PIPELINE LEAK DETECTION USING EDGE DATA ANALYTICS

Testbench setup

Task performed

- Literature Survey
- Hardware procurement
- Hardware setup
- Data Collection
- Data Modelling
- Deployment

An edge device which is capable of detecting pipe leakage using machine learning model deployed in edge device.

IDEs

- Arduino IDE
- Jupyter Notebook

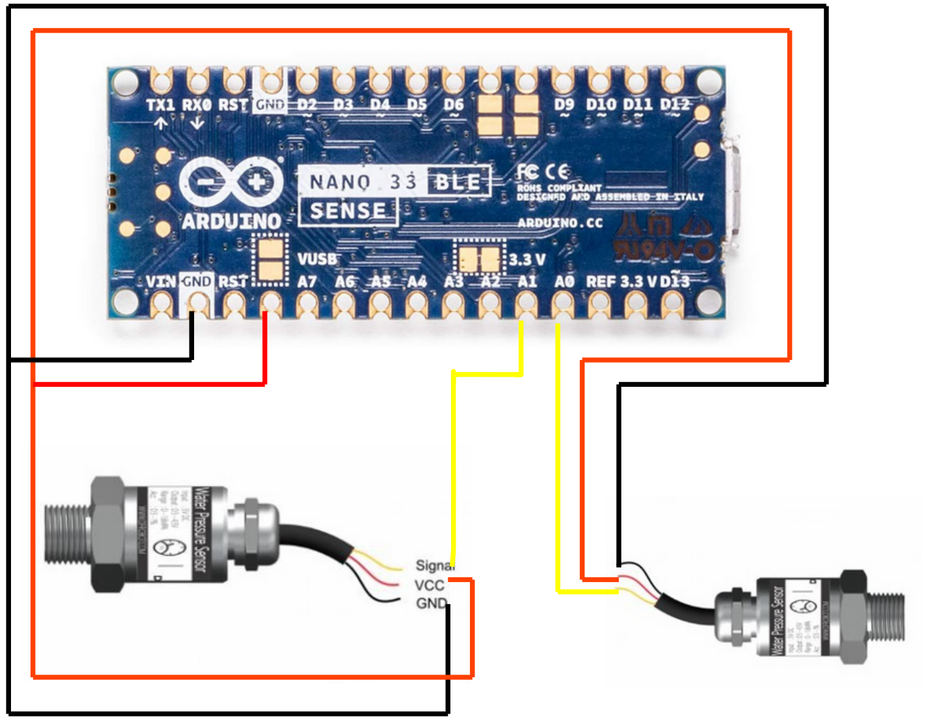
Frameworks

- Tensorflow
- Keras

Libraries

- Numpy
- pandas
- <TensorFlowLite.h>
- <ArduinoBLE.h>

HARDWARE SETUP



Components required :

