Real-Time Emergency Vehicle Siren Detection with Efficient CNNs on Embedded Hardware

1st Marco Giordano
Dpt. of Information Engineering, Computer Science and Mathematics (DISIM)
University of L'Aquila
L'Aquila, Italy
marco.giordano3@graduate.univaq.it

2nd Stefano Giacomelli
Dpt. of Information Engineering, Computer Science and Mathematics (DISIM)
University of L'Aquila
L'Aquila, Italy
stefano.giacomelli@graduate.univaq.it

3rd Claudia Rinaldi
National Inter-University Consortium for Telecommunications (CNIT)
University of L'Aquila
L'Aquila, Italy
claudia.rinaldi@univaq.it

4th Fabio Graziosi
Dpt. of Information Engineering, Computer Science and Mathematics (DISIM)
University of L'Aquila
L'Aquila, Italy
fabio.graziosi@univaq.it

***Abstract*—We present a full-stack emergency vehicle (EV) siren detection system designed for real-time deployment on embedded hardware. The proposed approach is based on E2PANNs, a fine-tuned convolutional neural network derived from EPANNs, and optimized for binary sound event detection under urban acoustic conditions. A key contribution is the creation of curated and semantically structured datasets—AudioSet-EV, AudioSet-EV Augmented, and Unified-EV—developed using a custom AudioSet-Tools framework to overcome the low reliability of standard AudioSet annotations.**

The system is deployed on a Raspberry Pi 5 equipped with a high-fidelity DAC+microphone board, implementing a multi-threaded inference engine with adaptive frame sizing, probability smoothing, and a decision-state machine to control false positive activations. A remote WebSocket interface provides real-time monitoring and facilitates live demonstration capabilities.

Performance is evaluated using both framewise and event-based metrics across multiple configurations. Results show the system achieves low-latency detection with improved robustness under realistic audio conditions. This work demonstrates the feasibility of deploying IoS-compatible SED solutions that can form distributed acoustic monitoring networks, enabling collaborative emergency vehicle tracking across smart city infrastructures through WebSocket connectivity on low-cost edge devices.

***Index Terms*—Sound Event Detection, Emergency Vehicle Sirens, Raspberry Pi 5, Real-Time Inference, Embedded Systems, Urban Acoustics**

I. Introduction

Emergency vehicle (EV) siren detection is a key enabler of intelligent transportation systems, supporting real-time decision-making for autonomous vehicles, traffic monitoring infrastructures, and urban sound analysis platforms [1]–[3]. The ability to recognize sirens promptly from live audio streams enhances situational awareness, enabling applications such as automated braking, smart rerouting, and prioritization of emergency response.

Unlike vision-based approaches that rely on line-of-sight, audio-based emergency vehicle detection provides 360-degree awareness and operates effectively in occluded scenarios, at night, and in adverse weather conditions. This makes sound-based detection a critical complement to visual systems in autonomous navigation.

This work aligns with the emerging Internet of Sounds (IoS) paradigm, where distributed acoustic sensors form interconnected networks for collaborative environmental monitoring. Unlike isolated detection systems, our approach is designed to integrate seamlessly into IoS infrastructures, enabling city-wide emergency vehicle tracking through coordinated edge nodes. The WebSocket-based interface facilitates real-time data sharing between detection nodes, supporting applications such as traffic light preemption, route optimization for emergency responders, and acoustic-based situational awareness in smart cities.

Despite growing research interest in sound event detection (SED) for safety-critical contexts, current solutions face limitations in terms of real-time operability, generalization to urban noise conditions, and hardware deployability [4]. Many deep learning (DL) approaches rely on heavy models or offline processing pipelines, while low-power embedded implementations often sacrifice accuracy and robustness [5], [6].

This paper addresses these challenges by introducing a real-time EV siren detection pipeline optimized for embedded platforms. Building upon Efficient/Emergency Pre-trained Audio Neural Networks (E2PANNs) [7] - a specialization of EPANNs framework [8], [9] - we propose a lightweight, convolutional architecture tailored for real-time framewise inference of emergency siren events. The model is fine-tuned on a curated subset of AudioSet [10] (AudioSet-EV) [11], and evaluated using

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART", CUP F83C22001690001) and by the European Union - NextGenerationEU under the Italian Ministry of University and Research (MUR) National Innovation Ecosystem grant ECS00000041 - VITALITY - CUP E13C22001060006.

both framewise and event-based metrics [12]. We introduce a set of post-inference processing strategies — including a moving average filter over the inference sequence, dynamic adjustment of frame width based on confidence thresholds, and a requirement for a minimum number of consecutive positive frames — designed to suppress spurious activations and minimize false positive detections while maintaining responsiveness and precision.

The system is deployed on a Raspberry Pi 5 equipped with a Raspiaudio Ultra++ DAC+mic board, supporting live audio input and efficient model inference. A companion remote interface provides runtime monitoring and performance feedback, enabling real-time visualization of classification results and inference metrics.

