

A Novel Approach Towards Calculating the Reusability Coefficient of Water From Nearby Water Bodies Using Artificial Intelligence

Authors: Ahana Jhamb¹; Vikramjeet Singh²; Reetu Jain³

Singapore American School, 40 Woodlands Street 41 Singapore 738547¹; On My Own Technology Pvt. Ltd. Mumbai MH^{2,3}

Abstract

Water is one of the most important elements and it is susceptible to pollution. Due to this, most of the water bodies around us are polluted now. Some of them are highly polluted, whereas some are moderately polluted. But there is no data available on the amount of pollution that is present in any specific water body. If we can get information regarding the quality of water in the body of water, then the water from that body can be used after some filtering processes depending on the quality of water. So we have made a solution where we have collected water samples from various bodies and places like sewage water, river water, farm water, Our solution has 3 parts. The first is detecting the turbidity of the water using image processing; the second is checking the TDS and pH of the water using the prototype that we have made using Arduino; and then finding out the quality coefficient using machine learning. We have made a 3D printed case to put all the sensors and microcontroller together in one device. Our model works with an accuracy of 89%. Whereas the circuit works perfectly with high accuracy in the detection of TDS and pH. To conclude, we can say that our solution is a real-time and fast solution to detect the quality of the nearby water body and it can also put that water to use if that can be used.

Keywords: water quality detection, water bodies, turbidity detection, pH, and TDS.

Introduction

When it comes to sustaining life, water is one of the most crucial components on Earth. Sadly, it is also incredibly vulnerable to pollution. This is mostly because water is a universal solvent that can dissolve a wide variety of compounds. This is a beautiful property that we utilize for routine activities like cooking, cleaning, and taking medications, but it is also the property that makes water so easily contaminated.

The reasons for water contamination are numerous. We'll concentrate on seven of the main ways that water might be contaminated below.

1. Industrial Waste

Worldwide, industries and industrial facilities have a significant role in water contamination. Although many industrial sites are controlled, others still lack adequate waste management systems and produce waste in the form of harmful chemicals and contaminants. Industrial waste is occasionally thrown into surrounding freshwater systems. Industrial waste has a very high potential to contaminate freshwater systems when improperly handled (or worse, not handled at all).



Industrial waste from mining operations, manufacturing facilities, and agricultural sites can enter rivers, streams, and other waterways that flow directly into the ocean. In addition to possibly making water unfit for human consumption, the hazardous compounds in the waste produced by these enterprises also could alter the temperature of freshwater systems, endangering a variety of aquatic creatures.

2. Sewage and Wastewater

Even after treatment, harmful chemicals, germs, and pathogens can still be detected in sewage and wastewater. Each residence releases effluent and sewage into the sea along with fresh water. The germs and bacteria in that wastewater spread disease, resulting in problems with both human and animal health.



3. Marine Dumping

Marine dumping is exactly what it sounds like—it involves throwing trash into the ocean. Even though it may sound absurd, several nations still collect domestic waste and dump it into oceans. Most of these goods can take 2 to 200 years to totally disintegrate.



4. Oil Leaks and Spills

When two substances do not mix well or at all, the idiom "like water and oil" is often used. Contrary to what is said in the proverb, oil does not dissolve in water. Despite being frequently unintentional, large oil spills and oil leaks are a major contributor to water contamination. Oil drilling operations in the ocean or ships that deliver oil to wildlife frequently result in leaks and accidents.



5. Global Warming

In terms of water contamination, rising temperatures brought on by global warming are a serious worry. Water-dwelling creatures may die because of rising water temperatures brought on by global warming. Large die-offs exacerbate the problem by polluting the water source further.

You may contribute to lowering water pollution and global warming in a variety of simple ways every day. These strategies include recycling, riding in a carpool, and utilizing CFL lights at home.

6. Agriculture

Farmers frequently employ chemicals and pesticides to protect their crops from bacteria and insects. These pollutants can be harmful to people, plants, animals, and the environment when they seep into groundwater. In addition, the chemicals combine with precipitation when it rains, which then flows into rivers and streams and eventually the ocean, increasing water contamination.



7. Radioactive Waste

It is important to properly dispose of radioactive waste from nuclear energy production facilities since it can be very dangerous to the environment. This is due to the very poisonous nature of uranium, the metal used to produce nuclear energy.