1.Arduino nano BLE sense

2.Water Pressure Sensor

Gravity: Analog Water Pressure Sensor SKU: SEN0257

DATA COLLECTION

Data Collection Setup

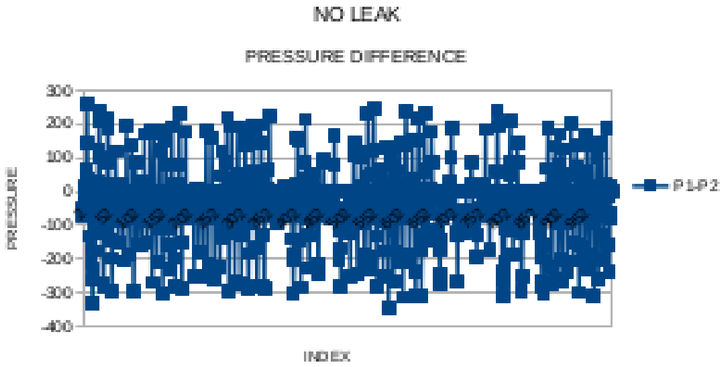
1.Hardware setup:-Sensor Connectivity

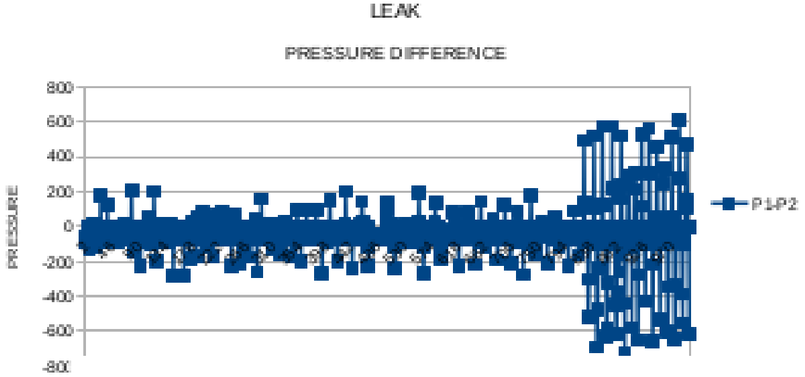
2.Recording /Save data :-

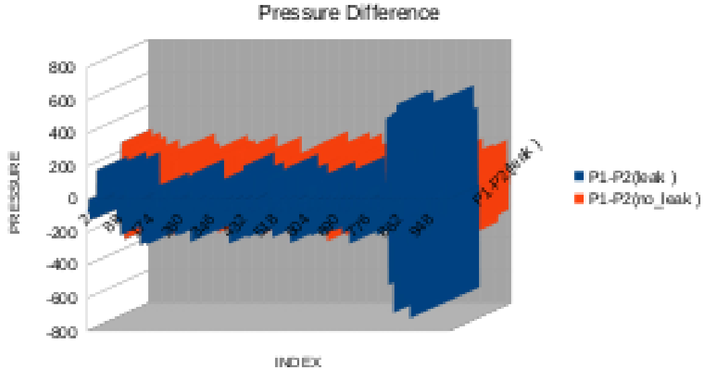
•upload the sensor connectivity code to the hardware module

•connect serially through python code and save the datas to csv file

DATA PREPARATION







Data Preparation

Around 2000 datas are used for training Ml Model for Deployment With the Label of 0 and 1 with respect to Leak and No Leak.

	A	B	C	D	E
1	P1	P2	P1-P2	(P1-P2)	LABEL
2	1033.9	1082.3	-48.39	-48.4	0
3	1020.6	1093.8	-73.1999999999999	-73.2	0
4	1035.3	1090.2	-54.9000000000001	-54.9	0
5	999.9	1049.9	-50.0000000000001	-50	0
6	897.3	923	-25.7	-25.7	0
7	829	816.7	12.3	12.3	0
8	711.8	645.9	65.9	65.9	0
9	526.2	379.7	146.5	146.5	0
10	138	-120.7	258.7	258.7	0
11	-81.7	-120.7	39	39	0
12	-120.7	-120.7	0	0	0
13	-120.7	-120.7	0	0	0
14	-120.7	-118.3	-2.40000000000001	-2.4	0
15	-40.2	-8.4	-31.8	-31.8	0
16	193	326	-133	-133	0
17	283.3	455.4	-172.1	-172.1	0
18	389.5	601.9	-212.4	-212.4	0
19	517.7	777.7	-260	-260	0
20	653.2	984	-330.8	-330.8	0
21	766.7	1096.3	-329.6	-329.6	0
22	869.2	1100	-230.8	-230.8	0
23	912	1086.5	-174.5	-174.5	0
24	1012.1	1106.1	-93.9999999999999	-94	0
25	1025.5	1098.7	-73.2000000000001	-73.2	0
26	1027.9	1087.7	-59.8	-59.8	0
27	999.9	1043.8	-43.9	-43.9	0
28	951	987.6	-36.6	-36.6	0

PHASES OF EMBEDDED MACHINE LEARNING MODELLING

- *Imports &Environment setup*
- *Data Preparation*
- *Model Training*
- *Model Evaluation*
- *Converting Models*

A representative dataset is provided and two models saved, one in TFLite format without quantization and one in TFLite format with int8 quantization for activations and weights.

