Environmental Sound Classification and Gas Detection using IoT with Deep Learning

Naveen Mishra[1], Priya Mishra[2], Balaji Boopal[3]

Vellore Institute of Technology, Vellore, Tamil Nadu, India

Abstract. The IoT device is designed to assess the air quality, temperature, and humidity in an environment, while also detecting gases using the MQ6 sensor to identify LPG gas commonly used in households. For air quality, the device utilizes the MQ135 sensor, and the DHT11 sensor is employed to measure temperature and humidity. The Arduino UNO serves as the microcontroller, collecting sensor data and transmitting it to both the ThingSpeak cloud and a Convolutional Neural Network (CNN) model. By connecting to ThingSpeak, the device ensures that environmental data is stored and visualized for easy monitoring. In the event of the sensor detecting dangerous gases, the device activates an alarm, and the device owner receives a notification promptly through the IFTTT API integration. To enhance environmental awareness, a microphone is incorporated into the device, capturing audio recordings. These recordings are then processed by a CNN-based deep learning model, which interprets the sounds to provide insights into ongoing activities in the environment. The model employs techniques such as spectrogram generation, Convolution2D, and MaxPooling2D within a Sequential Model to achieve high accuracy.

Keywords: MQ6 MQ135 DHT11 sensors, IFTTT, CNN Model, Convolution2D, MaxPooling2D, API

1 Introduction

1.1 IoT based Air-Quality and Gas Detecting

Air pollution has become a widespread issue globally, particularly in urban areas where it poses a significant health risk. Factors such as the prevalence of petrol and diesel vehicles, industrial zones surrounding major cities, and the evident impacts of climate change contribute to the complexity of the problem. Governments worldwide are taking measures to address this crisis, with many European countries aiming to replace traditional vehicles with electric ones by 2030, and India setting a target of 2025 for a similar transition.

In the recent month, Delhi has experienced a slight improvement in air quality levels. However, there are still concerning reports of elevated levels of PM 2.5 – fine particles that can deeply penetrate the lungs – in various parts of the city and the National Capital

Region (NCR). For instance, as of the latest data at 9:34 am, Faridabad and Ghaziabad recorded PM 2.5 levels at 278 and 275, respectively. The air pollution in Delhi, known as one of the most polluted cities globally, sparks heated public debates, especially during the onset of winter, garnering international attention.

Continuous exposure to areas with poor air quality is a pressing public health issue, leading to an estimated 2.5 million premature deaths worldwide each year. Notably, 1.5 million of these deaths are attributed to indoor air pollution, emphasizing the significant health risks associated with poor indoor air quality for over half of the global population. The impact of pollution on public health is linked to industrial development and affects both developed and developing countries, contributing to premature deaths.

Reports indicate a concerning frequency of over 1,500 LPG accidents daily in India, translating to a similar number of deaths, including young individuals. Such accidents often involve neighboring communities, emphasizing the urgent need for technological interventions to prevent these incidents. The Internet of Things (IoT) emerges as a rapidly advancing technology with transformative potential, particularly in the automotive industry, serving as a foundational element for the development of Industrial 4.0.

1.2 Environmental Sound Classification using CNN based learning model

The primary objective of research in machine listening, particularly within the expansive domain of computational auditory scene classification (CASA), is to empower machines to comprehend their surroundings through the analysis of sound [1]. Machine listening systems form part of a broader research field that intersects machine learning, robotics, and artificial intelligence, mirroring the processing tasks executed by the human auditory system.

Acoustic Scene Classification (ASC), which involves assigning a semantic label to an audio stream indicating the environment in which it originated, constitutes a significant aspect of this study. Within the ASC literature, a distinction exists between psychoacoustic/psychological studies focused on understanding human cognitive processes enabling acoustic scene comprehension and computational algorithms attempting to autonomously execute this task using signal processing and machine learning techniques. The investigation of soundscapes as auditory counterparts to landscapes is termed soundscape cognition in perceptual studies, while computational research is referred to as computational auditory scene recognition, specifically applied to environmental sounds. Although the emphasis in this paper leans towards computer-based research, insights from human listening tests may be provided for comparison.