Unfortunately, mishaps still happen at these locations, releasing harmful waste into the ecosystem. In many ways, the coal and gas companies are no better. One of the main driving forces behind the creation of alternative, clean energy sources like solar and wind is this.

Literature Review:

1. Evaluation of water quality based on a machine learning algorithm and water quality index:

This study creates a model for estimating and evaluating the WQI by combining a machine learning algorithm, the WQI, and remote sensing spectral indices (difference index, DI; ratio index, RI; and normalized difference index, NDI). The findings demonstrate that the computed WQI

values fall within the range of 56.61 to 2,886.51. We also investigate the connection between reflectance information and the WQI. Following a high after 1.6 orders, the number of bands with correlation coefficients passing a significance test at 0.01 first rises and then falls. After 1.6 orders, WQI, DI, RI, and NDI correlation coefficients between the peak's ideal band combinations also arise, with R2 values of 0.92, 0.58, and 0.92. To examine the prediction power of these models, POS-SVR created 22 WQI estimation models. With an R2 of 0.92, an RMSE of 58.4, an RPD of 2.81, and a slope of curve fitting of 0.97, it was discovered that models based on a spectral index of 1.6 performed significantly better than the others.

2. Water Quality Prediction Using Artificial Intelligence Algorithms:

In this work, cutting-edge AI algorithms are created to forecast the water quality index (WQI) and water quality classification (WQC). Artificial neural network models, specifically the long short-term memory (LSTM) deep learning algorithm and nonlinear autoregressive neural network (NARNET), have been created for the WQI prediction. Three machine learning techniques have also been utilized for the WQC forecasting: support vector machine (SVM), k-nearest neighbor (K-NN), and Naive Bayes. The created models were assessed based on several statistical criteria, and the employed dataset has 7 significant parameters. The findings showed that the suggested models can identify the water quality according to superior robustness and predict WQI with accuracy. According to prediction findings, the WQI values could be predicted more accurately by the NARNET model than by the LSTM, and the WQC values could be predicted with the maximum accuracy (97.01 percent) by the SVM method. Furthermore, with only a minor difference in the regression coefficient, the NARNET and LSTM models both achieved equivalent accuracy for the testing phase. The management of water resources can

benefit greatly from this kind of promising research.

3. **An optimized machine learning approach to water pollution variation monitoring with time-series Landsat images:**

In this article, we investigate a strategy for routine remote sensing monitoring on the DWSA of Shanghai's upper Huangpu River. First, the Mixed Kernel ELM with Particle Swarm Optimization (PSO-MK-ELM) Extreme Learning Machine (ELM) classification algorithm was developed. Four NPS contamination sources—farmland, building land, woodlands, and water—were correctly and effectively identified based on the PSO-MK-ELM. After that, 30 years' worth (1989-2019) of Landsat pictures were used to do their appropriate spatiotemporal analysis. The primary pollutants discharged into DWSA were also computed using the Common Export Coefficient Model based on NPS pollution source region and census data from 1989 to 2017. (ECM). Finally, the effects of NPS pollution sources' geographical and temporal changes on pollutant emissions were examined. Our findings are anticipated to provide a scientific foundation and data support for NPS pollution control and DWSA protection for better practices for environmental management in megacities around the world. The results show that the PSO-MK-ELM has an advantage in efficiency and accuracy in NPS pollution source extraction.

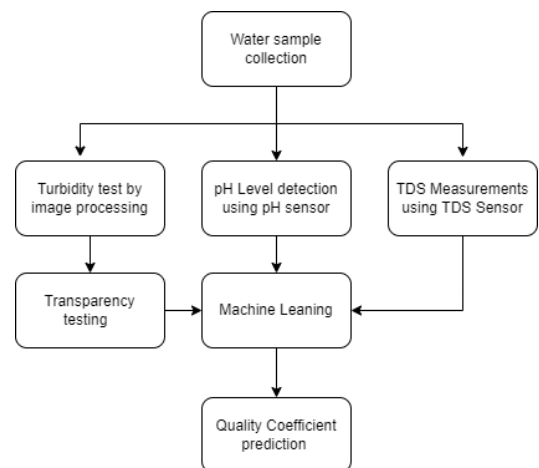
4. **Water Pollution Control in India:**

