Report on Smart Irrigation System

Member 1 - Subhajyoti Saha
Roll No - 21111269
Email - subhajyoti21@iitk.ac.in

Member 2 - Shiv Kumar Yadav
Roll No- 21111057
Email- shivky21@iitk.ac.in

3rd November, 2021

1 Introduction :

In the recent scenario of Water Crisis in Urban and Rural areas, it has become highly necessity to optimize the use of fresh water. As the source of our domestic water need is ground water, we need to use domestic water optimally to preserve the ground water level. A large amount of domestic water is wasted in irrigation. Therefore, if we can model the water needed for irrigation optimally based on the past history, it would be a highly valuable task for preservation of ground water. In this project we have tried to shape this. In our project, we have built a smart Irrigation System, which is combined with IoT and ML based system. Our system gets input from the humidity and temperature sensor in WokWi Simulator, and then we run a Machine Learning Model (a simple Neural Network based Model), which predicts the water flow needed to be sent to the irrigation field from the sensor values based on the previously trained pattern. Due to the resource constraint nature of our edge nodes, we have trained our ML model offline, and in the online setting, we just have predicted the necessary amount of water flow. Our system does not send any water during night time.

2 Smart Irrigation System :

Here, we have build the smart irrigation system to optimally make use of ground water for irrigation in a domestic setting. Here, we have build the whole project on the WokWi IoT Simulator which is freely available and currently a ongoing open source project. We have deployed sensors in the simulator setting which would detect the temperature and humidity values from the environment, which we assumed to be the most important factor for predicting the water needed for our irrigation system. We also used a sensor to detect whether it is day time or

not, and based upon that we have stopped flow of water in the irrigation field at night time. There is one LCD screen which denotes the percentage of water flow sent to a irrigation field. And the servo motors were used to control the amount of water sent to our irrigation field. The rotation of angle of the servo motor is mapped from the percentage amount of water needed predicted by the ML model. We have set up 4 such units, which would separately detect temperature and humidity, send the values to the ML model, receive the prediction from the ML model about the amount of water needed in percentage, and the servo motor presents in each of the four units rotates its arms accordingly to facilitate the flow of the necessary amount of water. The LCD screen displays the amount of water flowed to all the four system. We have at first make the necessary connection on the MEGA arduino Board, and then we provided the necessary codes which would connect all the sensors and servo motor together. We have used some arrays to hold the necessary things f the senssor, servo motor of the 4 units.

3 Sensor :

We have used two sensors to detect the humidity and temperature.

3.1 Temperature and Humidity Sensor :

We have used the DHT22 sensor to detect the humidity and the temperature of the irrigation system.These sensors are very basic and slow, but are great for hobbyists who want to do some basic data logging. The DHT sensors are made of two parts, a capacitive humidity sensor and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is fairly easy to read using any microcontroller.We have used for DHT22 sensor for four units of our system. The codes to read temperature and humidirty from these sensors bvalues are also pretty much easy. These sensors sometimes may capture 'NaN' values, where it might be unable to get the values.

3.2 Light Detecting Sensor (LDR) :

A Light Dependent Resistor (LDR) is also called a photoresistor or a cadmium sulfide (CdS) cell. It is also called a photoconductor. It is basically a photocell that works on the principle of photoconductivity. The passive component is basically a resistor whose resistance value decreases when the intensity of light decreases. This optoelectronic device is mostly used in light varying sensor circuit, and light and dark activated switching circuits. Some of its applications include camera light meters, street lights, clock radios, light beam alarms, reflective smoke alarms, and outdoor clocks. We have used this LDR sensor for detecting the day or night condition. If the ilumination is high than a certain value it is detected as a day. And if the illumination is low, it is detected as

night condition. It has 4 pins i.e. one Digital Output Pin, One Analog Output Pin, One Vcc and one GND. We have taken our input from the Digital Output Pin.

3.3 Servo Motor :

Servo motors or "servos", as they are known, are electronic devices and rotary or linear actuators that rotate and push parts of a machine with precision. Servos are mainly used on angular or linear position and for specific velocity, and acceleration. We have used four servo motor for controllingh the amount of water flow in the irrigation system. The servo motor has actually three pins i.e. one servo signal pin, one Vcc and one GND.We had make the connections of Servo Motor as on the Mega Arduino Board and the amount of water flowed into the system is controlled in the four system.

3.4 LCD :

A (2 * 16) LCD screen is added with our Arduino Mega Board to display the amount of water flowed into the system.