The evolution of ASC's work aligns with various interconnected scientific challenges. Techniques for classifying noise sources, for instance, have found application in noise monitoring systems and enhancing the efficacy of speech processing algorithms. Algorithms for sound source recognition closely relate to event detection and classification algorithms, as they aim to identify the sources of acoustic events in recordings. The latter methods play roles in surveillance systems, geriatric assistance, and speech analysis through the segmentation of acoustic scenes, focusing on locating and labeling temporal zones containing singular events of a specific class. Additionally, semantic analysis

algorithms for audio streams, relying on the recognition or clustering of sound events, have been employed for personal archiving and audio segmentation and retrieval. The distinction between event detection and ASC may blur in certain cases, such as multimedia indexing and retrieval systems, where identifying events like a baseball batter hitting a home run also characterizes the overall environment of a baseball game. Conversely, ASC can enhance sound event detection by providing preliminary information about the likelihood of specific events. This discussion will focus solely on systems aimed at modeling complex physical environments with numerous occurrences to maintain the scope of this work within reasonable bounds.

1.2 Proteus Schematic for IoT Device

The schematic circuit for the entire system is designed using Proteus 8 software. Data from sensors is transmitted to the ThingSpeak cloud through the Arduino UNO microcontroller. While the software simulation allows for user-defined inputs to the sensors, the device is fully functional in the physical world. It swiftly transfers data to the cloud, visualizing it within 2-3 minutes, and triggers remote notifications to the user via Wi-Fi. The resulting device is compact, resembling the size of a smartphone screen, making it portable and convenient for deployment.

The MQ135 sensor, capable of detecting gases like NH3, NOx, Alcohol, Benzene, and CO2, is ideal for monitoring environmental air quality [2]. Beyond air-quality monitoring, other gas sensors, exemplified by the MQ6 LPG Detection sensor in this project, can identify the presence of explosive or hazardous gases. This feature ensures the device is applicable for securing households from potential LPG cylinder blasts. Moreover, customization allows users to adapt the device to detect specific gases relevant to their industry. Additionally, the DHT11 sensor is integrated to inform the user about the humidity and temperature of the environment. The Arduino is connected to an LCD, displaying air quality in parts per million (PPM), along with humidity and temperature readings. A Buzzer, linked to the Arduino, provides alerts for LPG detection. To facilitate data transfer from Arduino to the ThingSpeak cloud, Virtual Serial Port Emulator (VSPE) and a Python Script are employed instead of NodeMCU.

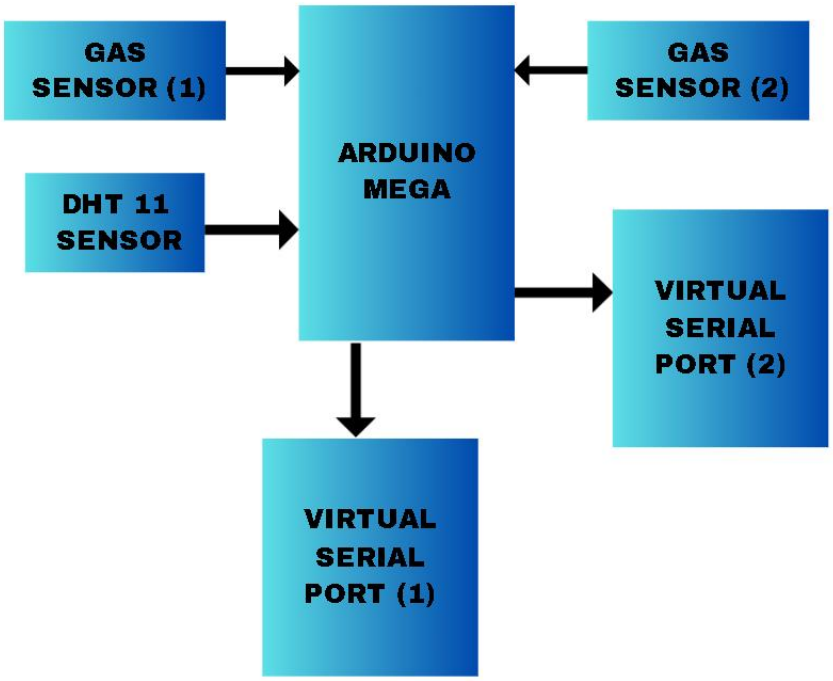

Fig. 1. Schematic of the Device.