The main law in India for controlling water pollution is the Water (Prevention and Control of Pollution) Act of 1974 (the "Water Act"). To stop and control water pollution, the Water Act was created, which prohibited the discharge of contaminants into the water system in excess of permitted levels. It also calls for the creation of Central Pollution Control Boards (CPCBs) for the federal government and State Pollution Control Boards (SPCBs) for state governments. The Water Act, which was approved and continues to be the first environmental regulation in India, was

published before the Environment (Protection) Act, 1986. An amendment made in 1988 brought the provisions of the Water Act into line with those of the Environment (Protection) Act, 1986. The Indian Constitution gives the states the power to control water, and, no federal water legislation can be enacted. As a result, the central government's primary responsibilities are the development of an all-encompassing strategy for the management of water resources and state coordination. However, the Water Act was passed in accordance with Article 252 of the Constitution as an exceptional national water law. The Water Act's subsidiary rule, the Water (Prevention and Control of Pollution) Rules, was created in 1975. The Rules include a thorough overview of the CPCB's salary, authority, spending, and responsibilities.

Concept:

As we know there are multiple sources which contribute to the overall water pollution. So the concept is to detect the level of pollution and contamination in the water body by doing some tests and analysis using a smart device. That device will check the turbidity, TDS, pH, and the transparency of the water and then it will give a quality coefficient to the water so that it can be used according to its quality either after filtering or without it.



Methodology:

To make a reliable solution we did an experiment which have following steps:

1. Sample collection
2. Circuit making
3. Coding the circuit
4. Image processing
5. Machine learning
6. Result Prediction

Sample Collection:

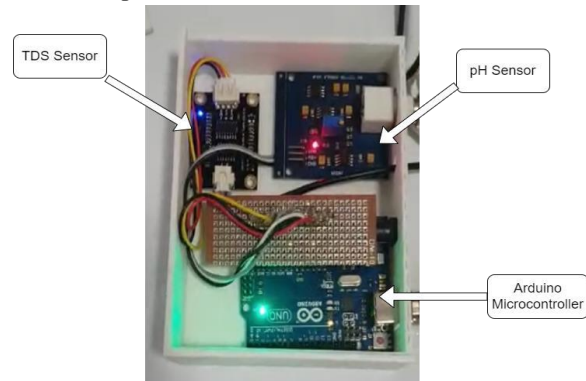
To begin with the experiment, I have gone to multiple places and collected various water samples which includes tap water, sewage water, organic waste water, gray water, river water, farming field water.



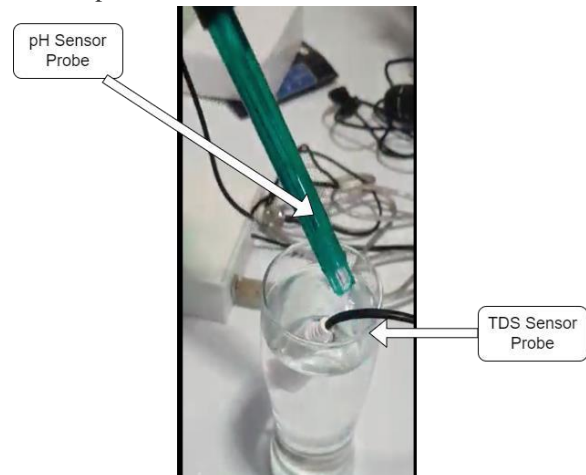
Prototype:

Our solution consists of two different parts, one is an electronic device that gets all the important information and parameters of the water using various sensors like pH, TDS sensor. And another is the image processing of the water that tells the visible features of the water.

In our prototype we have made a small box device which has the sensors attached to it that goes inside the water sample.

