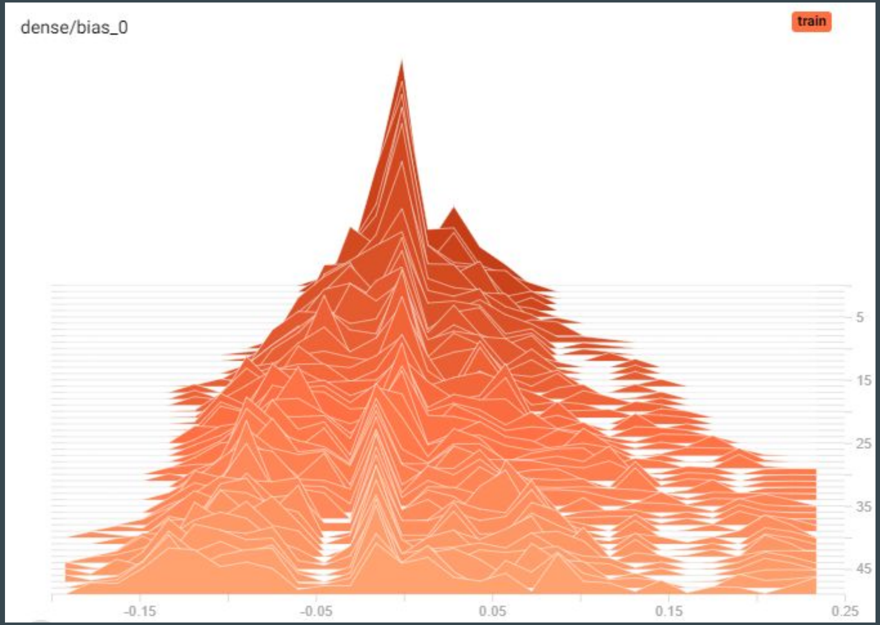


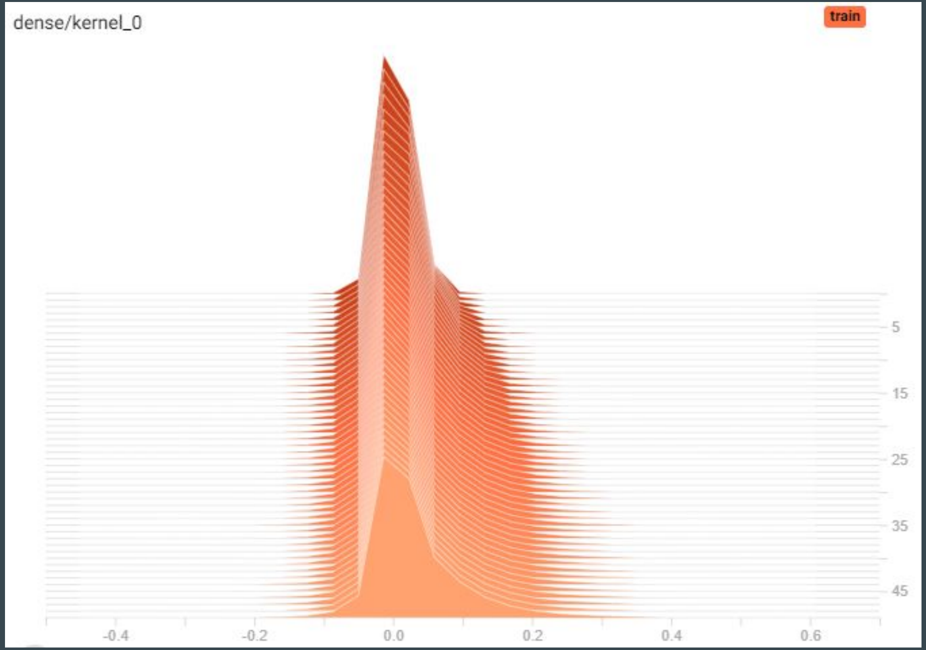
Hidden Convolutional Layer #2 Histogram



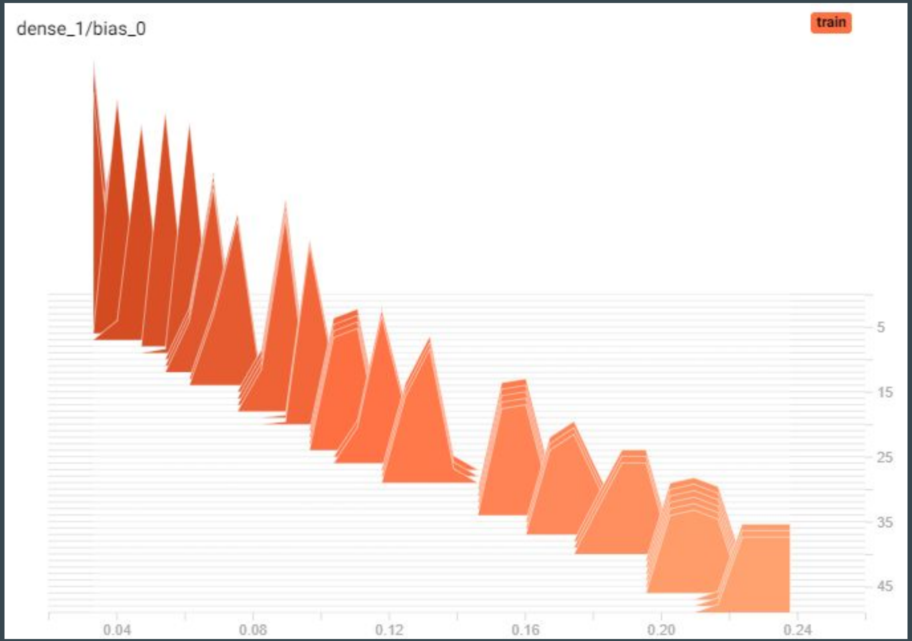


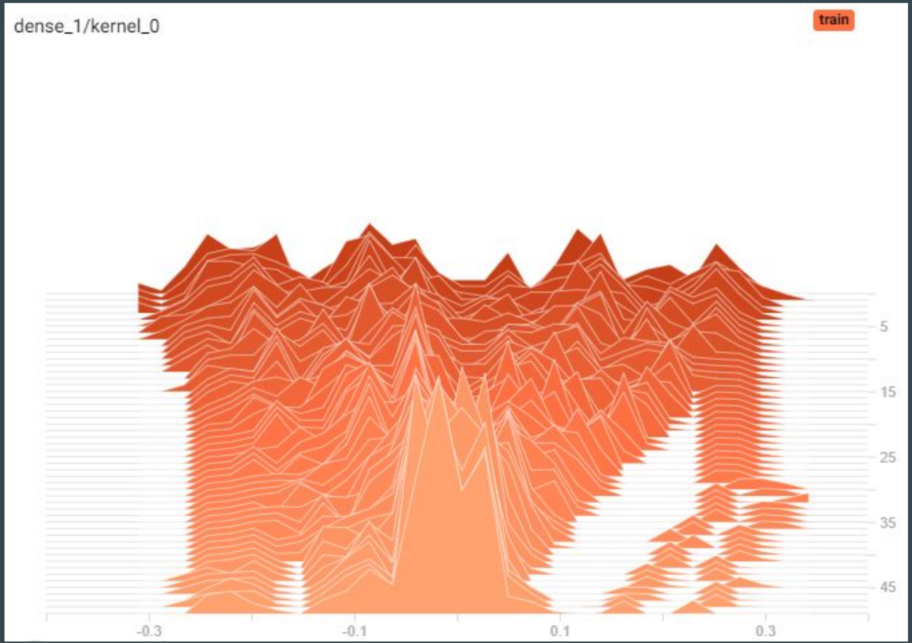
Hidden Dense Layer Histogram





Output Dense Layer Histogram

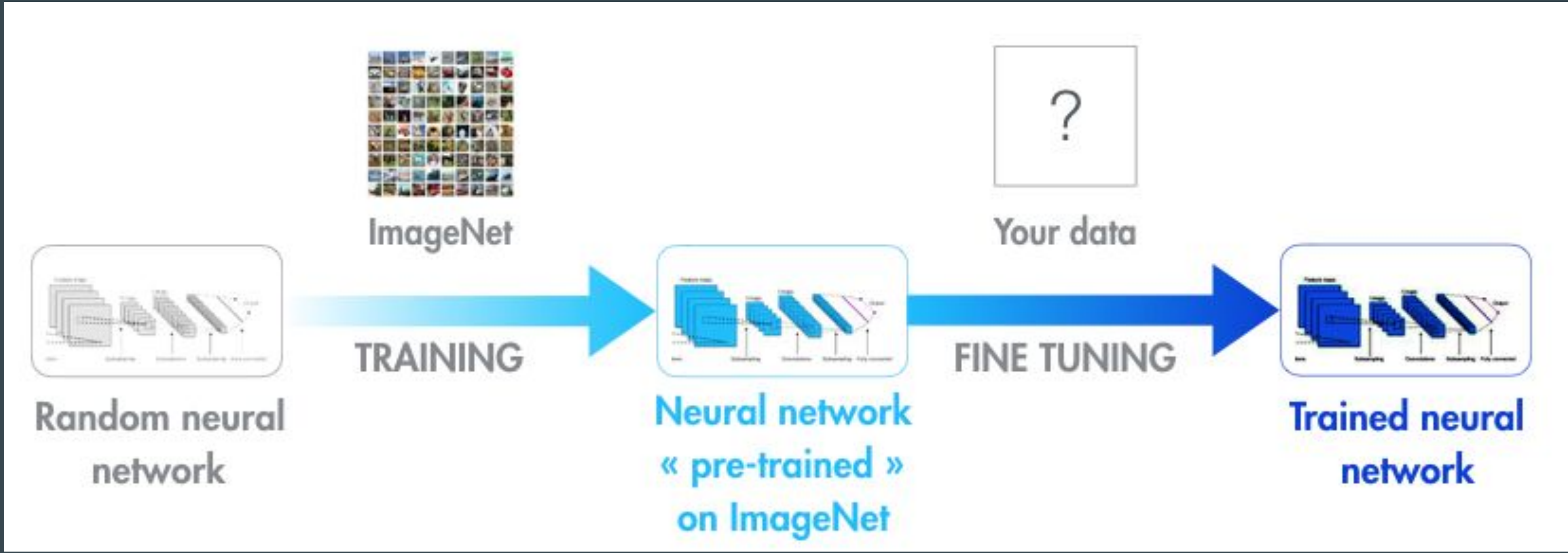




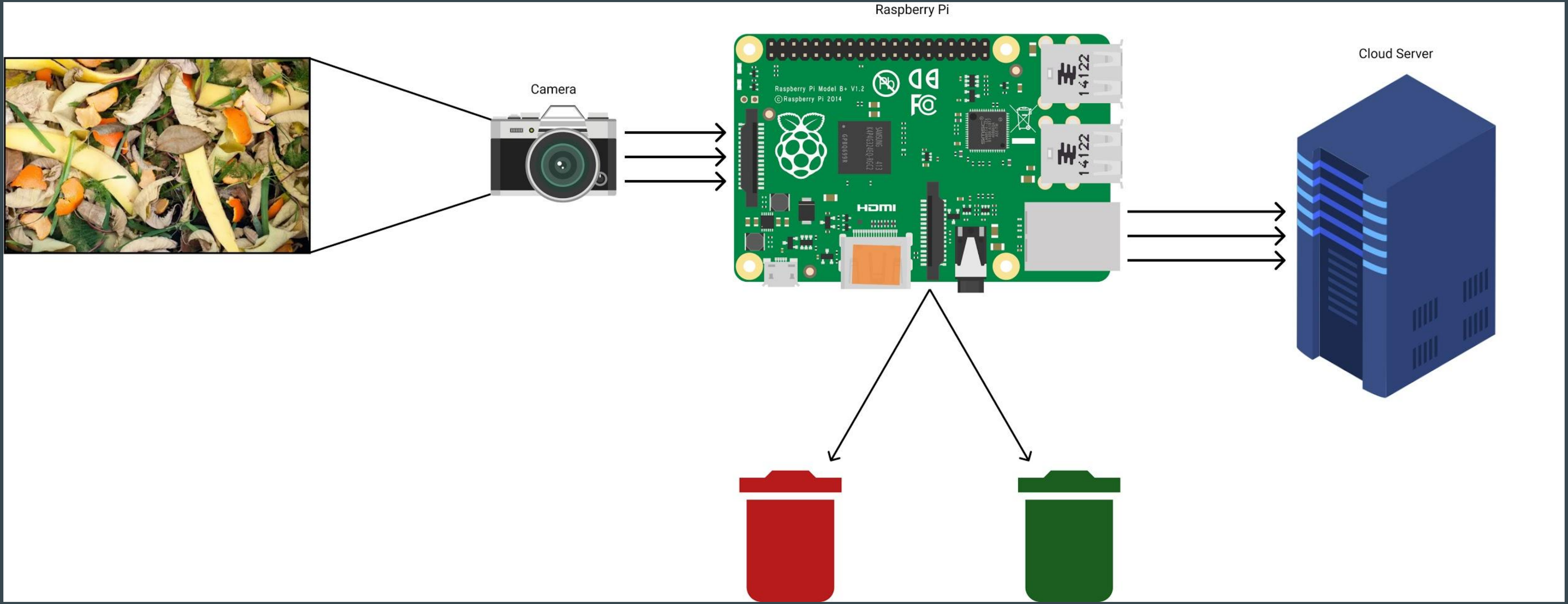
For the next Review!

1) Implement a model to classify images using transfer learning.
2) Create a IoT hardware that will run the classifier.

Transfer Learning



Hardware and software integration



📑References

IEEE PAPERS -

- Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.
- N. Jmour, S. Zayen and A. Abdelkrim, "Convolutional neural networks for image classification," 2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET), Hammamet, 2018, pp. 397-402, doi: 10.1109/ASET.2018.8379889.
- T. Guo, J. Dong, H. Li and Y. Gao, "Simple convolutional neural network on image classification," 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), Beijing, 2017, pp. 721-724, doi: 10.1109/ICBDA.2017.8078730.

🌍 Additional References

URL's -

- https://www.tensorflow.org/tutorials/images/cnn
- https://www.tensorflow.org/tutorials/images/classification
- https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html
- https://www.tensorflow.org/tensorboard/get_started
- https://www.kaggle.com/techsash/waste-classification-data
- https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks

🙏Thank You🙏

#staysafe