1.3 Integration ThingSpeak cloud with Arduino and alerting user with the help of IFTTT

Within ThingSpeak, there are two applications available for customizing triggers: ThingHTTP and React. React comes into play when the Arduino detects LPG, initiating the trigger ticket and interfacing with ThingHTTP. This, in turn, activates two IFTTT (If This Then That) applets. The first applet is designed to send an email to the user upon LPG detection, while the second applet delivers a notification directly to the user's mobile device through the IFTTT application. Once React initiates the ThingHTTP process, ThingHTTP calls the API for the IFTTT applet, prompting IFTTT to send the notification to the user. Remarkably, this entire process is completed within a swift 3-minute timeframe from the Arduino detecting the gas. This showcases the robust and efficient nature of the product in promptly securing the environment from potential harm.

1.4 CNN Based ASC Deep Learning Model

The goal is to recognize various sound environments or situations in the surroundings. For instance, if a dog barks, the current system is designed to classify the sound pattern specifically associated with a dog's bark. Acoustic scene classification aims to assign a test recording to a predefined category that characterizes the recording environment. The dataset utilized comprises a total of 2000 environmental audio recordings spread across various classes and subclasses. It encompasses five primary classes representing different categories of sounds commonly heard in everyday environments, with each class further divided into ten subclasses, each representing a specific type of sound derived from its respective class. Notably, each subclass consists of 40 distinct sound recordings, resulting in a total of 2000 sound recordings used for training the model.

Table 1. Dataset of the environmental audio files where columns are classes and rows are subclasses.

	Animals	Natural soundscapes & water sounds	Human/ non-speech sounds	Interior/domestic sounds	Exterior/urban noises
0	Dog	Rain	Crying baby	Door knock	Helicopter
1	Rooster	Sea waves	Sneezing	Mouse click	Chain saw
2	Pig	Crackling fire	Clapping	Keyboard typing	Siren
3	Cow	Crickets	Breathing	Door,wood creaks	Car horn
4	Frog	Chirping birds	Coughing	Can opening	Engine
5	Cat	Water drops	Footsteps	Washing machine	Train
6	Hen	Wind	Laughing	Vacuum cleaner	Church bells
7	Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
8	Sheep	Toilet flush	Snoring	Clock tick	Crackers
9	Crow	Thunderstorm	Drinking/sipping	Glass breaking	Hand saw

Convolutional Neural Networks (CNNs) have emerged as the predominant choice for tasks like acoustic scene classification. The most effective CNNs primarily utilize audio spectrograms as input, drawing architectural inspiration mainly from computer vision principles [3]. Consequently, all audio files are transformed into spectrograms using the 'librosa' Python library. This proposal examines the capacity of convolutional neural networks to classify concise audio recordings of environmental sounds.

Upon training the model, it was observed that the existing dataset lacked adequacy for achieving satisfactory accuracy. To address this, Data Augmentation is employed, incorporating white noise into copies of the original dataset, resulting in a total of 4000 training examples. Subsequently, 10% of the data is reserved for testing, and another 10% for model evaluation.

Following augmentation, four Convolution2D layers are introduced to enhance accuracy, complemented by the addition of MaxPooling2D to mitigate overfitting. Prior to model compilation, the data is flattened from 2D to 1D using 'flatten'. The 'dense' layer is utilized to connect all layers, and 'dropout' is applied to randomly deactivate selected neurons after each epoch, ensuring ongoing improvement in model accuracy.

Finally, the Sequential model is compiled using the 'adam' optimizer, initiating the model training phase with 80% of the dataset and setting the epochs value to 30.

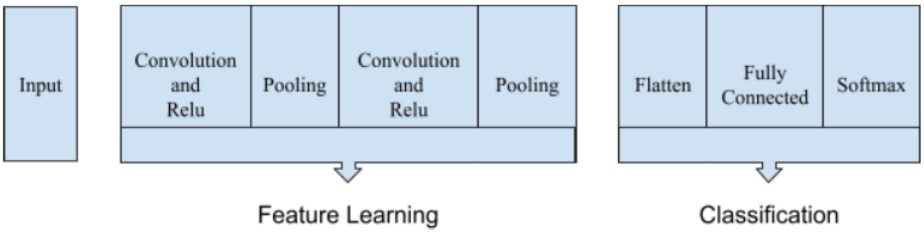

Fig. 6. Flowchart of the ASC Model.

