Project Architecture

Gems Sorting Machine

Made By:

Utkarsh Gupta - R177219194
Vanshaj Goel - R177219196
Tanya Malhotra - R177219191

Submitted to:

Dr. Kiran Kumar Ravulakollu

GitHub Repository:

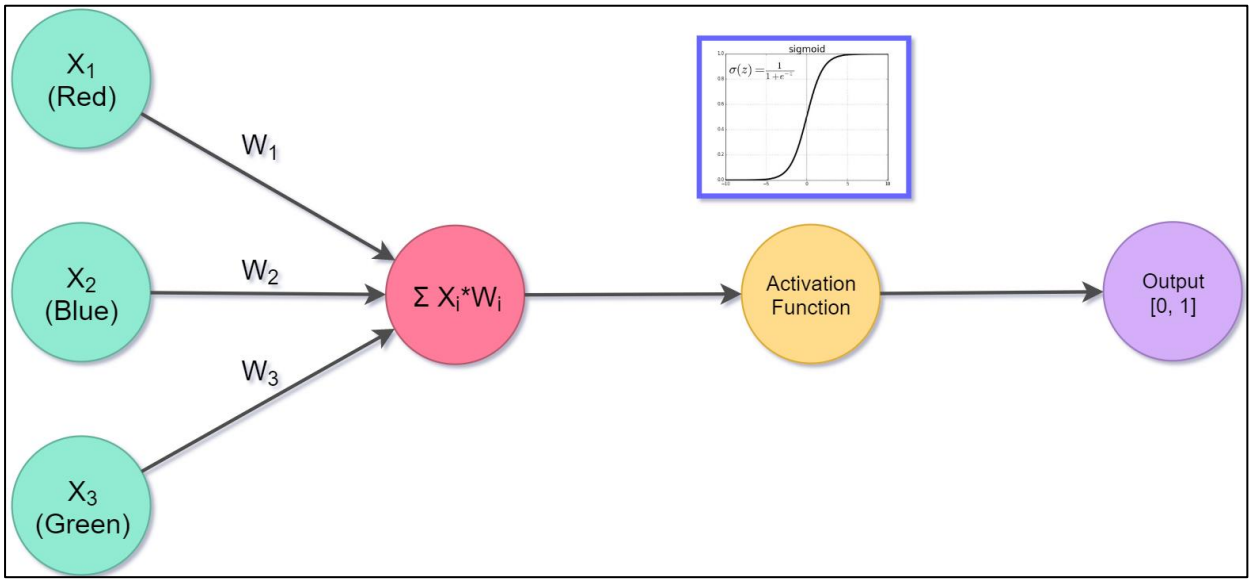
github.com/AvGeekGupta/Gems-Seperator.git

Website:

avgeekgupta.github.io/gems-seperator/

About Model

Neuron



- Type – Feed Back Neuron.
- Input – Three inputs, each for a colour code.
- Synaptic Weights – Three, Each for an input.
- Bias – None.
- Activation Function – Sigmoid function.
- Output – Binary Output [0, 1].
- Percent Error = $\frac{Expected\ Output - Output}{Output} \times 100$
- Correction –> $SynapticWeights = SynapticWeights + \frac{Expected\ Output - Output}{Output}$

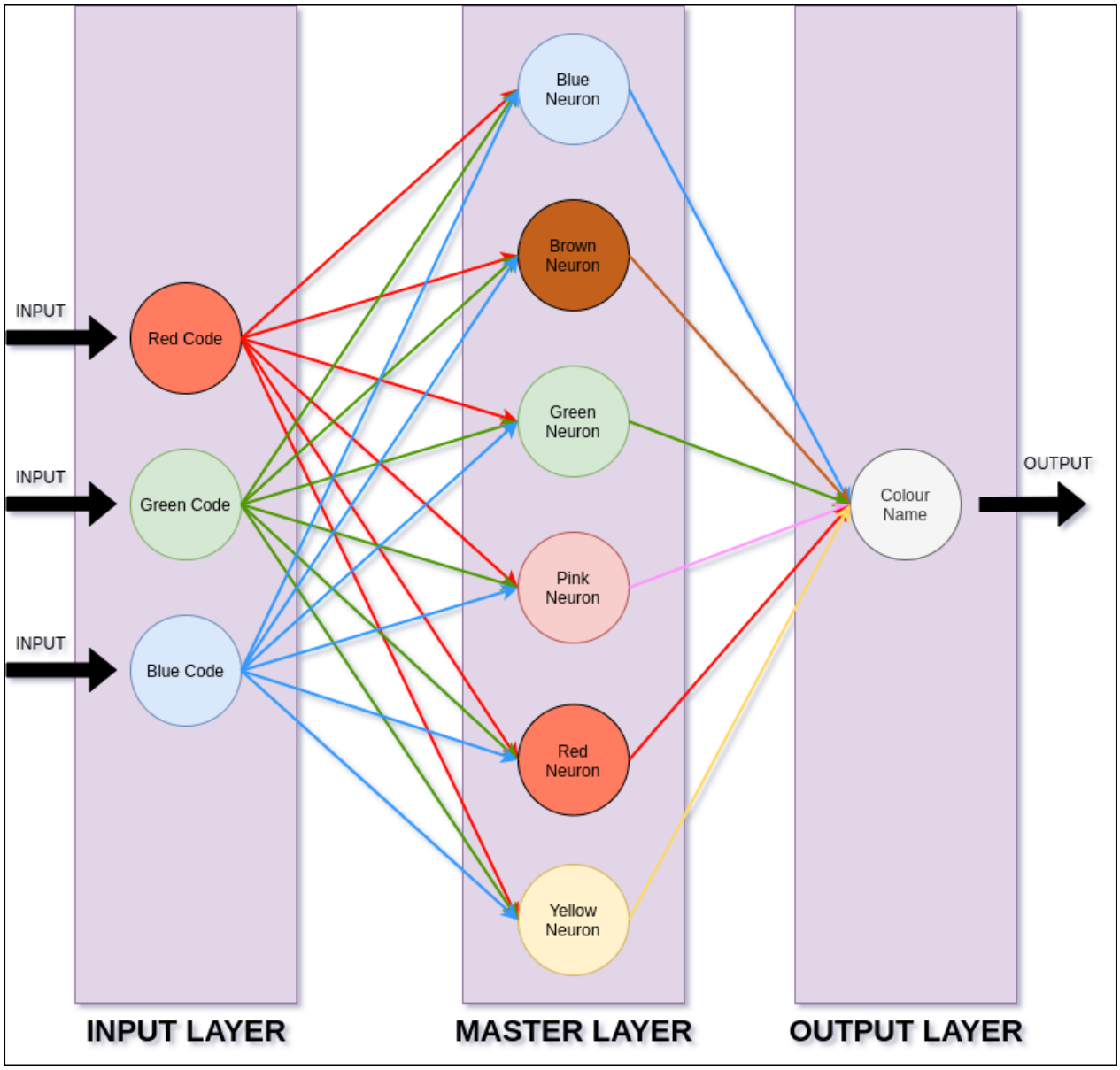
Construction

Neuron is constructed in a class named “Neuron” in file named “Neuron.py”. The class contains an “__init__” function to initialize synaptic weights.

“train” function is used to train the neural network and improving synaptic weights after every iteration.