- *Compare Model Performance*

Compare loss or performance on the test set to evaluate how much fidelity is lost with compression and quantization.

a. prediction
b. loss
c. size

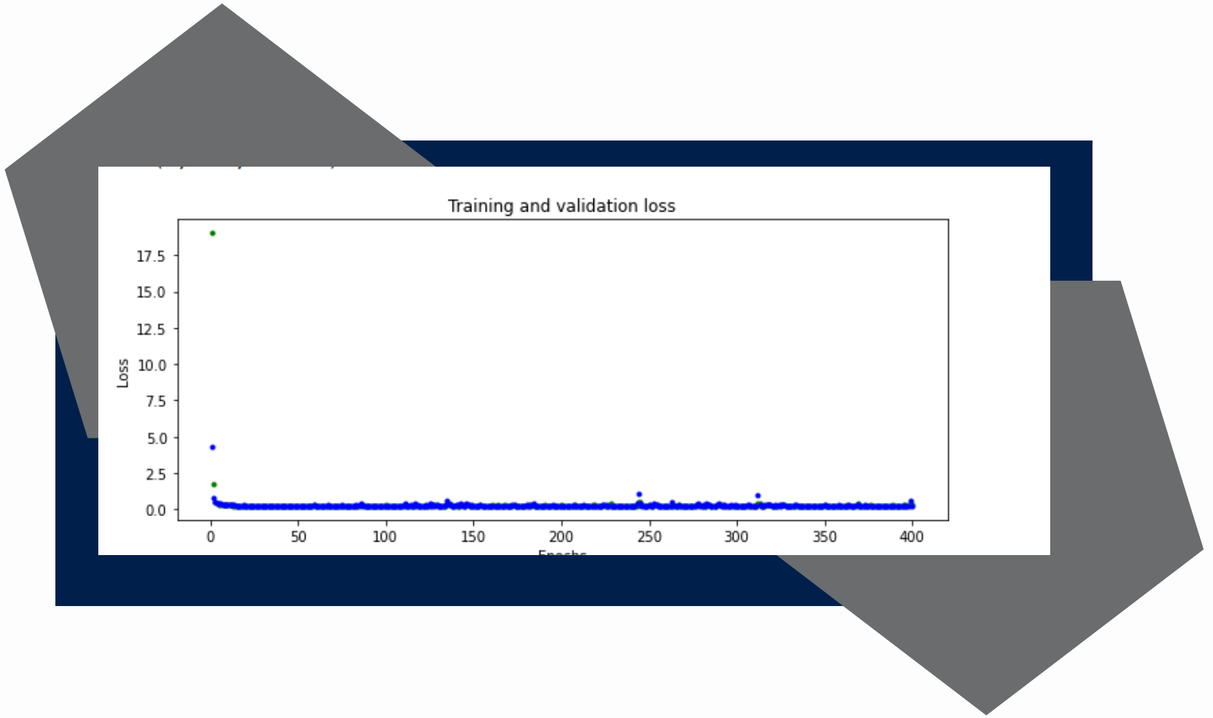
- *Generate a Tensorflow Lite for microcontroller model*