Table 2. Summary of the ASC Model.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 214, 64)      640

 max_pooling2d (MaxPooling2D  (None, 63, 107, 64)      0
 )

 conv2d_1 (Conv2D)           (None, 61, 105, 128)      73856

 max_pooling2d_1 (MaxPooling  (None, 30, 52, 128)      0
 2D)

 conv2d_2 (Conv2D)           (None, 28, 50, 256)       295168

 max_pooling2d_2 (MaxPooling  (None, 14, 25, 256)      0
 2D)

 conv2d_3 (Conv2D)           (None, 12, 23, 256)       590080

 max_pooling2d_3 (MaxPooling  (None, 6, 11, 256)       0
 2D)

 flatten (Flatten)           (None, 16896)             0

 dense (Dense)               (None, 256)               4325632

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 50)                12850

=================================================================
Total params: 5,298,226
Trainable params: 5,298,226
Non-trainable params: 0
```

1.5 Collaborating IoT device with the ASC Model

In the preceding sections, we discussed the creation of the IoT device and the audio scene classification model as distinct entities. In the final phase, a web application can be developed to bring these components together. This integration allows users to send audio recordings of their environment from the IoT device to the web

application at regular intervals, facilitated by a Python script scheduled to run every 5 minutes. Through this application, users can receive notifications about ongoing activities in the environment using the Audio Scene Classification (ASC) Model. Additionally, the web application provides information on air quality, temperature, and humidity obtained through Arduino sensors. Moreover, it has the capability to trigger alarms and notifications in the event of detecting any hazardous gas, such as LPG in the context of this project.

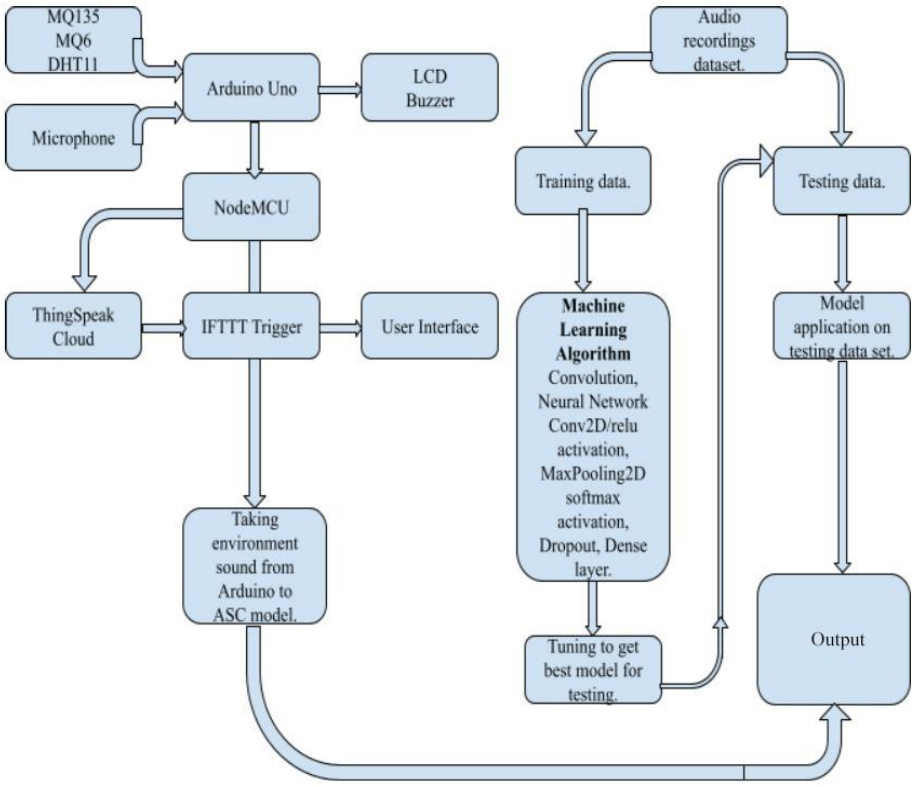

Fig. 2. Project Demonstration after collaborating IoT device with ASC Model.

2 Results and Discussions

In the Proteus schematic, all the necessary library files were incorporated, along with the inclusion of Arduino Code. The output screen utilizes the Virtual Terminal, where one virtual terminal displays the air quality, temperature, and humidity of the environment using data from MQ135 and DHT11 sensors. Meanwhile, the other terminal indicates the detection of LPG, represented in binary where 0 signifies 'not detected' and 1 denotes 'detected'. These virtual terminals are exclusively linked to the Virtual Serial Port Emulator (VSPE) ports. The next step involves transmitting the Arduino outputs to a Python Script through these virtual ports.

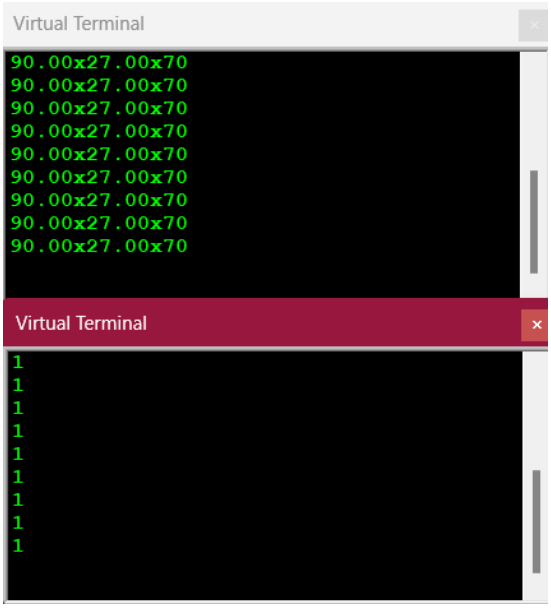

Fig. 3. Circuit Simulation showing air-quality, temperature, humidity and hazardous gas detection.

The Python Script is provided with the Write API keys of the ThingSpeak channel. The script is then scheduled to run every 20 seconds, enabling it to transmit the Arduino outputs to the ThingSpeak channel. This setup allows the user to access real-time visualizations of their environment at any given moment through the ThingSpeak channel.