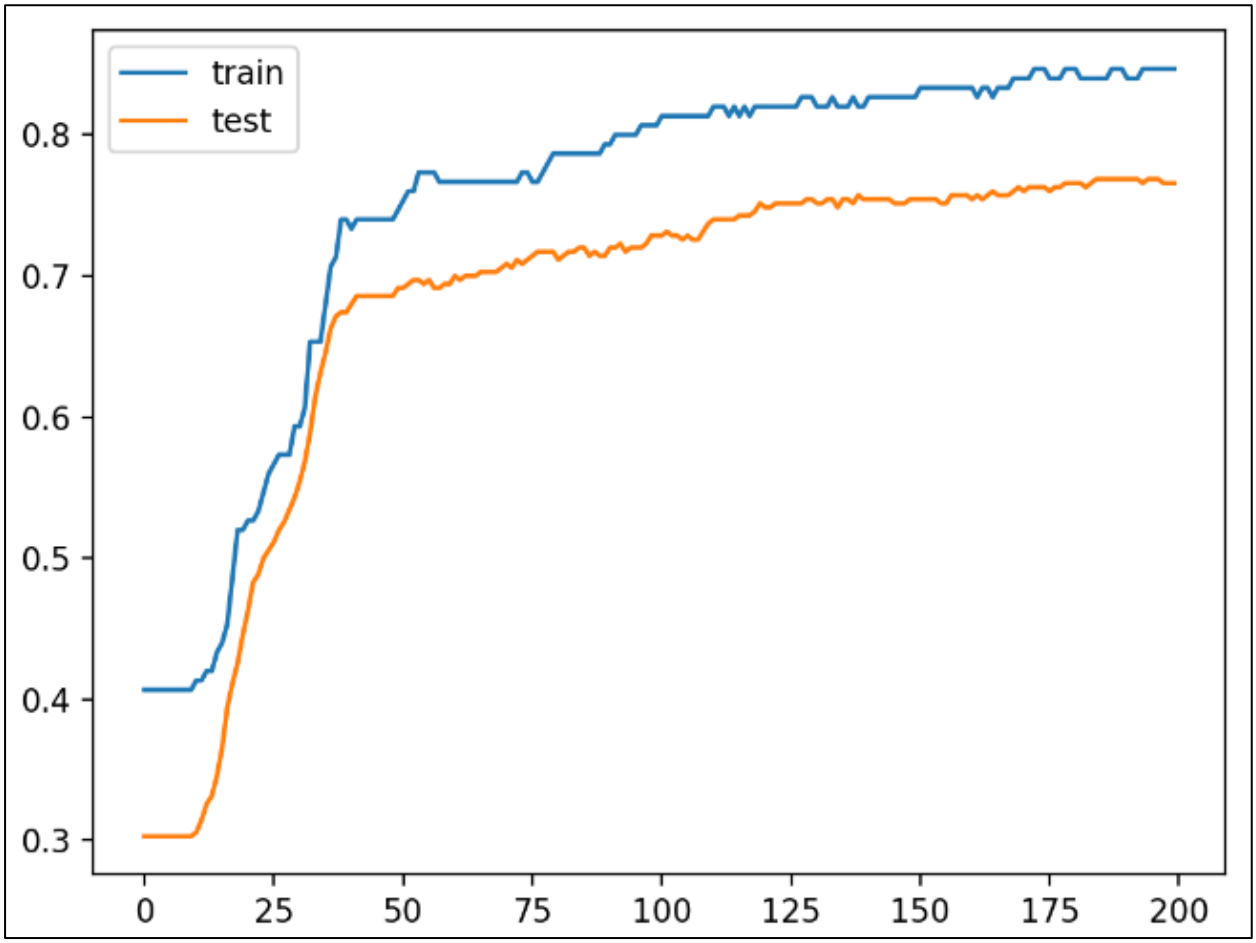
“think” function is used to determine whether the given input colour code in RGB format is the colour for which the neuron is trained or not. It gives out binary output [0, 1].

Neural Network



- Type – Feedback neural network.
- Layers – Three (Input, Master, Output).
- Input – Colour Code in RGB Format.
- Bias – None.
- Output – Colour name [Blue, Brown, Green, Pink, Red, Yellow]
- No. of Epochs for training – 200.
- Training accuracy – 83.3% (approximately).
- Testing accuracy – 75% (approximately).

Training effeciency graph



- No. of Epochs for training – 200.
- Training accuracy – 83.3% (approximately).
- Testing accuracy – 75% (approximately).

Construction

Neural Network is constructed in a class named "NeuralNetwork" in file named "Neural_Network.py". The class contains an "__init__" function to create six neurons and fetch the Synaptic Weights from the database to feed it to neurons.

"train" function to train each neuron as per the input of colour codes and expected output.