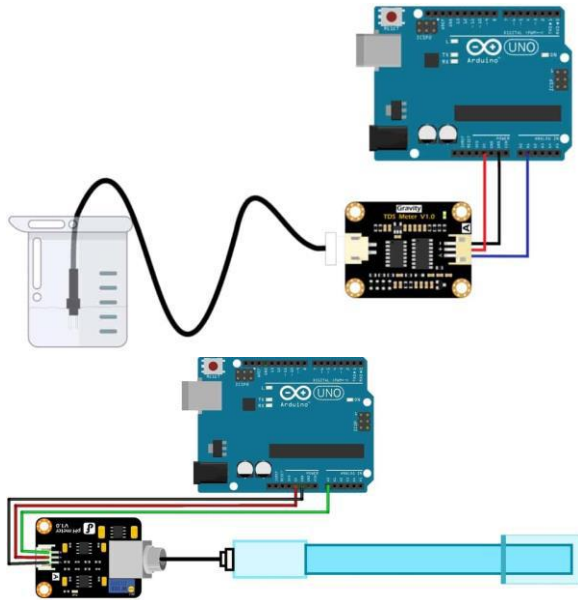


We have designed a special 3D Printed case to make it look like a proper and a portable device. It is a 3-inch X 2 inch box which holds all the electronic components inside it. Only the sensor probes go out in the sample water.



Circuit Diagram:

The circuit is really simple as both the sensors are analog sensors and in this circuit, we have connected pH sensor on the A0 pin and TDS Sensor on the pin A1. Arduino is connected to the computer serially via USB port.



Coding the circuit:

Circuit coding is done in C++ language on Arduino IDE. Using this code we are receiving the sensor data using analog pins and sending it serially via com port.

```
#include <EEPROM.h>
#include "GravityTDS.h"
```

```
#define TdsSensorPin A1
GravityTDS gravityTds;
```

```
float temperature = 25,tdsValue = 0;
```

```
float calibration_value = 21.34;
```

```
int phval = 0;
```

```
unsigned long int avgval;
```

```
int buffer_arr[10],temp;
```

```
//int ph_pin = A0; //This is the pin number connected to Po
```

```
void setup() {
```

```
    Serial.begin(9600);
```

```
    gravityTds.setPin(TdsSensorPin);
```

```
    gravityTds.setAref(5.0); //reference voltage on ADC, default 5.0V on Arduino UNO
```

```
    gravityTds.setAdcRange(1024); //1024 for 10bit ADC;4096 for 12bit ADC
```

```
    gravityTds.begin();
```

```
}
```

```
void loop() {
```

```
    gravityTds.setTemperature(temperature); // set the temperature and execute temperature compensation
    gravityTds.update(); //sample and calculate
    tdsValue = gravityTds.getTdsValue();
```

```
    for(int i=0;i<10;i++)
    {
        buffer_arr[i]=analogRead(A0);
        delay(30);
    }
    for(int i=0;i<9;i++)
    {
        for(int j=i+1;j<10;j++)
        {
            if(buffer_arr[i]>buffer_arr[j])
            {
                temp=buffer_arr[i];
                buffer_arr[i]=buffer_arr[j];
                buffer_arr[j]=temp;
            }
        }
    }
    avgval=0;
    for(int i=2;i<8;i++)
    avgval+=buffer_arr[i];
    float volt=(float)avgval*5.0/1024/6;
    float ph_act = -5.70 * volt + calibration_value;
```

```
    Serial.print("pH = ");
    Serial.print(ph_act/1.3,0);
    Serial.print(" ");
    Serial.print("TDS = ");
    Serial.print(tdsValue,0);
    Serial.print("ppm");
    Serial.println("");
    delay(1000);
}
```

Image Processing:

So one data element we get from the circuit and another is the software element where we do machine learning and Image processing to extract the visual features of the water sample.

Code:

```
import cv2
import numpy as np
```

```

img = cv2.imread(r'C:\Users\Admin\Pictures\Camera
Roll\water\water-tap\13.jpg')

img1 =
cv2.imread(r'C:\Users\Admin\Pictures\Camera
Roll\water\water-tap\13.jpg',0)
blur = cv2.GaussianBlur(img,(299,299),0)
hsv = cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
# img = cv2.imread('messi5.jpg',0)
edges = cv2.Canny(img1,130,200)
# l = np.array([55,0,19])
# u = np.array([73,255,255])

l = np.array([61,234,208])
u = np.array([74,255,239])

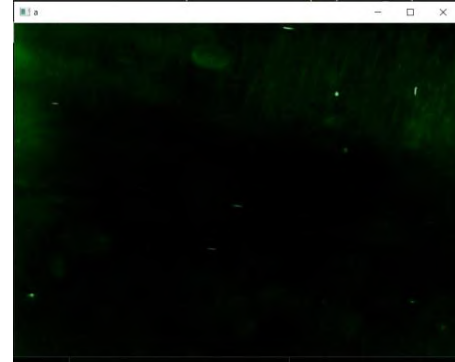
mask = cv2.inRange(hsv,l,u)

cont,har =
cv2.findContours(mask,cv2.RETR_TREE,cv2.CHAI
N_APPROX_SIMPLE)

for i in cont:
    area = cv2.contourArea(i)
    cv2.drawContours(img,[i],-1,(0,0,255),2)
    # if area<200:
    #     cv2.drawContours(img,[i],-1,(0,0,255),2)
cv2.imshow('a',img)
cv2.imshow('b',mask)
cv2.imshow('c',edges)
cv2.imshow('d',blur)
cv2.waitKey(0)
cv2.destroyAllWindows()
print(blur)
hsv1 = cv2.cvtColor(blur,cv2.COLOR_BGR2HSV)
hue = hsv[:,:,0]
sat = hsv[:,:,1]
val = hsv[:,:,2]

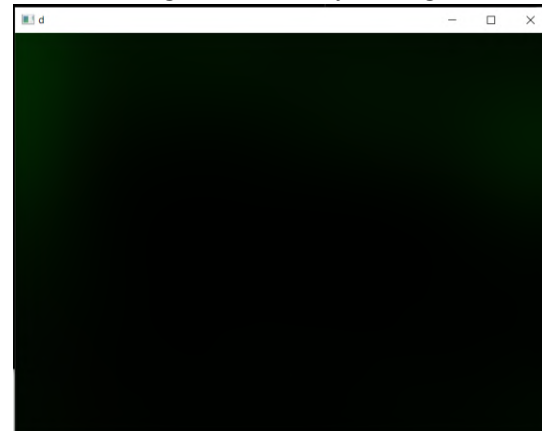
# print(hue)
# print(hue.shape)
print(hue.min())
print(hue.max())
print(sat.min())
print(sat.max())
print(val.min())
print(val.max())

```

Results:**Original image:****Particle detection:****Gradient of the image to check the transparency:**

We also blurred the image to a very high level to check the visibility through the water. More black it is, the clearer it will be, and more green means less transparency.

As in the given picture the image is black mostly which is telling us the visibility was high



Checking the transparency level:

We are checking the minimum HSV and the maximum HSV of the image to find out the average color gradient

```
hue min 0
hue max 173
sat min 0
sat max 255
val min 0
va; max 229
PS C:\Users\Admin\Desktop> █
```

Machine learning:

After image processing we have used a CNN machine learning algorithm to train a model to predict the turbidity of the water by doing the image classification.

First of all we have created a CNN Model

```
# create CNN Model

class LeNet:
    @staticmethod
    def build(width, height, depth, classes):
        # initialize the model
        model = Sequential()
        inputShape = (height, width, depth)

        # if we are using "channels first", update the input
        shape
        if K.image_data_format() == "channels_first":
            inputShape = (depth, height, width)

        # first set of CONV => RELU => POOL layers
        model.add(Conv2D(20, (5, 5), padding="same",
            input_shape=inputShape))
        model.add(Activation("relu"))
        model.add(MaxPooling2D(pool_size=(2, 2),
            strides=(2, 2)))

        # second set of CONV => RELU => POOL
        layers
        model.add(Conv2D(50, (5, 5), padding="same"))
        model.add(Activation("relu"))
```

```
model.add(MaxPooling2D(pool_size=(2, 2),
    strides=(2, 2)))

# first (and only) set of FC => RELU layers
model.add(Flatten())
model.add(Dense(500))
model.add(Activation("relu"))

# softmax classifier
model.add(Dense(classes))
model.add(Activation("softmax"))

# return the constructed network architecture
return model
```

After setting up the model we have defined dataset and EPOCH Numbers to train the model

```
DATASET = "Data_Water" # this folder must contain
three subfolder with images
MODEL = "Water.model" # name to store the model
on disk
PLOT = "plot.png" # plot name
# initialize the number of epochs to train for, initial
learning rate,
# and batch size
EPOCHS = 50
INIT_LR = 1e-3
BS = 32
```