```
<------------------------------------------->
Established serial connection to Arduino
<http.client.HTTPResponse object at 0x000002C28290A430>
<http.client.HTTPResponse object at 0x000002C28290A400>
<http.client.HTTPResponse object at 0x000002C28290A430>
Collected readings from Arduino: [90.0, 27.0, 70.0]
<http.client.HTTPResponse object at 0x000002C28290A4C0>
Collected readings from Arduino: [0.0]
Connection closed
<------------------------------------------->
```

Fig. 4. Python Script sending outputs from the Arduino to ThingSpeak channel.

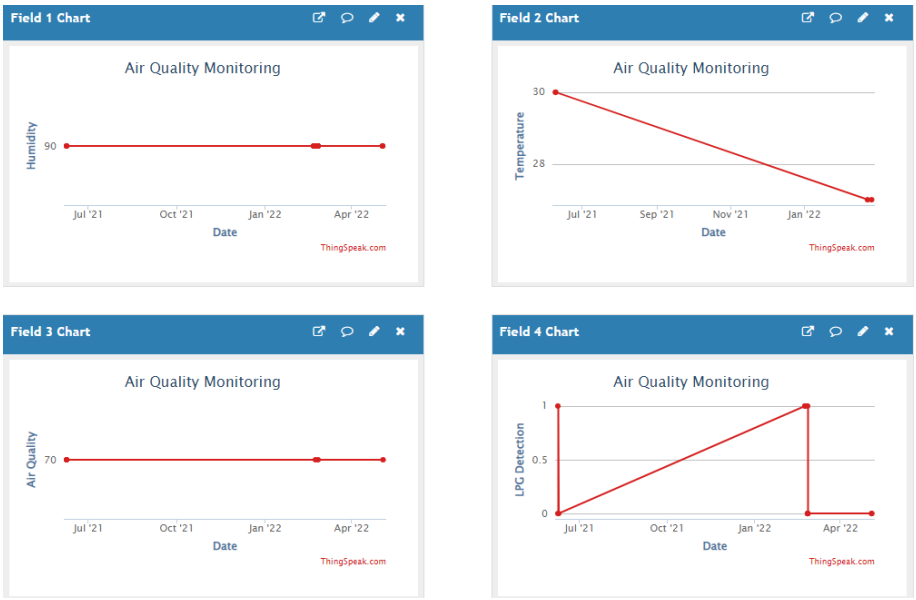

Fig. 5. ThingSpeak visualizing the environment statistics.

As a result, the IoT device is fully operational, efficiently updating its cloud in real-time, and promptly alerting the user within 2-3 minutes of gas detection. Shifting focus to the Audio Scene Classification (ASC) model, after successfully compiling and training it, an impressive accuracy of 96% was achieved. This surpasses the accuracy of all previously encountered models in the literature survey. The notable improvement is attributed to the augmentation of the dataset by introducing white noise, indicating that this approach significantly contributed to enhancing the model's accuracy and overall optimization compared to previous iterations.

.

```
Epoch 16/30
108/108 [==============================] - 16s 150ms/step - loss: 0.6647 - accuracy: 0.8370
Epoch 17/30
108/108 [==============================] - 16s 150ms/step - loss: 0.3753 - accuracy: 0.8883
Epoch 18/30
108/108 [==============================] - 16s 150ms/step - loss: 0.3906 - accuracy: 0.8932
Epoch 19/30
108/108 [==============================] - 16s 151ms/step - loss: 0.3965 - accuracy: 0.8972
Epoch 20/30
108/108 [==============================] - 16s 149ms/step - loss: 0.3316 - accuracy: 0.9157
Epoch 21/30
108/108 [==============================] - 16s 150ms/step - loss: 0.2823 - accuracy: 0.9278
Epoch 22/30
108/108 [==============================] - 16s 150ms/step - loss: 0.2244 - accuracy: 0.9364
Epoch 23/30
108/108 [==============================] - 16s 150ms/step - loss: 0.2527 - accuracy: 0.9429
Epoch 24/30
108/108 [==============================] - 16s 149ms/step - loss: 0.3478 - accuracy: 0.9207
Epoch 25/30
108/108 [==============================] - 16s 150ms/step - loss: 0.2069 - accuracy: 0.9414
Epoch 26/30
108/108 [==============================] - 16s 150ms/step - loss: 0.2176 - accuracy: 0.9460
Epoch 27/30
108/108 [==============================] - 16s 150ms/step - loss: 0.3399 - accuracy: 0.9176
Epoch 28/30
108/108 [==============================] - 16s 150ms/step - loss: 0.2082 - accuracy: 0.9404
Epoch 29/30
108/108 [==============================] - 16s 149ms/step - loss: 0.2558 - accuracy: 0.9417
Epoch 30/30
108/108 [==============================] - 16s 149ms/step - loss: 0.1682 - accuracy: 0.9633
```

Fig. 6. Accuracy observation at the end of 30 epochs.

Following the model training phase, testing and validation procedures were carried out using the remaining 20% of the dataset.

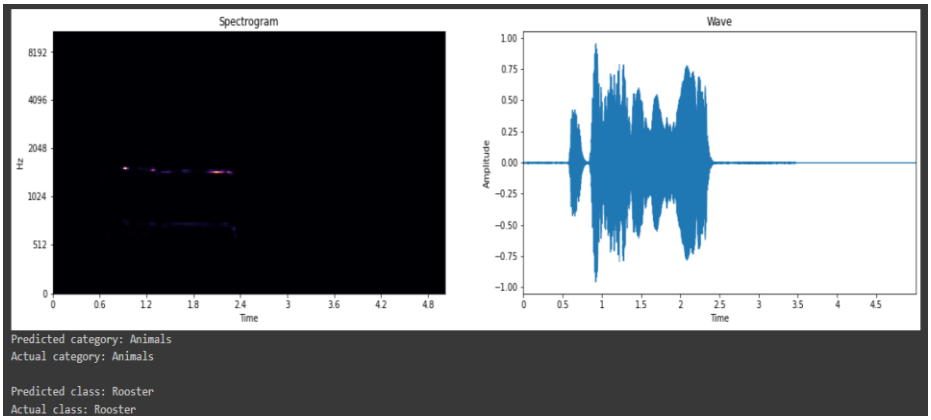

Fig. 7. Output Prediction of the model from a random input from the testing set.

Therefore, the model is capable of identifying all environmental sounds within the classes present in the utilized dataset. It's acknowledged that encompassing every sound

in the world within a single dataset is impractical. However, the dataset comprehensively covers a wide range of everyday environmental sounds, enabling the model to effectively detect and classify such sounds.

3 Conclusion

This device utilizes an Arduino microcontroller and IoT technology to identify air pollution in the environment, aiming to improve overall air quality. A gas leakage detection system is integrated to provide timely alerts and notifications to the user's mobile device, potentially saving lives. The incorporation of Internet of Things (IoT) technology enhances the monitoring of various environmental aspects, with a focus on air quality. The project relies on MQ135 and MQ6 gas sensors for assessing air quality, with Arduino serving as the central component. A Wi-Fi module establishes the connection to the internet, and an LCD screen offers visual output. ThingSpeak is employed for monitoring, analyzing, and displaying data, while the IFTTT application is used for user notifications and alerts.

The study also delves into exploring an alternative deep Convolutional Neural Network (CNN) architecture configuration, specifically tailored for distinct maximum receptive fields across audio spectrograms. This approach aims to enhance the design of deep CNNs for acoustic classification tasks and adapt successful CNNs from other domains, particularly image recognition, to acoustic scene classification.

