Kiran Srivastava , 2026, 14:4
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Adaptive Sound Profiles: Using Machine Learning

Kiran Srivastava, Shreyansh Singh, Syed Shahzeb Ali, Uddhav Garg
Electrical And Electronics Engineering Galgotias College of Engineering and Technology
Greater Noida, India

Abstract- The personal audio devices often fall short because they can't keep up with the constantly changing condition, we use them in. People end up adjusting the volume over and over-whether the background noise suddenly gets louder, the listening environment shifts, or their own habits change. All of this interrupts the experience and makes listening less enjoyable. This study explores how to fix that by introducing a fully autonomous, personalised audio system called adaptive sound profiles. At the core of this approach is a blend of machine-learning-based personalisation, real-time sound monitoring, and a hands-free gesture control system. It uses two microphones: one listens to what's coming out of the speaker, and the other listens to what's happening around the user. By comparing these two audio signals, the system can automatically adjust the volume so that the sound stays clear and comfortable, no matter what's going on in the environment. The framework doesn't stop at sound alone. It also considers factors like time of day and predicted user behaviour. A built-in Memory Component keeps track of each person's listening preferences and applies them on its own, allowing the system to grow and adapt with the user over time. Instead of just reacting when the noise around you suddenly changes the system learns from what is happening in the environment and prepares in advance. The result is a smart, privacy-friendly and personalized way to control audio making listening feel smoother and much more enjoyable.

Keywords: Adaptive Sound Profiles, Personalized Audio Systems, Machine Learning, Real-Time Audio Processing, Ambient Noise Detection, Automatic Volume Control, Dual-Microphone System

I. INTRODUCTION

Most personal or public audio devices don't adapt well to real world situations. Because of this people often find themselves constantly adjusting the volume whenever the surrounding noise or situation changes which can break the flow of the listening experience.This project fixes that by making a fully independent personalized audio system through adaptive sound profiles.

The different feature is mixing ML driven changes, live sound awareness, and a quick, no touch gesture interface. The core goal is making and running the efficient ML model on an Arduino for auto volume control. It blows past basic auto-gain that only chases overall sound energy by throwing in smart context.

Here how the system juggles different inputs:

- **Sound Awareness:** It classifies environmental noises using a convolutional neural networks on melspectrograms, giving a deeper vibe on the scene like calling it busy street or quiet park not just small decibels. Helps make way smarter volume adjustments.
- **Contextual Personalization:** It factors in non-sound details, including the time, day of the week, or inferred activities, to proactively tweak based on your routines and typical trends.
- **Explicit Feedback Loop:** A memory component saves and applies your preferred volumes, while an APDS-9960 gesture sensor lets you override instantly without contact. Gestures provide rock-solid feedback to keep refining the ML profiles.

By running the system on low power hardware using TinyML techniques like quantization it becomes both efficient and privacy friendly. This approach helps tackle a common challenge in audio control while bringing advanced machine learning research on sound analysis into a practical easy to use system. In the following sections I will explain the methods, implementation, results and conclusions.

Adaptive and Context Aware Control Systems

Adaptive systems are made to maintain performance when things are changing [1]. In modern applications, especially those with human elements, we learn hard on AI and ML like reinforcement

learning to learn from data and adapt, stability way beyond fixed rules [2, 3].

A. Personalized Sound Profile Modelling

The system aims to implement a memory component that stores and uses your volume preferences, building an adaptive profile. This fits right in with context aware recommender systems that get personal by weaving in psychological, social, and environmental factors [4].

In audio terms, it learns from your feedback manual volume changes as a function of context . By tracking and classifying these user context links, the ML model can dynamically switch or mix profiles, letting it not just react to current noise but apply a preferred profile based on the received pattern. Pattern based adaptation is a game changer for user experience.

B. Contextual Feature Integration

The literature pushes for context-aware systems to use a rich set of non-acoustic metadata for meaningful personalization. The features we leverage include time of day, day of week, user activity (inferred from location or sensors), and recurring usage patterns. Proactive personalization models use this temporal and spatial context to anticipate needs, like predicting a volume drop during a user's typical quiet period [5, 6].

These contextual features are key inputs to the ML model, working with acoustic data to define the operational state. Traditional adaptive noise control usually adjusts audio only by looking at decibel levels. By adding extra variables the system can better understand the situation and tell the difference between places where louder audio makes sense and those where quieter sound is more suitable resulting in smarter and more efficient adjustments [7].

II. MACHINE LEARNING FOR ACOUSTIC SENSING

Adaptive volume control depends on environmental awareness which comes from machine hearing a field that uses machine learning to interpret and understand sound data [8]. This makes it possible to classify the surrounding environment providing much richer information than simply measuring how loud the sound is.

A. Environmental Sound Classification

For the system to control volume effectively it first needs to classify environmental sounds to understand the surrounding noise. Since audio signals constantly change over time, careful preprocessing is necessary to make the data suitable for analysis. Go to approach transforms the one dimensional audio signal into a time frequency representation, especially the melspectrogram [9]. This maps acoustic energy onto the non-linear melscale, copy human audit perception, and creates a two dimensional, image like feature map that is good for pattern recognition [10].

