UNIVERSITY OF JAMMU

UNIVERSITY OF JAMMU RESEARCH FUND (UoJRF)

Form-V

PROJECT COMPLETION REPORT

(Submit in duplicate)

1. **Title of the project:** Tracing out anomalies in the working of IoT sensors under various conditions and finding solution
2. **Name & Designation of Principal Investigator:** Prof. Lalit Sen Sharma, Department of Computer Sc. & IT, University of Jammu
3. **Name & Designation of Co- Principal Investigator/s:** Nil
4. **Duration of the project:** One year
5. **Sanctioned grant:** Rs. 2,00,000 (Two lakhs)
6. **Date of initiation of the project:** 23/01/2023
7. **Date of closure of the project:** 15/03/2024
8. **Whether the Utilization Certificate and statement of expenditure has been submitted?**

   Yes/No (If yes, mention the date and append the photocopy of the same)

   Yes. Copy attached
9. **Approved objectives:** Yes

   To study the anomalies in the working of a sensor in a given network conditions.

   To identify the abnormal pattern so that the anomalies in the working of the sensors.

   To find solution to remove the anomalies at the data level.
10. **Title of the research paper published from out of the current project work** (If any, attach reprint) Yes

    Study of Anomaly Detection in IoT Sensors, International Journal for Research in Applied Science & Engineering Technology, ISSN: 2321-9653, Volume II, Issue VIII, Pp 767-774
11. **Title of the research paper accepted for publication from current research work** (If any, attach copy of acceptance letter) NA
12. **Report of the completed research project highlighting the deliverables** (Attach document-Min. 3000 words) Copy Attached
13. **Details of the consumable and non-consumable (including equipment) material procured**

from current research project grant. Copy Attached

14. **Has the non-consumable material (including equipment) been handed over to the concerned department?** Yes/No (If yes, attach a certificate issued by concerned HoD in this regard) Yes. Certificate Attached
15. **Has the stock register carrying entries of consumable/ non-consumable (including equipment) handed over to the concerned department?** Yes/No (If yes, attach a certificate issued by concerned HoD in this regard) Certificate Attached
16. **Was power point presentation of the current research work made before DRPMC by PI/Co-PI?** Yes/No (If yes, attach a certificate issued by concerned Dean/ HoD in this regard) (If no, the reasons thereof) Not Required

Comments of the concerned DRPMC

On the basis of progress report the work undertaken in the project is as per the objectives and is satisfactory.

4/4/24

Prof. Lalit Sen Sharma, PI, Research and Seed Grant
Department of Computer Sc. & IT, University of Jammu

Members of the concerned DRPMC

Dean, Mathematical Sc.	Head, Computer Sc. & IT 4/4/24	Senior Professor of the Faculty	Senior Professor of the Department	Senior Associate Professor (By Rotation)

4/4/24

Head of the concerned Department

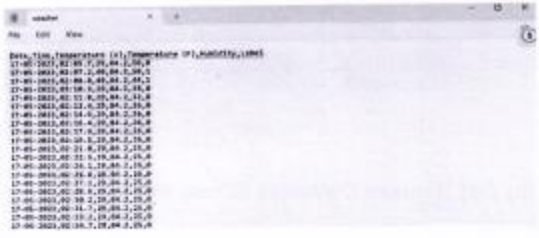
REPORT OF PROJECT UNDER RUSA RESEARCH GRANT

Background:

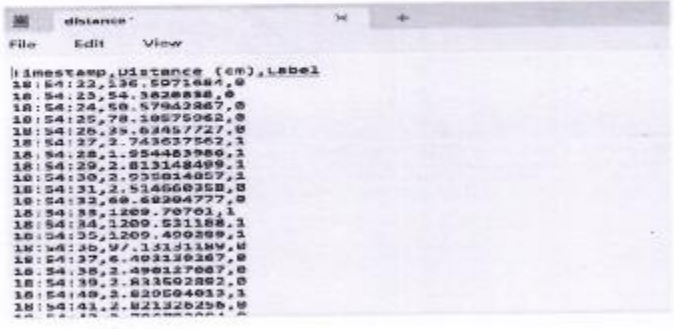
Anomalies in IoT sensor data occurs due to a variety of reasons, including sensor malfunctions, environmental changes, cyber-attacks, or even human errors. Detecting and identifying these anomalies is crucial for maintaining the reliability, security, and efficiency of IoT sensor networks. The problem under study focused to identify anomalies in IoT sensors using various techniques, including statistical techniques, cluster-based techniques and nearest neighbor techniques. These techniques offer different approaches and algorithms to detect anomalies in IoT sensor data. The anomalies have been identified as Point Anomalies, Contextual Anomalies, Collective Anomalies, Spatial Anomalies and Temporal Anomalies.

Methodology:

The data generated by the sensors, namely; Temperature Sensor, Ultrasonic Distance Sensor, LDR Sensor and Soil Moisture Sensor was collected using RaspberryPi board in CSV format.



Temperature Sensor Data



Ultrasonic Distance Sensor data

```
File  Edit  View

Time,Intensity,Label
14:00:13,369,0
14:00:33,510,1
14:00:48,471,0
14:01:04,248,0
14:01:21,317,0
14:01:37,226,0
14:01:54,317,0
14:02:13,219,0
14:02:30,276,0
14:02:48,48,1
14:03:05,248,0
14:03:22,338,0
14:03:38,224,0
14:03:57,288,0
14:04:14,291,0
14:04:30,240,0
14:04:47,300,0
14:05:04,220,0
14:05:24,256,0
14:05:41,225,0
```

Figure: LDR Sensor Data

```
File  Edit  View

Time,moisture,Label
14:00:13,0.7,0
14:00:31,0.7,0
14:00:48,0.69,0
14:01:04,0.69,0
14:01:21,0.69,0
14:01:37,0.68,0
14:01:54,0.69,0
14:02:13,0.69,0
14:02:30,0.68,0
14:02:48,0.68,0
14:03:05,0.68,0
14:03:22,0.68,0
14:03:39,0.68,0
14:03:57,0.68,0
14:04:14,0.68,0
14:04:30,0.68,0
14:04:47,0.68,0
14:05:04,0.68,0
14:05:24,0.68,0
14:05:41,0.68,0

Ln 1, Col 1                                    100%
```

Figure: Soil Moisture Data

The statistical techniques, namely; Z-score ($z = (x - \mu) / \sigma$), K-mean Clustering (Elbow method) and K-Nearest Neighborhood were used to detect anomalies.

The experiment was further extended to be performed on Arduino Uno board and the kits based on Arduino Uno board with the sensors as shown below.

Date&Time	Beats	Label
[2024-02-24 15:35:07	54	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0
[2024-02-24 15:35:07	53	0

Figure: Heartbeat Sensor Data

Date&Tim	Humidity	Temperat	Temperat	Label
[2024-02-:	59.30%	16.70Â°C	62.06Â°F	0
[2024-02-:	59.30%	16.70Â°C	62.06Â°F	1
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	1
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	1
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	1
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	0
[2024-02-:	59.20%	16.80Â°C	62.24Â°F	1
[2024-02-:	59.20%	16.80Â°C	62.24Â°F	1
[2024-02-:	59.20%	16.80Â°C	62.24Â°F	0
[2024-02-:	59.20%	16.80Â°C	62.24Â°F	0
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	0
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	0
[2024-02-:	59.30%	16.70Â°C	62.06Â°F	0
[2024-02-:	59.30%	16.70Â°C	62.06Â°F	0
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	1
[2024-02-:	59.30%	16.80Â°C	62.24Â°F	0
[2024-02-:	59.30%	16.70Â°C	62.06Â°F	0
[2024-02-:	59.30%	16.70Â°C	62.06Â°F	0
[2024-02-:	59.40%	16.70Â°C	62.06Â°F	0