The model implemented in this project employs a two-dimensional CNN with max-pooling 2D, utilizing a total dataset of 4000 samples. To address the initial data insufficiency, white noise is added to the existing 2000 datasets. Augmentation is then applied, converting audio files to spectrogram files to obtain spectrogram outputs. Training is conducted using TensorFlow, utilizing all available data with 30 epochs, resulting in an impressive accuracy of 96%.

This technology holds the potential to extend beyond individual devices, enabling the installation of air quality sensors throughout a city. This broader application could facilitate the mapping of air quality and the establishment of a website where individuals can track pollution levels in their respective areas.

References

1. D. Barchiesi, D. Giannoulis, D. Stowell, M. D. Plumbley, “Acoustic Scene Classification,” cited as arXiv:1411.3712 (cs), IEEE Transl. Signal Processing Magazine 32(3) (May 2015) 16-34.
2. K. Koutini, H. E. Zadeh, G. Widmer, “Receptive-field-regularized CNN variants for acoustic scene classification,” cited as asXiv:1909.02859 [eess.AS] accepted at DCASE Workshop 2019.

3. H. N. Shah, Z. Khan, A. A. Merchant, M. Moghal, A. Shaikh, P. Rane, “IOT Based Air Pollution Monitoring System,” International Journal of Scientific & Engineering Research Volume 9, Issue 2, February-2018 ISSN 2229-5518.
4. A. Varma, Prabhakar S., K. Jayavel, “Gas Leakage Detection and Smart Alerting and prediction using IoT,” IEEE 2017 2nd International Conference on Computing and Communications Technologies (ICCCT) 327-333.
5. K. Kumar, Hemanth, Sabbani, “Smart Gas Level Monitoring, Booking & Gas Leakage Detector over IoT,” IEEE 7th International Advance Computing Conference (IACC) 330–332.
6. Y. Lee, W. Hsiao, C. Huang, S. T. Chou, “An integrated cloud-based smart home management system with community hierarchy,” IEEE Transactions on Consumer Electronics, 62(1), 1–9.
7. J. Joshi,V. Rajapriya, S. R. Rahul, P. Kumar, S. Polepally, R. Samineni, D. G. K. Tej, “Performance enhancement and IoT based monitoring for smart home,” IEEE 2017 International Conference on Information Networking (ICOIN) 468–473.
8. G. Roma, W. Nogueira, P. Herrera, “Recurrence quantification analysis features for auditory scene classification,” IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events, 2013.
9. D. Wang, G. Brown, “Computational Auditory Scene Analysis: Principles, Algorithms, and Applications,” Wiley, 2006.
10. Y. Sakashita, M. Aono, “Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divi.” DCASE2018 Challenge, 2018.
11. K. J. Piczak, “ESC: dataset for environmental sound classification,” in Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM ’15, Brisbane, Australia, October 26 - 30, 2015.
12. W. Luo, Y. Li, R. Urtasun, R. Zemel, “Understanding the Effective Receptive Field in Deep Convolutional Neural Networks,” in Advances in Neural Information Processing Systems 29, 2016, pp. 4898–4906.
13. C. M. Bishop, “Pattern Recognition and Machine Learning,” (Springer, Berlin, 2006).
14. S. Chakrabarty, E. A. Habets, “Broadband DOA estimation using convolutional neural networks trained with noise signals,” in IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, IEEE (2017), pp. 136–140.