"think" function to get the output in [Blue, Brown, Green, Pink, Red, Yellow] for a given colour code.

"write_synaptic_weights" function when called it writes the synaptic weights of respective neurons in the database. It is called at the end of training the model to update the synaptic weights in the database.

About Database Collection

- Software used – Oracle's MySQL.
- Interface – MySQL Workbench
- Schema Name – Gems Separator
- No. of Databases – 4 (colour_data, run_results, synaptic_weights, user_details.)

"colour_data" Database

- This database contains six tables. Each table corresponding to each colour.
- Each table has four columns "s. no.", "red_component", "green_component", "blue_component".
- These tables are used to store the RGB colour codes of the gems with the serial number.
- All the tables have approximately 100+ rows each that was used for train and test the Neural Network model.

"run_results" Database

- This database contains two tables. One for storing the no. of sorted gems of each colour after every sorting named "sorting_result". Another to Store Errors after every epoch during training of the Neural Network named "errors".
- "sorting_result" table has nine columns "s.no.", "datetime", six for no. of gems of each colour and "total".
- "errors" table has two columns "S. no." and "percent_error". This table is used to construct the efficiency graph of the model.

"synaptic_weights" Database

- This database contains six tables. Each table corresponding to each colour.
- Each table has four columns "s. no.", "red_synaptic_weight", "green_ synaptic_weight", "blue_ synaptic_weight".
- These tables are used to store the synaptic weights of each neuron.
- The synaptic weights in the table is updated after every epoch while training.

Connection with Python Application

The python application connects to the MySQL schema through the “mysql.connecter” class.

The class “Database.py” contains a class named “Database”. The “__init__” function of the class connects to the database at MySQL server.

“get_synaptic_weights” function fetches the synaptic weights for each neuron from the database which is used to initialize the neurons in the neural network.

“write_synaptic_weights” function writes the synaptic weights for each neuron to the database after each epoch while training the Neural Network model.

“get_colour” function fetches the colour codes of the gem stored in the “colour_data” databse for training and testing of the Neural Network model.

“write_colour” function writes a new entry in the “colour_data” database so that it can be used later to train and test the Neural Network model.

“get-error” function fetches the error from the database that was used to construct the efficiency graph for the Neural Network model.

“write_error” function writes the error to the database after every epoch while training the Neural Network Model.

Image Capturing & Colour Extraction

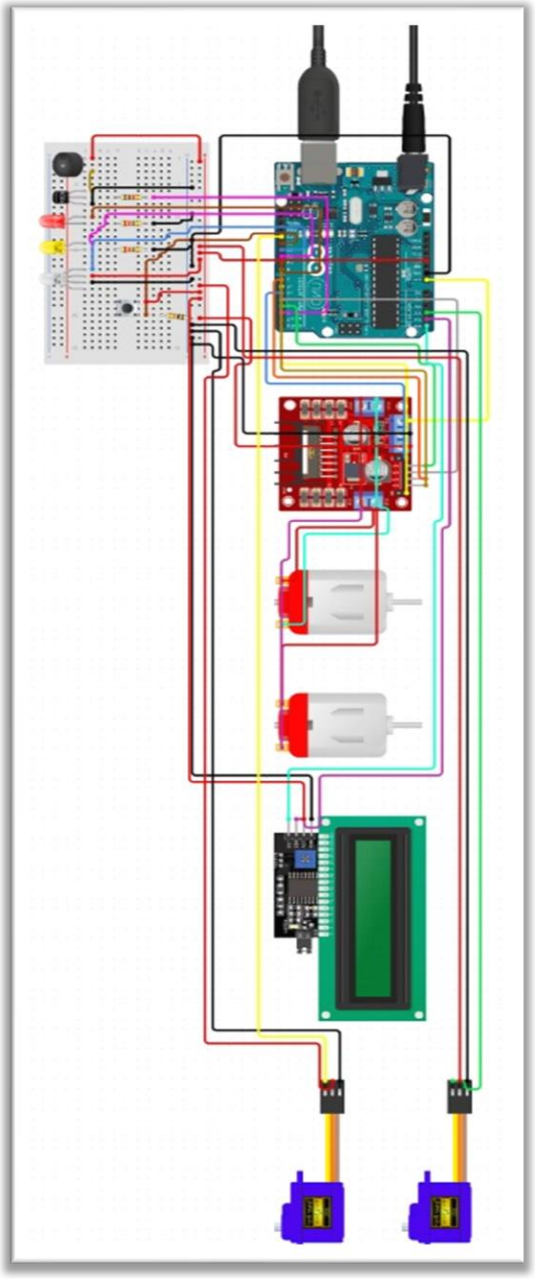
OpenCV library is used for this purpose. The method name “click_image” is defined in the class “ImageCapture” in the file named “Image_Capture.py”.

The method includes following steps –>

1. Clicking of picture through webcam.
2. Crops down the image to a smaller size for easy processing.
3. Saving Image in “Images” folder with the name depending on date and time.
4. Taking out the colour which has most occurred in the picture.
5. Returning the RGB colour code as a numpy array

About Hardware

- Software Used – Arduino IDE.
- Board – Genuino/Arduino Uno R3.
- Programming Language – C/C++.
- Major parts – LCD Screen, I2C Interface Module, L293D Motor Driver, Motors, Servo, LED's, Buzzer.



Working of Hardware

Arduino board works in an infinite loop in the following steps –>

1. Reading Input from serial port.
2. Identifying the instructions to follow through switch case.
3. Printing the colour on the LCD Screen.
4. Lighting up the RGB LED to the correct colour.
5. Turning the dropper towards correct box.
6. Turning motors on to drop the gem.
7. Pushing next gem in front of camera.
8. Go to step 1.

Connection with Python Application

The hardware and the Python Application through "SerialCommunication" class defined in "Serial_Communication.py". The python library used for this is "serialcommunication".

"__init__" function establishes the connection with the Arduino board.

"print_colour" function provides the colour to the Arduino board so that the hardware drops the gem in the specified box.

Datasets used

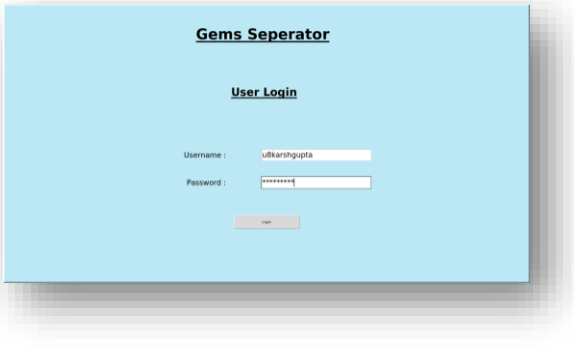
Datasets used for training the Neural Network model are generated through this application by clicking pictures of the gems and extracting colours. The data is stored in MySQL database first and then exported to JSON files later. The JSON files are stored in Folder “Datasets”. The folder contains six JSON files. Each file has 100+ rows of data for each colour that were used for training and testing of the Neural Network model.

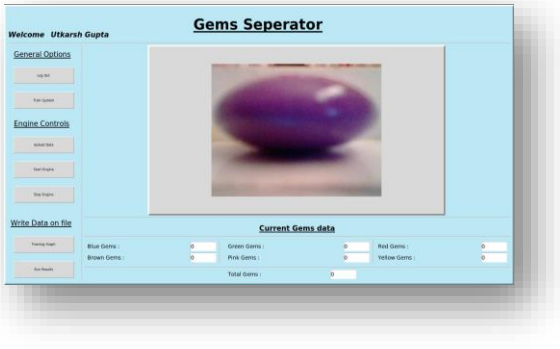
The actual images from which the colours are extracted are also saved in the folder named “Images”.

User Interface

- Software used – PAGE.
- Framework used – Tkinter.
- No. of GUI files – Three (Main Page, Login Page, Data Collection Page).

All the objects and the functioning of every button is defined in of the GUI are defined in the respective classes of their respective files.





The support files provides the necessare objects and functions that required to run these GUIs.