After this we have initialized the data and labels:

```
# initialize the data and labels
print("[INFO] loading images...")
data = []
labels = []

# grab the image paths and randomly shuffle them
imagePaths = sorted(list(paths.list_images(DATASET)))
random.seed(42)
random.shuffle(imagePaths)

# progress bar
with tqdm(total=len(imagePaths)) as pbar:

    # loop over the input images
    for idx, imagePath in enumerate(imagePaths):
        # load the image, pre-process it, and store it in the
        data list
```

```

image = cv2.imread(imagePath)
image = cv2.resize(image, (28, 28))
image = img_to_array(image)
data.append(image)

# extract the class label from the image path and
update the
# labels list
label = imagePath.split(os.path.sep)[-2]

# label = 1 if label == "Good_pizza" else 0
# print("ex: ", label)

if label == "Turbidity Greater Than 1":
    label = 0
elif label == "Highly turbid":
    label = 1
elif label == 'Non - Turbid':
    label = 2

# print("pr: ", label)

labels.append(label)

# update the progressbar
pbar.update(1)

```

After this next step is to split the data into test and train data

```

#Step 4: Creating your training and testing splits
# partition the data into training and testing splits using
75% of
# the data for training and the remaining 25% for
testing
(trainX, testX, trainY, testY) = train_test_split(data,
labels, test_size=0.25, random_state=42)
trainX.shape
trainY.shape
# convert the labels from integers to vectors
trainY = to_categorical(trainY,num_classes=1)
testY = to_categorical(testY,num_classes=1)

```

Next step - Data Processing:

```

#Step 5: Data Preprocessing (Augmentation)
# construct the image generator for data augmentation
aug = ImageDataGenerator(rotation_range=30,

```

```

width_shift_range=0.1,
height_shift_range=0.1,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode="nearest")

```

After data processing i have done the compiling of tensor flow CNN Model

```

#Step 6: Compiling your tensorflow CNN model
# initialize the model
print("[INFO] compiling model...")
model = LeNet.build(width=28, height=28, depth=3,
classes=2)
opt = Adam(lr=INIT_LR, decay=INIT_LR /
EPOCHS)
model.compile(loss="categorical_crossentropy",
optimizer=opt, metrics=["accuracy"])

print("[INFO] model compiled...")

```

Training the model:

```

#Step 7: Training your model on your training data
# train the network
print("[INFO] training network...")
H = model.fit(x=aug.flow(trainX, trainY,
batch_size=BS),
validation_data=(testX, testY),
steps_per_epoch=len(trainX) // BS,
epochs=EPOCHS,
verbose=1)

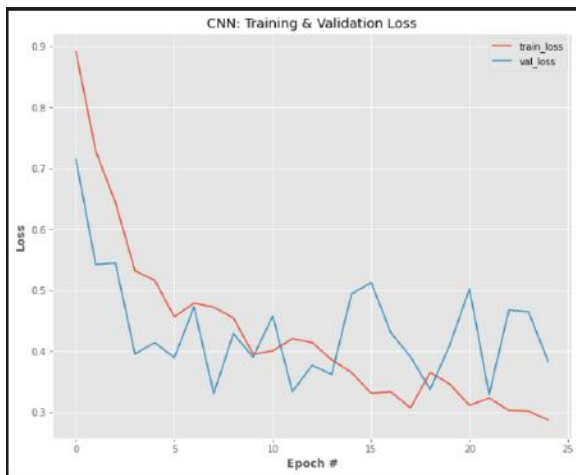
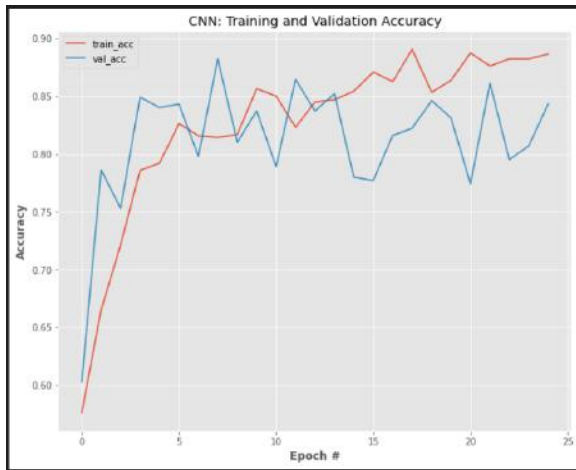
```