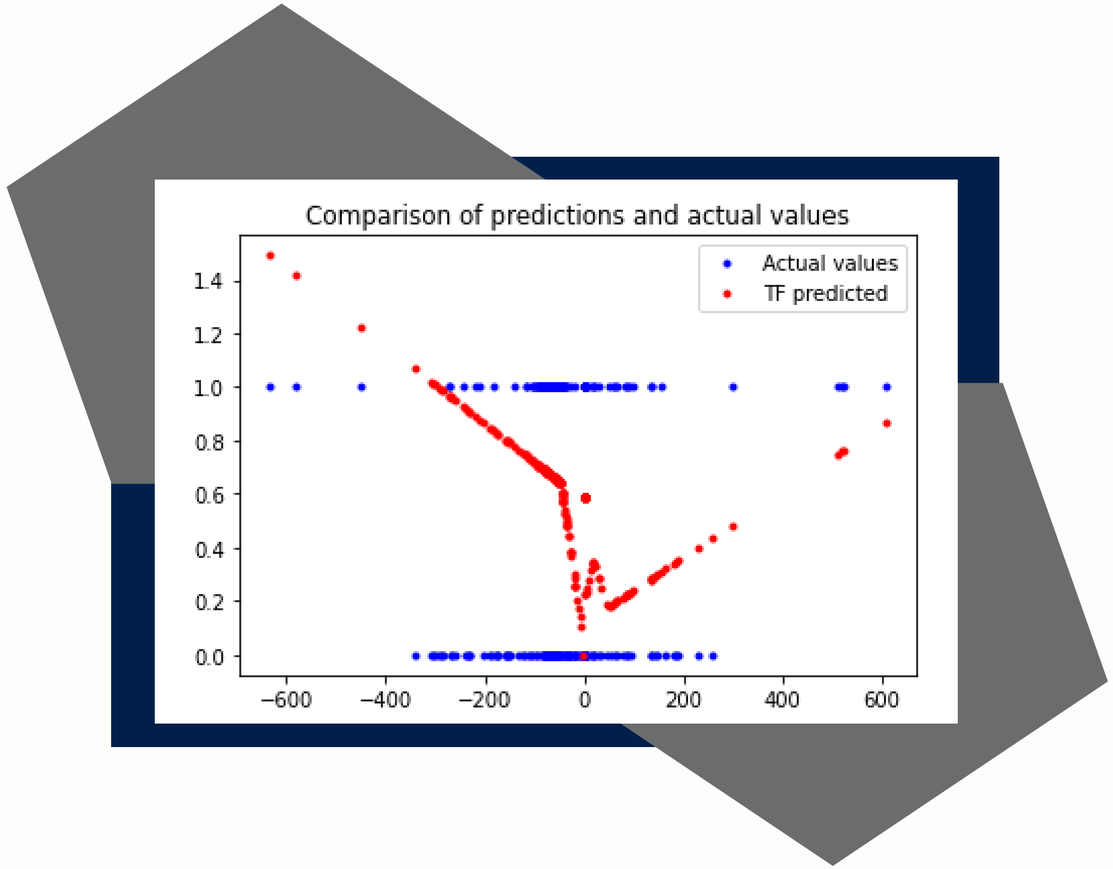
Convert the TensorFlow Lite quantized model into a C source file that can be loaded by TensorFlow Lite for Microcontrollers.

- *Deploying model to Arduino BLE nano Sense board.*

MODEL EVALUATION

The training loss is a metric used to assess how a deep learning model fits the training data.
Validation loss is a metric used to assess the performance of a deep learning model on the validation set.