4 Machine Learning Model

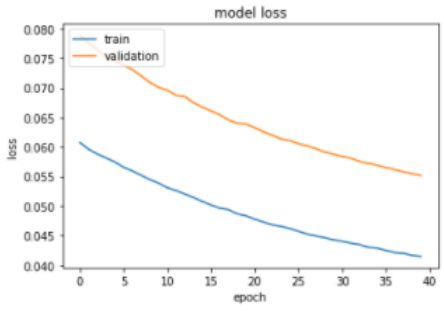
We build a Neural Network to understand the mapping from the parameters i.e., how the attributes of temperature and humidity add up to produce the waterflow percentage. The model uses simple dense layers of Keras. The developed model has 2 such hidden layers. Our objective was to decrease the error in prediciton and also keep the model smaller to deploy it efficiently on the simulator.

```
model = Sequential()
model.add(Dense(8, input_dim=2,  activation='relu'))
model.add(Dense(4, activation='relu'))
model.add(Dense(1, activation='relu'))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 8)                 24
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 36
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 5
=================================================================
Total params: 65
Trainable params: 65
Non-trainable params: 0
_________________________________________________________________
```

5 Epochs of the ML model

We observe how the models loss decreases with the increasing epochs and try to identify when the model tries to overfit the training dataset and perform badly on the test dataset. This epoch is selected as the number of epoch to be trained for. We after computing found that the epoch number of 20 is good for most of the hyper parameter settings.



6 Evaluation of model/Accuracy

The prediction made by our model is analyzed via the Mean Squared Error(MSE) to observe the average deviation from the actual value. A good model tries to minimise the MSE so that the predictions are more accurate. In the end we also compute the R^2 of the model built.

The R^2 is a measure of how much the model explains the underlying relationship when compared to having a prediction equal to the average of the prediction values. It is also a measure of the amount of variance that is explained by the model. Our model gets a R^2 score of around 80

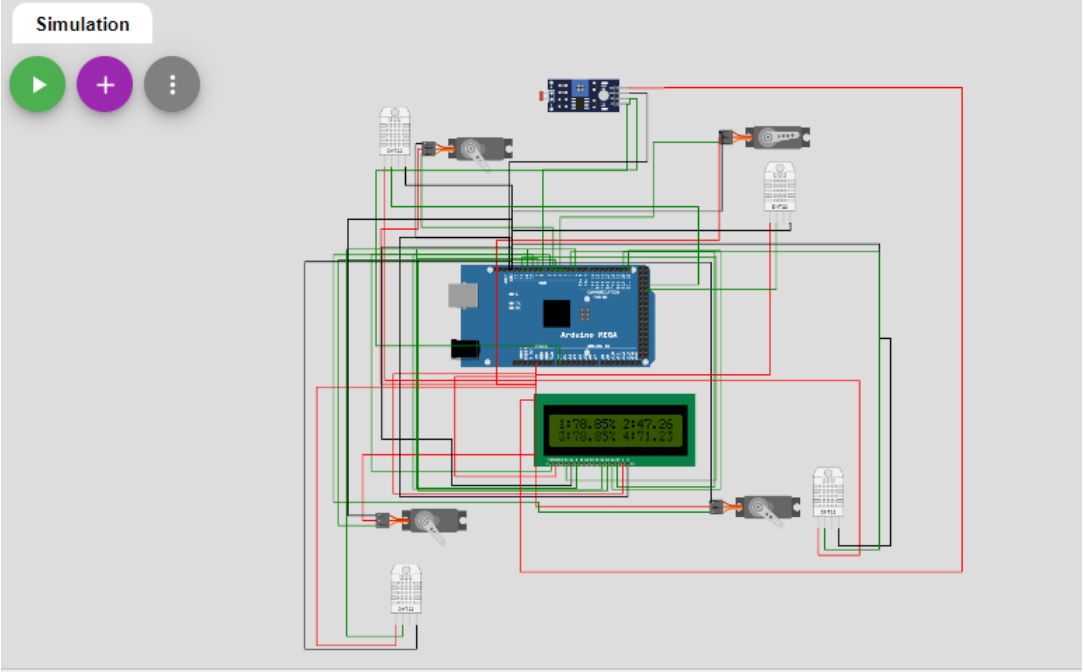
7 Steps to run the model

The model is build using the ipynb for better understanding and having more comments. The weights that are obtained from the model are passed in a matrix in the ".ino" file at the simulator. The simulator computes the dense layer operations by doing a matrix multiplication and the activation is performed by putting a condition on the outputs of the hidden layer. The choosen activation function relu introduces non linearity and is simple to implement mathematically.

Make sure you have downloaded or placed the csv file in the same location as the ipynb notebook.

8 Simulator :

As we have discussed here we have extensively used the WokWi Online IoT platform Simulator as it was not possible for us to implement the whole model in real life situation. Wokwi is an online simulator for Arduino and Electronics. It's designed for makers, by makers.Wokwi can be extensively used to build our ideas, quickly make prototype of our ideas before implementing it in real world scenarios. Simulation in the Wokwi can be easily shared across friends. Wokwi provide us the simulate the behaviour of Arduino Boards, EPS32 Boards etc IoT Systems. Here, we have simulate the behaviour of a Arduino Mega Board in the Wokwi to make the prototype of the irrigation system for our assignment.



9 Links:

The link of our Wokwi Simulator project is as: Simulated Project Link. The notebook file containing the code is added in the zip file and is also : present at this link