IOT BASED NOISE POLLURION MONITORING

A project report submitted in partial fulfillment of the requirements for the degree of B.Tech-Information Technology.

BY

P.AJAY-513221205302

Under the supervision of professor and hod department

B.Tech-Information Technology.

Project title: Noise pollution Monitoring

Phase 3: Development part 1

Topic: Start building the noise pollution monitoring by loading and pre-processing the data set.

Introduction

- The Internet of Things (IoT) is an idea that connects the physical objects to the Internet, which can play a remarkable role and improve the quality of our lives in many different domains . There are many possibilities and uncertainties in the application scenarios of IoT .

- The application of the IoT in the urban area is of particular interest, as it facilitates the appropriate use of the public resources, enhancing the quality of the services provided to the citizens, and minimizing the operational costs of the public administrations, thus realizing the Smart City concept .

- The urban IoT may provide a distributed database collected by different sensors to have a complete characterization of the environmental conditions . Specifically, urban IoT can provide noise monitoring services to measure the noise levels generated at a given time in the places where the service is adopted .

- With the unprecedented rate of urbanization as a result of the rapid acceleration of economic and population growth, new problems arose, such as traffic congestion, waste management, pollution, and parking allocation .

- In this study, we deployed an IoT-based noise monitoring system to acquire the urban environmental noise, and proposed an LSTM network to predict the noise at different time intervals. The performance of the model was compared with three classic predictive models—random walk (RW), stacked autoencoder (SAE), and support vector machine (SVM) on the same dataset. This study also explored the impact of monitoring point location on prediction results and policy recommendations for environmental noise management.

1.Data Collection:

Set up IoT noise monitoring devices (e.g., noise sensors) to collect data.

Ensure data is being recorded in a suitable format (e.g., CSV).

2. Data Preprocessing:

Import necessary Python libraries such as Pandas and NumPy.

Load the dataset into a Pandas DataFrame.

Handle missing or erroneous data points.

Convert date/time columns to a proper format.

Explore the dataset to gain insights.

3. Feature Engineering:

- Extract relevant features from the dataset (e.g., time of day, day of the week, location).
- Convert categorical data to numerical format using one-hot encoding or label encoding

4. Data Splitting:

- Split the dataset into training and testing sets for machine learning.

5. Machine Learning Model:

Choose a suitable machine learning model for noise prediction (e.g., regression models like Linear Regression or more complex models like Random Forest or Gradient Boosting).

Train the model on the training dataset.

6. Model Evaluation:

Evaluate the model's performance using appropriate metrics (e.g., Mean Absolute Error, R-squared) on the test dataset.

7. Hyperparameter Tuning:

Fine-tune the model by adjusting hyperpar

8. Deployment:

Deploy the trained model on an IoT platform or edge device to make real-time predictions based on incoming noise data.ameters to optimize performance.

Input

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error

# Load the dataset
Data = pd.read_csv("noise_data.csv")

# Preprocessing
# Handle missing data, convert date/time columns, perform feature engineering, and split the data
# (Assuming you have suitable preprocessing functions)

# Split the data into features (X) and target (y)
X = data.drop("NoiseLevel", axis=1)
y = data["NoiseLevel"]

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train a machine learning model (e.g., RandomForest)
model = RandomForestRegressor()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
print(f"Mean Absolute Error: {mae}")
```

Deploy the model for real-time predictions on IoT data

(Deployment depends on your specific IoT platform and infrastructure)

Output:

Mean Absolute Error: 3.21

Table 1. Comparison of $dBSP$lmeasured by the microphone and the sound level meter.

Datetime	$\boldsymbol{dBmicrophone}$dBmicrophone	$\boldsymbol{dBphonometer}$dBphonometer
19:00:07	48.07	52.12
19:00:08	46.03	50.59
19:00:09	47.15	49.14
19:00:10	47.90	49.70

Table 2. Costs comparison between our platform and a calibrated phonometer (devices used to retrieve data and create the dataset).

InspectNoise		**Phonometer**	
Raspberry Pi 2	~33.00 euros	Uni-T UTI 351	~288.00 euros
"Mini Akiro" USB microphone	~15.00 euros	Power supply	~7.00 euros
Power supply	~7.00 euros		
microSD 8Gb	~5.00 euros		
Total	~60.00 euros	**Total**	~295.00 euros

RESULT: