Sensor applications

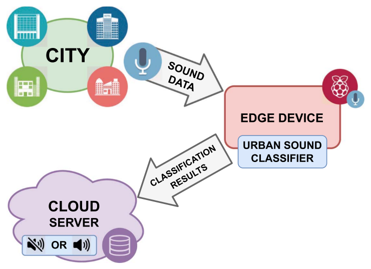
Edge-Based Data Sensing and Processing Platform for Urban Noise Classification

Marc Jayson Baucas* and Petros Spachos**

School of Engineering, University of Guelph, Guelph, ON N1G2W1, Canada
**Member, IEEE*
***Senior Member, IEEE*

Manuscript received 29 February 2024; accepted 3 April 2024. Date of publication 22 April 2024; date of current version 30 April 2024.

Abstract—The increasing population in urban areas contributes to noise pollution. So, cities look to locate areas with high noise levels to regulate them and improve urban well-being. However, dispatching personnel for noise data collection is time-consuming and expensive. Therefore, we propose a low-cost Internet of Things (IoT)-based urban noise classification platform to address noise collection and processing challenges in urban environments. We designed a prototype sound-sensing setup consisting of an STM32 NUCLEO-64 board with an attached X-NUCLEO-CCA02M1 expansion board as the digital micro-electrical-mechanical system microphone connected to a Raspberry Pi for collecting and classifying urban sound. Then, it sends the results via WiFi to the cloud server for noise analysis. We examined our design's feasibility with experiments evaluating its classification accuracy and power consumption. We further examined its latency when processing is at the edge or cloud. The experimental results suggest our platform's potential in noise analysis for urban environments.



Index Terms—Sensor applications, cloud computing, edge computing, Internet of Things (IoT), multilayer perceptron (MLP), neural network, remote sensing, smart city applications, sound sensing, stochastic gradient descent (SGD) optimizer, urban sensing.

I. INTRODUCTION

As the population grows in cities, so does the introduced noise pollution from several sound sources, such as traffic, construction, subways, and general industrial activities. It poses a significant concern in urban areas due to its socioeconomic disruptiveness. Environmental noise pollution reduces the quality of living for residents, causing them to be dissatisfied with their living conditions [1]. A goal for many smart city applications is to improve the quality of life of its inhabitants. The monitoring of noise pollution is among their priorities [2]. The challenge is to efficiently and timely determine the sound source and location to effectively assist each area in reducing noise pollution.

Urban sound classification is a widely used recognition model that categorizes sounds under specific characteristics. One of its applications is to identify noise sources for isolation and mitigation. However, an efficient sound classification system needs several devices to collect sufficient data and a capable medium to deploy the classification model. Utilizing Internet of Things (IoT) devices with sound-collecting capabilities and establishing a wireless IoT system is a promising approach for smart cities and urban areas [3], [4]. Also, small-size, low-power, and easy-to-deploy IoT devices can be cost-efficient sound-collecting devices in these places, as low-latency wireless communication uses a stable medium for large-scale data collection and transmission [5]. Improving system latency is possible if the processing is at the edge instead of a central server.

We propose a platform that combines the benefits of IoT systems and edge computing in synergy with a noise classification model, enabling the remote collection of sound data within various urban areas. The proposed design is flexible to differentiate noise sources, providing a feasible approach for smart cities to address growing concerns about noise pollution. We investigate its accuracy and energy efficiency and examine the classification performance when processing is at the edge and in the cloud.

Corresponding author: Petros Spachos (e-mail: petros@uoguelph.ca).
Associate Editor: W. Pu.
Digital Object Identifier 10.1109/LSENS.2024.3392163

II. NOISE CLASSIFICATION IN IOT

Recently, extracting urban sound information from smart cities has been made possible through IoT-based systems [5]. Its collection has enabled monitoring and regulating unwanted sources, such as environmental noise. As a result, many IoT-based designs try to address the noise-related urban sound collection challenge. Meng et al. [6] presented a smartphone-enabled IoT system for vibration and noise monitoring in urban rail transit areas. They use smartphones to process sensing data and evaluate vibration and noise impact to improve railway management and maintenance. Rauniyar et al. [7] presented a cloud-enabled Nautilus platform for real-time noise and exhaust emission monitoring for sustainable and intelligent monitoring systems. Both works identify the need to mitigate environmental noise within populated areas, such as transit and railways. Other implementations also focus on a broader aspect of urban computing but still touch on noise pollution monitoring. Bine et al. [8] presented an approach using drones to leverage environmental data collection. They maximize the mobility of these devices to extend the scope of the monitoring systems for better investigation and isolation of troubled sectors.

These designs present merit in noise monitoring applications utilizing different technologies, such as smartphones, drones, and other large-scale custom remote sensing designs. Our approach differs as we propose a platform design aiming for a more efficient network, transmitting data with lower latency. We focus on reducing energy costs using more cost-effective substitutes. Also, we focus on exploring the potential of edge devices in optimizing noise monitoring application designs. We have previously presented research on the merits of IoT-based urban sound classification with similar platform designs in [3] and [4]. However, this current work differs as we focus more on the noise analysis aspect of urban sound classification and present an updated neural network architecture with training and validation accuracy results.

III. PROPOSED DESIGN

We present an overview of the design, followed by the classifier and dataset selection and the design's data flow and components.

A. Overview

We propose an IoT-based platform as a low-cost approach for urban noise classification within smart cities. The flow of operations starts with each end device collecting sound data from the urban environment. Next, it classifies it through an urban sound classifier for noise levels through a neural network. Then, the end device sends the data to the server. Finally, the server analyzes and stores the results for proper utilization and further noise-regulating applications. We aim to emphasize the strengths of IoT systems in partnership with data sensing and classification through the proposed design.

B. Classifier and Dataset

We used an UrbanSound8K dataset and an Urban Sound classifier for testing and experimentation [9]. We can implement the noise classifying application using a binary model since we only need to classify if the sound is noise. However, it will limit our platform in showcasing its full potential. We chose a multilabel classification model because we wanted a more complex design to present our platform's capability to cater to larger datasets. Also, moving the noise analysis to other parts of the network shows our platform's modularity in adapting and working around different classifiers, specific or general, and creating more application-specific implementations.

The UrbanSound8K dataset uses different urban-related sounds from field recordings uploaded to www.freesound.com. The types of sound files range from mechanical (i.e., drilling, air conditioning, and gunshot) to natural (dog barking, children playing, and people singing). For preprocessing the data, we use Mel-Scale Frequency Cepstral Coefficients [10] due to its efficiency in extracting sound features.

The UrbanSound8K dataset labels each sound clip based on the type of sound. We categorized each label as either noise or regular sound to simulate how the server can analyze the classification results and use them for noise-identifying applications, turning the urban sound classifier into an urban noise classifier. This addition shows the platform's adaptability and modularity in utilizing datasets and neural networks to cater to specific sound-sensing applications. The noise categorization of the different types of sounds from this dataset is given in Table 1.

C. Data Flow and Components

The process starts with the sound collection from the end device. In this case, it is a sound-sensing setup consisting of a Raspberry Pi 3B and a digital microphone. We chose a Pi because they are easy to program, modular, and reconfigurable for rapid prototyping.

Table 1. Noise Categorization of UrbanSound8K Dataset

Sound type	Category	Number of files
Air conditioner	Regular sound	1000
Car horn	Noise	429
Children playing	Regular sound	1000
Dog barking	Regular sound	1000
Drilling	Noise	1000
Engine idling	Regular sound	1000
Gunshot	Noise	374
Jackhammer	Noise	1000
Police siren	Noise	929
Street music	Regular sound	1000

Table 2. Hyperparameters of SGD Optimizer

Parameter	Value
weight_decay	None
learning_rate	0.001
momentum	0.0
nesterov	False

Attached to each Pi is an STM32 NUCLEO-64 board with an attached X-NUCLEO-CCA02M1 expansion board to serve as a digital micro-electrical-mechanical system microphone for sound collection. Another advantage of the Pi is its compatibility with many add-ons and peripherals. The Pi is programmed to enable each microphone to record sound for a specified time. The urban sound classifier within the Pi as a Python script will then take the recorded sound file and classify it. During this process, the file will undergo feature extraction and categorization. Then, the Pi sends the results via WiFi to the server for storage and further analysis as either noise or regular sound, fulfilling its intended urban noise classification application. The server is a generic desktop computer running a Python script that receives the incoming data from the Pis via TCP/IP socket communication.

IV. TESTS AND EVALUATION

We present the conducted tests and the metrics used to evaluate the platform's feasibility.

A. Accuracy Verification

We first verified the accuracy of our sound classifier. We designed it as a multilayer perceptron (MLP) neural network using a sequential model from the Tensorflow and Keras libraries. A diagram illustrating this architecture and the sizes of each layer is in Fig. 1. This neural network uses a stochastic gradient descent (SGD) optimizer and a categorical cross-entropy loss function. A summary of the hyperparameters of our model's optimizer is in Table 2. We selected these options to keep the model design simple while maximizing its accuracy and staying within the processor constraints of the Pi.

As for the UrbanSound8K dataset, we used a 70–30 training and validating split. The graph showing the accuracy results of training the model is in Fig. 2. According to the results, the classifier showed signs of slight underfitting earlier in the training process. Using 100 epochs to train the model was ideal because the underfitting fixes as it reaches 30 and accuracy stabilizes around 85. The resulting training and validation accuracy is 95.39% and 89.95%, respectively. These results are reasonable as we attempt to model a complex dataset with multiple labels and 2-D features using a simple neural network

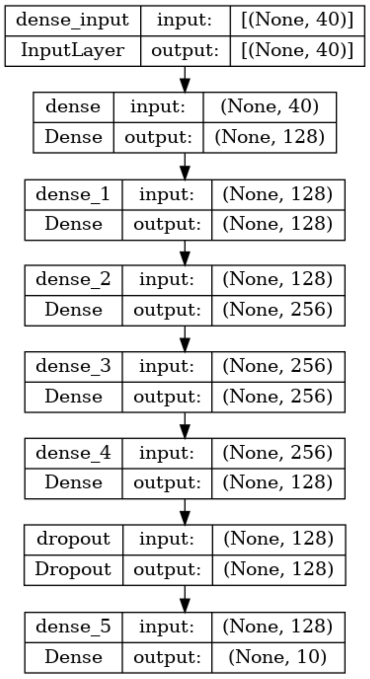

dense_input input: [(None, 40)]
InputLayer output: [(None, 40)]
dense input: (None, 40)
Dense output: (None, 128)
dense_1 input: (None, 128)
Dense output: (None, 128)
dense_2 input: (None, 128)
Dense output: (None, 256)
dense_3 input: (None, 256)
Dense output: (None, 256)
dense_4 input: (None, 256)
Dense output: (None, 128)
dropout input: (None, 128)
Dropout output: (None, 128)
dense_5 input: (None, 128)
Dense output: (None, 10)

Fig. 1. MLP architecture for urban sound classification.

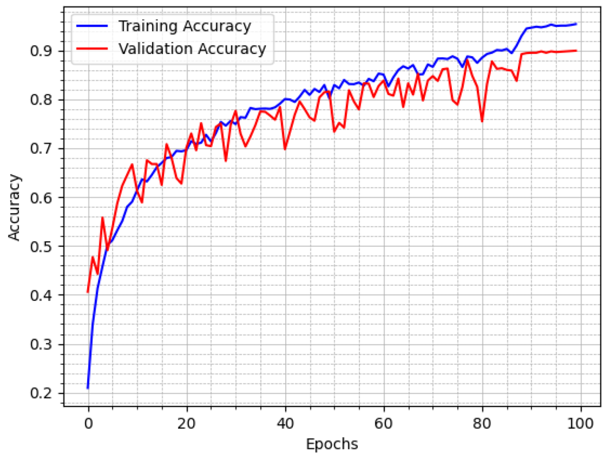


Fig. 2. Training and validation accuracy plot of sound classifier yielding a testing accuracy of 89.95%.

design to avoid overwhelming the Pi. Our approach remains relatively accurate as the validating accuracy stays close to the training results.

B. Power Consumption Comparison

We conducted the power consumption and latency experiments with two configurations. The first setup is our proposed platform. First, it executes the urban sound classification locally within the Pi. Then, it sends the results to the server for noise level categorization. The second setup is the centralized configuration. Initially, it sends the sound data to the server. The server runs it through the urban sound classifier to determine its sound type. Then, it categorizes its noise level based on the results. The difference between the two configurations is the work conducted by the end device and the cloud. The first, our proposed platform, will have most of the processing within the Pi as it will run the classifier. This arrangement delegates the complex process of processing the sound data and running Tensorflow to classify

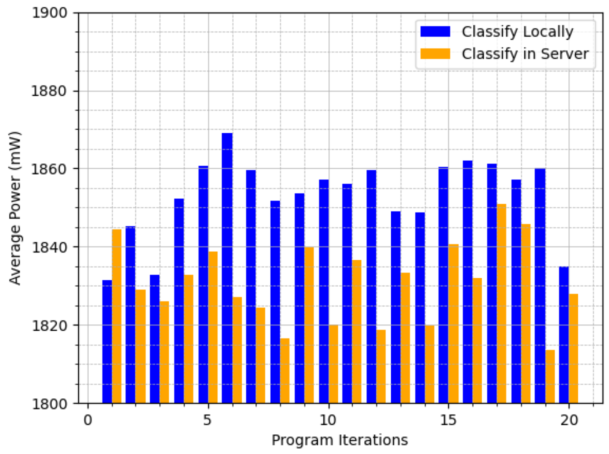


Fig. 3. Power consumption of configurations at 20 iterations.

it within the Pi. The second has all the processing allocated to the cloud server. Aside from collecting and transmitting the sound data, all the application's complexity is in the cloud. Each setup was tested by measuring the power consumption of the Pi as it executes each configuration. We used a monsoon high voltage power monitor to measure it in milliWatts (mW).

A plot of the power consumption of each setup for 20 iterations of the operations from Pi to the server is in Fig. 3. We averaged the power measurements of each configuration for further comparison. The first configuration, which classifies the sound locally, yielded an average power consumption of 1853.00 mW. The second configuration moves the classification process to the server. We recorded an average power consumption of 1830.84 mW from the Pis. The difference in average power is 22.16 mW. This result shows that having the classifier within the Pi does not significantly increase the energy required to run the platform. We can attribute some of the offset power to the disparity in data the Pi has to send. The first configuration only needs to send the results of the classification. However, the second needs to transmit the whole sound clip to the server. This difference in the data size could have resulted in demanding the same amount of power from the Pi as to how much the device spends on running the classification. As a result, the tests show no significant drawbacks in moving the sound classification from the server to the Pis.

We confirm through our results that our platform carried out the application while allocating resource-heavy processes within Raspberry Pi 3B. Tensorflow demands significant processing power to generate neural network models. Also, pre-processing sound data adds more complexity to the flow of data within the Pi and the work its processors need to do. However, our platform configuration was able to cater to it and maintain power efficiency, staying on par with a standard implementation where the server runs the classifier. Therefore, the experimental results present the feasibility of the proposed configuration as a lightweight and efficient option.

C. Latency Comparison

The platform's application is to measure sound within an urban setting. Therefore, multiple instances of the Pis must simultaneously sense and eventually send data to the server to cover the area. The next test was to investigate the impacts of either configuration on the network latency. We tested each setup by measuring the latency while increasing the network size to 4, 8, and 12 connected Pis, each attempting to send data to the server. Each Pi connects to the server wirelessly through TCP/IP socket communication and sends the data

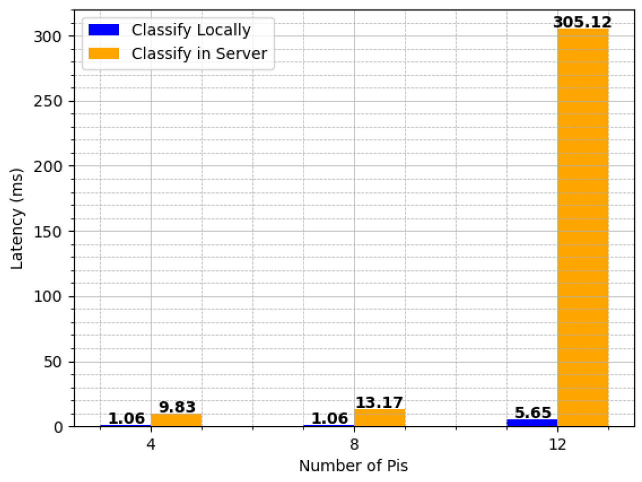


Fig. 4. Latency of configurations at an increasing number of Pis.

as packets. We measured the average latency as the elapsed start-to-end time of the end device's data transmission over the total number of data packets it sent, as shown in the following equation:

$$\text{latency}_{\text{average}} = \frac{(t_{\text{end}} - t_{\text{start}})}{\text{totalpackets}}. \tag{1}$$

The plot showing the resulting average latencies of each configuration is in Fig. 4. Based on the results, classifying the sound within the server is not a feasible option as the network grows. The graph shows that as the number of Pis connecting to the server increases to 12, its latency shoots up to 305 ms. On the other hand, locally classifying maintains a reasonably low latency during all three test iterations. Since each Pi only needs to send the classification results, it transmits less data, keeping the TCP/IP socket free for other Pis to use. However, with the second configuration, the Pi must send the whole sound clip to the server, requiring them to transmit more data packets. As a result, the socket is more occupied, leading to longer latencies. This observation gives the upper hand to our proposed configuration's network management. Combining it with the previous test's results, we can safely infer that our proposed design that locally classifies urban sound benefits the platform's performance.