Figure: Temperature Sensor Data

	A	B	C	D	E
1	Date&Tim	Distance c	Label		
2	[2024-02-:	12	0		
3	[2024-02-:	13	0		
4	[2024-02-:	12	0		
5	[2024-02-:	12	0		
6	[2024-02-:	13	0		
7	[2024-02-:	12	0		
8	[2024-02-:	13	0		
9	[2024-02-:	1	1		
10	[2024-02-:	1200	1		
11	[2024-02-:	12	0		
12	[2024-02-:	13	0		
13	[2024-02-:	210	0		
14	[2024-02-:	45	0		
15	[2024-02-:	45	0		
16	[2024-02-:	45	0		
17	[2024-02-:	45	0		
18	[2024-02-:	45	0		

Figure: Ultrasonic Sensor Data

K-means cluster algorithm was applied on the data obtained from Heartbeat sensor in the above experiment

```
In [4]: sensor_data = data['Beats'].values.reshape(-1, 1)

        # Apply K-means for anomaly detection
        n_clusters = 3  # Number of clusters for K-means
        kmeans = KMeans(n_clusters=n_clusters, random_state=0)
        kmeans.fit(sensor_data)

        # Assign each data point to a cluster
        cluster_labels = kmeans.predict(sensor_data)

        # Calculate the centroid distances for each data point
        centroid_distances = kmeans.transform(sensor_data)

        # Set a threshold to define anomalies
        threshold = np.percentile(centroid_distances[:, 1], 95)  # Adjust the percentile as needed

        # Identify anomalies
        anomalies = data.loc[centroid_distances[:, 1] > threshold]
        print(threshold)
        print(anomalies)

        77.47478991596634
                          Date&Time  Beats  Label
        26  [2024-02-24 15:35:14.282]      4      1
        27  [2024-02-24 15:35:15.296]      5      0
        32  [2024-02-24 15:35:20.314]      4      0
        33  [2024-02-24 15:35:21.325]      4      1
```

KNN was also applied on the data

```
In [8]: # Extract sensor data
        sensor_data = data['Beats'].values.reshape(-1, 1)

        # Apply KNN for anomaly detection
        n_neighbors =4 # Number of neighbors for KNN
        knn = NearestNeighbors(n_neighbors=n_neighbors)
        knn.fit(sensor_data)

        distances, indices = knn.kneighbors(sensor_data)
        k_distances = distances[:, -1]  # Distances to the k-th nearest neighbor

        # Set a threshold to define anomalies
        threshold = np.percentile(k_distances, 95)  # Adjust the percentile as needed

        # Identify anomalies
        anomalies = data[k_distances > threshold]


        # Print the detected anomalies
        print("Detected Anomalies:")
        print(anomalies)
```

Results were also obtained using statistical technique

```
plt.figure(figsize=(50,8))
plt.plot(data['Date&Time'], data['Beats'], color='blue')
plt.scatter(anomalies['Date&Time'], anomalies['Beats'], color='r',label='Anomalies')
plt.xlabel("Time")
plt.ylabel("Beats")
plt.xticks(rotation=45)
plt.title("HeartBeat  Data with Anomalies")
plt.legend()
plt.show()
```

The above mentioned techniques were also applied to data collected from Ulterasonic Sensor and Temperature Sensor. Statistical Test to check the performance of these anomaly detection techniques on Raspberry Pi and Arduino Uno was performed on the values of accuracy of the controller boards under study at 0.05 level of significance.

```
from scipy import stats

accuracy_raspberrypi = [0.98,0.99,0.95,0.955,0.99,0.98,0.94,0.97,0.99]
accuracy_aurdino = [0.92,0.89,0.94,0.089,0.88,0.87,0.90,0.86,0.87]
# Perform the paired t-test for accuracy
t_stat_accuracy, p_val_accuracy = stats.ttest_rel(accuracy_raspberrypi, accuracy_aurdin
                                                  )


print(f"Accuracy: t-statistic = {t_stat_accuracy}, p-value = {p_val_accuracy}")
```

```
Accuracy: t-statistic = -1.9273792363787432, p-value = 0.09008423359807434
```

Now Paired T test was used to compare the accuracy values between Raspberry Pi and Aurdino Uno to test the Null Hypothesis given below.