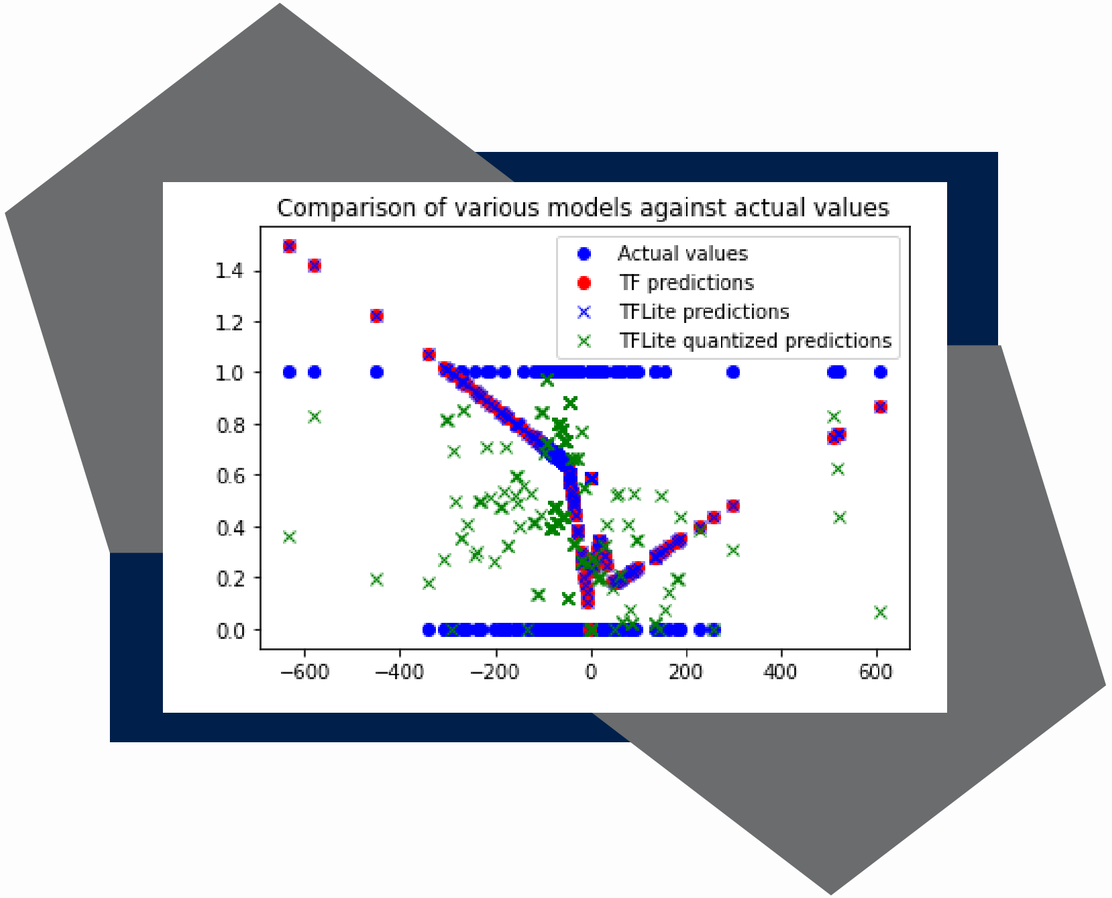


Over all Accuray of the system :63.38%

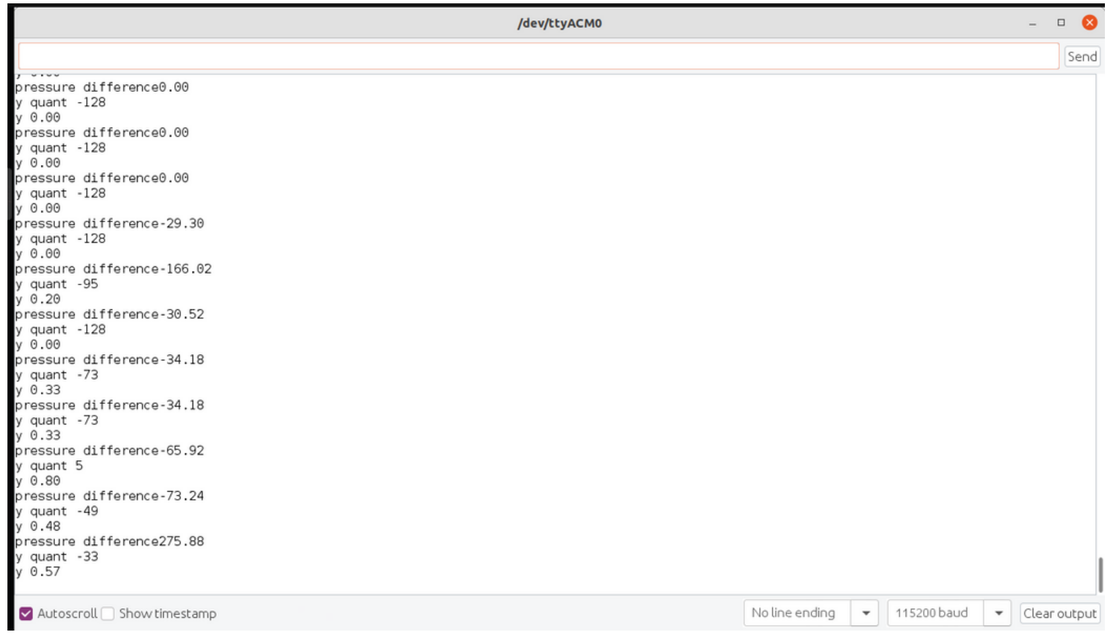
MODEL PERFORMANCE COMPARISON

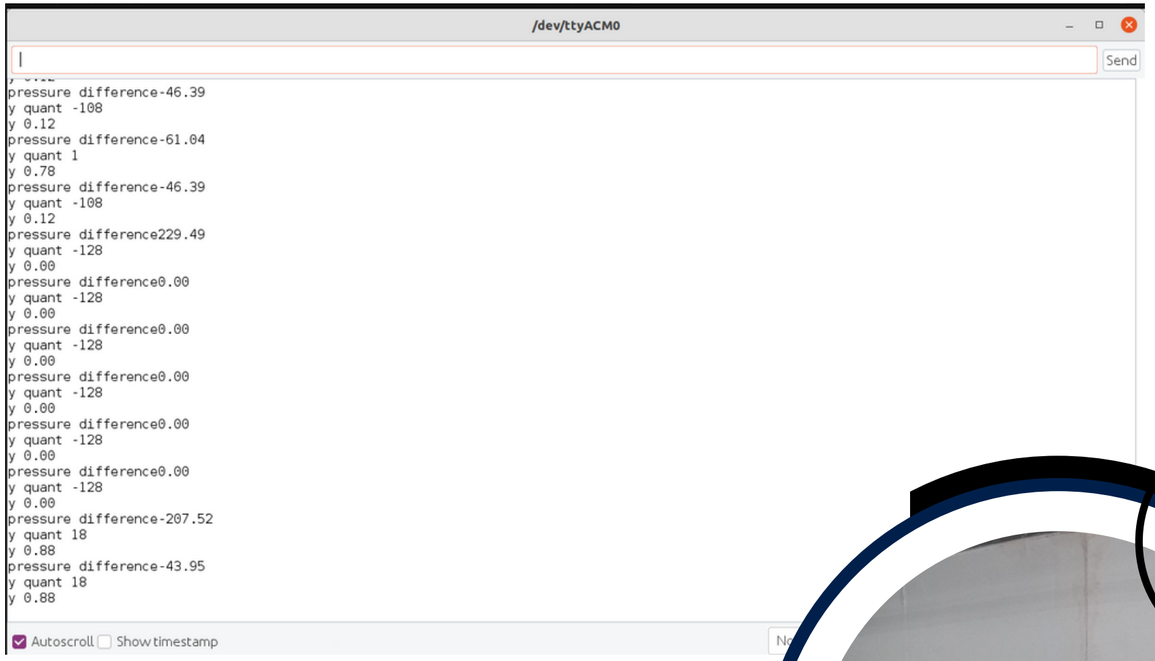
Compare loss or performance on the test set to evaluate how much fidelity is lost with compression and quantization.

- prediction
- loss
- size



LIVE TESTING & CLASSIFICATION





Live Classification gives quatized output as y with a detection of around 0.6-0.88 as leak and 0-0.22 as no leak