D. Discussion

By comparing the two configurations, the advantages IoT systems give to the proposed design are as follows.

1) Data Collection Efficiency: Without the IoT system, it will be harder for the data to be sent to the server effectively, while a wired connection would increase the overall cost. On the other hand, IoT devices can be placed almost everywhere with limited maintenance requirements. In this way, deploying the system is faster overall. Connecting our proposed design requires less effort when setting up the platform. As a result, there is less downtime during maintenance.

2) Energy Consumption Efficiency: The Pis will have significantly lower power consumption while still being able to do their function. Compared with the conventional wired setup, which uses desktop computers to collect data, the power required to run it on a citywide scope is significantly higher. Therefore, using the Pis as a substitute is better for a low-cost option.

3) Cost Efficiency: Our proposed design provides a more scalable design with the Pis and transmits data wirelessly. Due to the number of sound-collecting devices, deploying a fully server-based implementation will be costly. It will also not be cost-effective due to the intended placement of the device. The proposed platform requires the devices to be around the city where natural forces can weather and damage it. Overall, it uses less expensive parts that are easier to replace and redeploy.

Following the experimental results, our proposed platform has advantages in accuracy, power consumption, and latency when the processing is at the edge and not in the cloud.

V. CONCLUSION

In this letter, we proposed a platform using an urban sound classifier within an IoT system. We demonstrate the efficiency of edge-based processing for urban noise classification under a low-cost IoT platform. Our neural network design presents an accuracy of 89.95%, which reasonably models the dataset for urban sound classification. Experimental results highlight that the proposed platform has low power requirements on distributed mediums. On the other hand, a central and cloud-based network arrangement increases latency. Through a distributed, edge-based platform, we present an approach for future classification schemes in urban settings that require close-to-real-time analytics.

ACKNOWLEDGMENT

This work was supported by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN/2016-04007.

REFERENCES

[1] T. Johnson and E. Kanjo, "Urban wellbeing: A portable sensing approach to unravel the link between environment and mental wellbeing," *IEEE Sens. Lett.*, vol. 7, no. 3, Mar. 2023, Art. no. 5500704.
[2] A. S. Alashaikh and F. M. Alhazemi, "Efficient mobile crowdsourcing for environmental noise monitoring," *IEEE Access*, vol. 10, pp. 77251–77262, 2022.
[3] M. Baucas and P. Spachos, "A scalable iot-fog framework for urban sound sensing," *Comput. Commun.*, vol. 153, pp. 302–310, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419312502
[4] M. Baucas and P. Spachos, "Using cloud and fog computing for large scale IoT-based urban sound classification," *Simul. Modelling Pract. Theory*, vol. 101, 2020, Art. no. 102013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X19301431
[5] E. Tan, F. Karnapi, L. Ng, K. Ooi, and W. Gan, "Extracting urban sound information for residential areas in smart cities using an end-to-end IoT system," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14308–14321, Sep. 2021.
[6] Q. Meng, P. Lu, and S. Zhu, "A smartphone-enabled IoT system for vibration and noise monitoring of rail transit," *IEEE Internet Things J.*, vol. 10, no. 10, pp. 8907–8917, May 2023.
[7] A. Rauniyar et al., "NEMO: Real-time noise and exhaust emissions monitoring for sustainable and intelligent transportation systems," *IEEE Sensors J.*, vol. 23, no. 20, pp. 25497–25517, Oct. 2023.
[8] L. Bine, A. Boukerche, L. Ruiz, and A. Loureiro, "Leveraging urban computing with the Internet of Drones," *IEEE Internet Things Mag.*, vol. 5, no. 1, pp. 160–165, Mar. 2022.
[9] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 1041–1044.
[10] Y. Shao et al., "Multisignal joint HVCB fault diagnosis research based on adaptive framing MFCC feature extraction method," *IEEE Sensors J.*, vol. 23, no. 22, pp. 27779–27794, Nov. 2023.