H_0 (Null Hypothesis) states that there is no significant difference in the performance of the anomaly detection models between the Raspberry Pi and Arduino Uno.

Results:

The results obtained in the experiments performed on RaspberyPi are given as under:

Anomaly Detection of sensors on RaspberyPi through Statistical Analysis using Z-Scores:

Results	Temperature and Humidity Sensor	LDR sensor	Ultrasonic Distance Sensor	Soil Moisture Sensor
Precision	1.0	0.75	1.0	0.1818
Recall	0.5	1.0	0.5454	1.0
F1	0.66666	0.8571	0.7058	0.30769
Accuracy	0.98	0.99	0.95	0.955

Anomaly Detection of sensors on RaspberyPi Through Cluster-Based Technique using K-Means:

Results	Temperature and Humidity Sensor	LDR sensor	Ultrasonic Distance Sensor	Soil Moisture Sensor
Precision	1.0	0.6	0.4	0. 333333
Recall	0.75	1.0	0.181818	1.0

F1	0.8571	0.74999	0.25	0.5
Accuracy	0.99	0.98	0.88	0.98

Anomaly Detection of sensors on RaspberyPi Through Nearest Neighborhood Technique using KNN:

Results	Temperature and Humidity Sensor	LDR sensor	Ultrasonic Distance Sensor	Soil moisture Sensor
Precision	1.0	0.6	1.0	0.25
Recall	0.75	1.0	0.4545	1.0
F1	0.8571	0.74999	0.625	0.4
Accuracy	0.99	0.98	0.94	0.97

The results obtained in the experiments performed on Arduino Uno are given as under:

Anomaly Detection of sensors on Arduino Uno through Statistical Analysis using Z-Scores:

Results	Temperature and Humidity Sensor	Heartbeat Sensor	Ultrasonic Distance Sensor
Precision	0.2692307692307692	0.6666666666666666	0.34615384615384615
Recall	0.4375	0.631578947368421	0.2571428571428571
F1	0.33333333333333337	0.6486486486486486	0.29508196721311475
Accuracy	0.8715596330275229	0.9437229437229437	0.8781869688385269

Anomaly Detection of sensors on Arduino Uno through Cluster-Based Technique using K-Means:

Results	Temperature and Humidity Sensor	Heartbeat Sensor	Ultrasonic Distance Sensor
Precision	0.125	0.5833333333333334	0.5384615384615384
Recall	0.0625	0.3684210526315789	0.2
F1	0.08333333333333333	0.4516129032258065	0.2916666666666667
Accuracy	0.8990825688073395	0.9264069264069265	0.9036827195467422

Anomaly Detection of sensors on Arduino Uno through Nearest Neighborhood Technique using KNN:

Results	Temperature and Humidity Sensor	Heartbeat Sensor	Ultrasonic Distance Sensor

Precision	0.2	0.25	0.16666666666666666
Recall	0.1111111111111111	0.15789473684210525	0.08571428571428572
F1	0.14285714285714285	0.1935483870967742	0.11320754716981132
Accuracy	0.8899082568807339	0.8917748917748918	0.8668555240793201

The results obtained in the study of performance of the selected anomaly detection techniques on Raspberry Pi and Arduino Uno are given below.