Plotting the training and validation loss

```

# plot the training and validation loss
N = np.arange(0, EPOCHS)
plt.style.use("ggplot")
plt.figure(figsize = [10,8])
plt.plot(N, H.history["loss"], label="train_loss")
plt.plot(N, H.history["val_loss"], label="val_loss")
plt.title("CNN: Training & Validation Loss")
plt.xlabel("Epoch #", weight="bold")
plt.ylabel("Loss", weight="bold")
plt.legend()
plt.show()

```



Result Prediction :

There is a specific code that is used to do the prediction, which is called test code and that code uses the model file trained using CNN to do the prediction on the new image.

```
import cvzone
import cv2
from cvzone.ClassificationModule import Classifier

cap = cv2.VideoCapture(0)
myclassifier = cvzone.Classifier("C:\\Users\\khatr\\OneDrive\\Desktop\\ahaana jamb\\My_model.h5", "C:\\Users\\khatr\\OneDrive\\Desktop\\ahaana jamb\\labels.txt" )
fpsReader = cvzone.FPS()
while True:
    _, img = cap.read()
```

```
predictions, index = myclassifier.getPrediction(img)
fps = fpsReader.update()
print(fps)

cv2.imshow('image',img)
if cv2.waitKey(10) & 0xFF == ord('q'): # Break gracefully
    break
cap.release() # release our webcam
cv2.destroyAllWindows() # close down our frame
```

Conclusion:

This project is made to work on checking the quality of the water in the nearby water bodies and give it a quality coefficient. We have integrated 3 things to make it work. One is the the quality sensing device which senses the pH level and the TDS value of the water, second is image processing to detect the gradient of the water and third is machine learning to predict and give a quality coefficient to the water depending on its quality and reusability. Our machine learning model works with the accuracy of 89%. Whereas the circuit works perfectly with high accuracy in the detection of TDS and pH. To conclude we can say that our solution is a real implementation and fast solution to detect the quality of the nearby water body and it can also put that water in use if that can be used.

Future Scope:

Soon we will start the field testing of the solution. Once we start with that, we will be collecting more data every time. Which will increase our database and it will also improve the range of our solution. We need to work on the design of the prototype and make it easier to use and stable. Right now, we have made a 3D printed case which is holding everything but it is basic and it needs to be improved to make it look like an end product.

References:

1. Wang, X., Zhang, F. & Ding, J. Evaluation of water quality based on a machine learning algorithm and water quality index for the Ebinur Lake Watershed, China. *Sci Rep* 7,

- 12858 (2017).
<https://doi.org/10.1038/s41598-017-12853-y>
2. Theyazn H. H Aldhyani, Mohammed Al-Yaari, Hasan Alkahtani, Mashael Maashi, "Water Quality Prediction Using Artificial Intelligence Algorithms", *Applied Bionics and Biomechanics*, vol. 2020, Article ID 6659314, 12 pages, 2020.
<https://doi.org/10.1155/2020/6659314>
 3. Yi Lin, Lang Li, Jie Yu, Yuan Hu, Tinghui Zhang, Zhanglin Ye, Awase Syed, Jonathan Li,
An optimized machine learning approach to water pollution variation monitoring with time-series Landsat images, *International Journal of Applied Earth Observation and Geoinformation*, Volume 102, 2021, 102370, ISSN 1569-8432,
<https://doi.org/10.1016/j.jag.2021.102370>.
 4. Ding and Liu, 2019,X. Ding, L. Liu,"Long-term effects of anthropogenic factors on nonpoint source pollution in the upper reaches of the Yangtze river" *Sustain.*, 11 (2019), [10.3390/su11082246](https://doi.org/10.3390/su11082246)
 5. P. Zeilhofer, L. V. A. C. Zeilhofer, E. L. Hardoim, Z. M. . Lima, and C. S. Oliveira, "GIS applications for mapping and spatial modeling of urban-use water quality: a case study in District of Cuiabá, Mato Grosso, Brazil," *Cadernos de Saúde Pública*, vol. 23, no. 4, pp. 875–884, 2007.
 6. K. Farrell-Poe, W. Payne, and R. Emanuel, *Water Quality & Monitoring*, University of Arizona Repository, 2000,
<http://hdl.handle.net/10150/146901>.
 7. Y. C. Lai, C. P. Yang, C. Y. Hsieh, C. Y. Wu, and C. M. Kao, "Evaluation of non-point source pollution and river water quality using a multimedia two-model system," *Journal of Hydrology*, vol. 409, no. 3-4, pp. 583–595, 2011.