These spectrograms are fill into deep learning architectures, usually convolutional neural networks, which excel at taking local features and material patterns across frequency bands. This lets the model distinguish fine but contextually important sounds, classifying complex sound scenes instead of just returning a raw decibel measurement [11].

B. ML Algorithms and Model Selection for Regulation

The ML model, developed in Python using the TensorFlow/Keras framework, ultimately translates the classified acoustic scene and contextual data into an optimal volume setting. This is framed as a multi-class regression problem, with the output being a continuous volume level. The model acts as a control mechanism derived from data, aiming to minimize the discrepancy between the forecasted volume and the user's historical preferred volume for that situation [2].

Deciding between a Convolutional Neural Network and a hybrid CNN architecture is an important step. A standard CNN is very good at recognizing patterns in spectrograms, but a hybrid model can also capture contextual elements like how sounds change over time. This extra information can help the system make smarter and more reliable decisions.

III. HYBRID INTERACTION VIA GESTURE RECOGNITION

Since immediate user control and reliable feedback are important for any automated system this project includes a non contact interface that allows users to interact with the system easily.

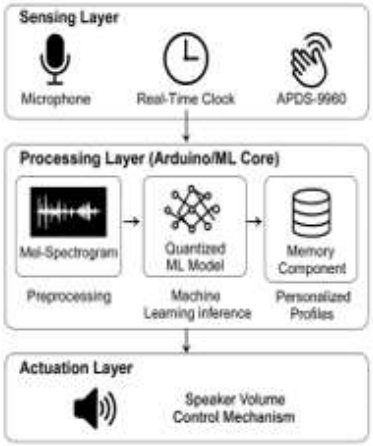
Near-Field Gesture Sensing (APDS-9960)

The APDS-9960 is a small low power sensor that is well suited for detecting gestures and proximity in embedded systems. It works by emitting infrared light and analyzing reflection intensity through four photodiodes (UP, DOWN, LEFT, RIGHT). This enables it to identify simple directional swipes (UP/DOWN for volume adjustment, LEFT/RIGHT for track selection) without needing complex computer vision. The integration provides two wins: quick, hands-free user control and a prime data source for the ML system. A recognized manual gesture serves as the most definitive form of explicit user feedback, acting as a high-priority override to the autonomous system and contributing a new labeled data point for ongoing training of personalized profiles.

IV. SYSTEM METHODOLOGY AND IMPLEMENTATION

The success of the "Adaptive Sound Profiles" project hinges on a careful implementation process, moving from cloud-based model training to resource-optimized embedded deployment, aka Tiny Machine Learning (Tiny ML).

Overall System Architecture Diagram (Figure1)



A. Hardware Selection and Interfacing

The system is built around an Arduino microcontroller, which serves as the main processing unit and offers a low cost low power solution for embedded computing. It is connected to several key components including a digital microphone to detect surrounding noise the APDS-9960 sensor for gesture recognition and an actuator like a digital potentiometer or small motor to adjust the speaker's physical volume. All sensors are initialized and managed using C++ code, and the APDS-9960 communicates with the microcontroller through the I^2C bus. [12].

B. Data Acquisition and Preprocessing

The project requires the acquisition of a custom, multi-modal dataset. This dataset comprises:

1. **Acoustic Feature Generation:** Raw audio recordings are windowed, transformed using Short-Time Fourier Transform (STFT), and converted into Mel-Spectrograms, which are then normalized for model training.
2. **Contextual Feature Encoding:** Non-acoustic data (Time of Day, Day of Week, etc.) is numerically encoded and concatenated with the acoustic features to form the complete input vector for the personalized model [15].
3. **Target Labelling:** The output (Volume Level) is labelled based on desired acoustic conditions and manual user adjustments, forming the supervised learning target.

C. Tiny ML Optimization and Training

The ML model is trained on a powerful host machine before being optimized for the embedded target [14]. To address the severe memory and computational constraints of the Arduino, the following optimization techniques are mandatory:

1. **Model Architecture Simplification:** Employing compact and efficient neural network designs (e.g., Mobile Nets or simple CNNs) to minimize parameter count.
2. **Quantization:** This is the most critical step, converting the model's weights and activations from 32-bit floating-point precision down to 8-bit integers.Tools such as TensorFlow Lite Micro make this process easier by shrinking the model size by up to four times and speeding up

inference. As a result real time processing directly on the device becomes practical and efficient [16].

D. Embedded Deployment and Real-time Execution

After optimization the model is converted into a lightweight C++ header file or C array. This file is then compiled together with the main arduino sketch allowing the microcontroller to run the model directly[13].

Embedded System Execution Flowchart (Figure2)
By processing everything directly on the device this approach provides fast response times, uses less power and helps protect user privacy[14].

V. GAPS IN THE LITERATURE AND HOW THIS PROJECT ADDRESSES THEM

1. Semantic vs energy based adaptation Many deployed systems still rely on simple SPL thresholds. Recent ESC work highlights the benefit of semantic scene understanding (e.g., differentiating music vs. machinery), but few applied projects demonstrate an end-to-end pipeline from ESC → personalized volume control on microcontrollers. This project implements the full chain on Arduino hardware, closing the gap between research prototypes and deployable systems.

2. Hardware aware ESC models Although compact ESC models and hardware-aware neural architecture search (NAS) methods already exist they are rarely used in adaptive control systems that include features like gesture based overrides and memory modules. This project bridges that gap by combining hardware efficient ESC models with quantization aware training (QAT) and gesture driven online labeling resulting in a practical solution designed with hardware limitations in mind.

3. Practical personalization with privacy constraints Federated learning is a promising idea but implementing it directly on microcontrollers is still quite complex because of their limited resources. Instead, this project follows a more practical approach logging data locally and occasionally retraining the model on a host device, with the possibility of adding federated learning later. This approach reflects the strategies recommended in many TinyML surveys as a realistic solution for current systems.

VI. EVALUATION APPROACHES RECOMMENDED BY THE LITERATURE

The evaluation should include:

- ESC accuracy on standard benchmarks (ESC-50 / UrbanSound8K) to validate scene classification performance.
- Regression error (RMSE) between predicted and user-selected volume across contexts, measured in the field.
- The time delay between detecting input and performing the action is measured in milliseconds to ensure the system feels responsive to users ideally staying below 100–200 ms.
- Power profiling during continuous inference to estimate battery life on target hardware.
- **User study metrics:** reduction in manual adjustments per session, user satisfaction scores, and false-positive override rates for gesture control.

VII. CONCLUSION

The research over the last few years provides a strong technical foundation (ESC feature engineering, Tiny ML toolchains, hardware-aware model design, and simple gesture HCI). However, an integrated, deployable example that unites ESC, personalized Tiny ML inference, gesture override, and privacy-aware personalization on microcontrollers is still rare. This project addresses that gap by implementing a quantized ESC/regression pipeline on Arduino hardware, a gesture-based explicit-feedback channel, and a pragmatic personalization strategy that balances privacy and performance.

REFERENCES

1. G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.
2. D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in Proc. 2nd Int. Conf. Learn. Represent. (ICLR), Banff, AB, Canada, Apr. 2014, pp. 1–14, DOI: 10.48550/arXiv.1312.6114.
3. S. Garg, K. Rai, D. Sharma, and S. Sharma, "Design and analysis of adaptive control systems: Adaptive control algorithms using machine learning," ResearchGate Technical Report, DOI: 10.13140/RG.2.2.24765.33762, July 2024.
4. G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in Recommender Systems Handbook, 2nd ed., F. Ricci et al., Eds. New York, NY, USA: Springer, 2015, pp. 191–226, DOI: 10.1007/978-1-4899-7637-6_6.
5. T. A. Baldissera, L. Camarinha-Matos, and C. De Faveri, "Context-Aware Proactive Personalization of Linear Audio Content," in Proc. 8th Doctoral Conf. Comput., Electr. and Ind. Syst. (DoCEIS), Costa de Caparica, Portugal, May 2017, pp. 1–12, DOI: 10.1007/978-3-319-56077-9_38.
6. K. Eves and J. Valasek, "Adaptive control for singularly perturbed systems examples," Code Ocean Engineering Capsule, Version 1.0, Aug. 2023.
7. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, no. 8, pp. 740–741, August 1987, DOI: 10.1109/TJMJ.1987.4549593.
8. J. Nordby, "jonnor/machinehearing: machine learning applied to sound," GitHub OpenSource Software, Version 2.1.4, 2022.
9. G. F., "How can machine learning be used in audio analysis?" Towards Data Science: AI & Engineering, Jan. 2023.
10. S. Liu, "Wi-Fi Energy Detection Testbed (12MTC)," GitHub Research Repository, Project ID: 12MTC, 2023.
11. T. Lattner and R. M. Nistal, "Stochastic Restoration of Heavily Compressed Musical Audio Using Generative Adversarial Networks," Electronics, vol. 10, no. 11, p. 1349, June 2021, DOI: 10.3390/electronics10111349.
12. L. J., "APDS-9960 Gesture and Color Sensor with Arduino," Makerguides Sensors & Embedded Systems, Jan. 2024.
13. S. Mistry and D. Pajak, "Get Started With Machine Learning on Arduino," Arduino Professional Documentation, July 2024.
14. Z. N., "Beginner's Guide To Embedded Machine Learning," Zero To Mastery: Embedded AI Series, 2025.
15. V. Pemmaraju, S. A. Pasala, P. Rishitha, P. Sanjana, and E. S. Bandaru, "Volume and Brightness Control with Hand Gestures using Computer Vision," Int. J. Novel Res. Dev. (IJNRD), vol. 8, no. 3, pp. a426–a430, March 2023.
16. S. M. Hussain, J. Wang, and L. Zhang, "Design of Convolutional Neural Network Optimization Algorithm Based on Embedded System," IEEE Access, vol. 12, pp. 150–160, June 2024, DOI: 10.1109/ACCESS.2024.3394851.