Values of Accuracy with Raspberry pi are :

accuracy_raspberrypi= [0.98, 0.99, 0.95, 0.955, 0.99, 0.98, 0.94, 0.97, 0.99]

Values of Accuracy with Aurdino are:

accuracy_aurdino = [0.92, 0.89, 0.94, 0.89, 0.88, 0.87, 0.90, 0.86, 0.87]

The p-value obtained in Paired T test on the accuracy values between Raspberry Pi and Aurdino Uno to test the Null Hypothesis was obtained as 0.09.

H_0 (Null Hypothesis) =there is statistically no significant difference in the performance of the anomaly detection models between the Raspberry Pi and Arduino Uno.

Discussion:

the anomaly detection results using different techniques (Statistical Analysis with Z-Scores, Cluster-Based Technique with K-Means, and Nearest Neighbor Technique with KNN) for each sensor (Temperature and Humidity Sensor (Dht11), LDR Sensor, Ultrasonic Distance Sensor, and Soil Moisture Sensor) provide valuable insights into the performance of these methods for anomaly detection.

1. Statistical Analysis with Z-Scores:

   - The Z-Score method shows promising results with high precision for most sensors, meaning that when an anomaly is detected, it is likely to be a true anomaly.
   - The recall values are moderate, suggesting that the Z-Score method can identify a good percentage of actual anomalies.
   - The overall F1 scores are reasonable, indicating a decent balance between precision and recall for this method.
   - However, further optimization may be needed to improve the recall for certain sensors.

2. Cluster-Based Technique with K-Means:

- The K-Means method exhibits good precision for most sensors, indicating accurate detection of anomalies within the identified clusters.
- The recall values are generally high, indicating that the K-Means method effectively captures actual anomalies for most sensors.
- The F1 scores show a good balance between precision and recall, making this method effective for anomaly detection.
- K-Means shows a strong overall performance and could be a preferred method for certain sensors.

3. Nearest Neighborhood Technique with KNN:

- The KNN method demonstrates varying performance across sensors, with high precision and recall for some sensors but lower values for others.
- The overall F1 scores are moderate, indicating a reasonably balanced performance for most sensors.
- KNN might require further fine-tuning and possibly feature engineering to improve its performance, particularly for certain sensors.

The work on anomaly detection was further carried out on Aurduino controller with number of sensors; Temperature and Humidity Sensor, Heartbeat Sensor and Ultrasonic Distance Sensor attached to it. The sensor data was collected and statistical techniques as discussed above were applied. The observation obtained in the experiment aligns with the earlier experiment.

In sum up, the most appropriate anomaly detection technique would depend on the specific requirements of the application, the characteristics of the sensors, and the importance of precision and recall trade-offs.

It is also essential to consider the class distribution in the dataset and the potential consequences of false positives and false negatives in the context of the real-world application. Fine-tuning the models (e.g. trying different distance metrics for KNN) or incorporating ensembled techniques may produce better results.

Here P value obtained on the test of accuracy of sensors attached on Raspberry Pi and Arduino Uno is greater than chosen significance level; therefore the null hypothesis is rejected, indicating that there is no significant difference in the performance of the anomaly detection models between the Raspberry Pi and Arduino Uno.

Conclusion:

From the study undertaken on the tracing out anomalies in the working of IoT sensors under various conditions, it is concluded hereunder.

Anomaly detection technique depends on the specific requirements of the application, the characteristics of the sensors and the importance of precision and recall trade-offs.

It is also essential to consider the class distribution in the dataset and the potential consequences of false positives and false negatives in the context of the real-world application.

There is no significant difference in the performance of the anomaly detection models between the Raspberry Pi and Arduino Uno.

4/4/24

Prof. Lalit Sen Sharma, PI, Research and Seed Grant
Department of Computer Sc. & IT, University of Jammu

Dean, Mathematical Sc.

4/4/24

Head, Computer Sc. & IT

Senior Professor of the Faculty

Senior Professor of the Department

Senior Associate Professor (By Rotation)