Our contributions are as follows:

- We design and optimize a compact CNN-based siren detector based on E2PANNs [7], achieving real-time performance on Raspberry Pi 5.
- We release and utilize AudioSet-EV, a filtered and taxonomically harmonized subset of AudioSet specialized for EV detection.
- We propose and evaluate an adaptive framewise inference mechanism to improve latency-accuracy trade-offs and false positive rejection.
- We implement a full-stack embedded solution with remote access interface for real-time feedback and demo reproducibility.

The Internet of Sounds vision requires edge devices capable of both local intelligence and network collaboration. Our system addresses this dual requirement by combining efficient on-device inference with standardized IoT protocols. This enables scenarios such as:

- Mesh networks of acoustic sensors providing redundant coverage at critical intersections.
- Federated learning where edge nodes collaboratively improve detection models without raw audio sharing.
- Integration with existing urban IoT infrastructure for holistic emergency response coordination.

To the best of our knowledge, this is the first publicly documented implementation of a CNN-based EV siren detector achieving real-time inference on embedded hardware with live audio input and adaptive detection logic.

II. Related Work

Emergency siren detection from audio has been the subject of increasing attention, with approaches ranging from handcrafted feature extraction and classical machine learning models to modern deep learning-based frameworks. While accuracy has improved significantly, especially under controlled conditions, a majority of these systems remain unsuitable for real-time deployment on embedded hardware due to high computational demands or lack of inference latency evaluation.

Early works, such as Castorena et al. [1], proposed CRNN and YOLO-based models trained on synthetic mixtures of sirens and ambient noise, achieving high accuracy but requiring GPU-level resources for real-time inference. Similarly, Ramírez et al. [2] utilized spectrograms and 2D CNNs to classify siren types in urban recordings, though their evaluation showed high sensitivity to real-world SNR fluctuations.

More efficient models have been proposed recently. Shams et al. [13] combined EfficientNet with 1D CNNs and self-attention layers, demonstrating 209 ms inference time on short clips. However, their small dataset and lack of embedded testing limits generalizability. Mittal et al. [4] applied ensemble methods using CNNs, RNNs, and FC layers, reaching high classification accuracy at the cost of 1.5 s per inference.

Alongside spectrogram-based CNNs, recent work has explored alternative handcrafted features tailored to the harmonic structure of sirens. Damiano, Dietzen, and van Waterschoot [14] proposed frequency-tracking features extracted with adaptive notch filtering, enabling data-efficient training of a compact CNN that generalizes well even with limited data. Our approach, by contrast, leverages E2PANNs as a pretrained backbone, fine-tuned for the siren detection task in embedded deployment scenarios.

Only a few studies focused on embedded or resource-constrained implementations. Miyazaki et al. [5] and Meucci et al. [6] explored DSP and microcontroller-based siren detection using FFT and pitch tracking, but without deep learning or large-scale evaluation. Beritelli et al. [15] used LPCs for analog inference on TI DSPs, though false positives under real-world noise remained an issue.

In terms of datasets, benchmarks such as ESC-50 [16], UrbanSound8K [17], and FSD50K [18] have limited relevance due to taxonomic ambiguity and label sparsity. Specialized corpora like LSSiren and SireNNet [19]–[21] offer more structured siren detection setups but are often small or lack diversity. Even though affected by weak labeling, AudioSet remains the most comprehensive source, and our derived subset, AudioSet-EV [11], ensures taxonomic clarity and reproducibility.

An alternative approach to address the scarcity of high-quality siren data is the use of synthetic datasets. Damiano et al. [22] show that generating siren signals with simulated propagation and Doppler effects, combined with real-world background noise, can substantially improve generalization across unseen datasets. Our contribution instead focuses on curating and manually verifying AudioSet-derived clips to ensure label reliability, which complements synthetic approaches by grounding evaluation on real-world recordings.

Evaluation standards have evolved to include framewise and event-based metrics [12], yet false positive suppression and noise robustness remain underexplored. Our work extends this line by integrating adaptive post-processing, a novel real-time embedded deployment, and a monitoring interface for live metric inspection.

III. Baseline Model Architecture and Training

The architecture adopted in this work builds upon the Efficient Pruned Audio Neural Networks (EPANNs) framework [8], [9], a structured pruning variant derived from the CNN14 model within the PANNs family. EPANNs were selected as

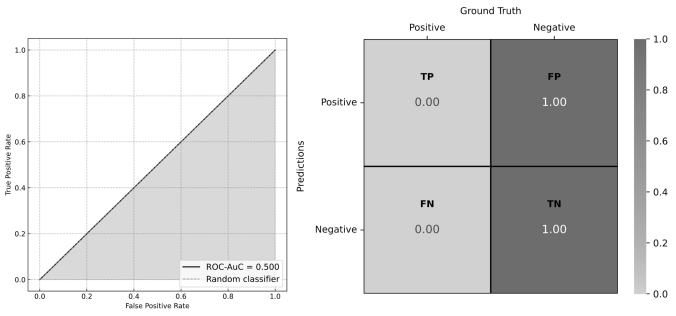

Fig. 1. E-PANNs Confusion Matrix and ROC-AUC computed on preliminary AudioSet filtered samples.

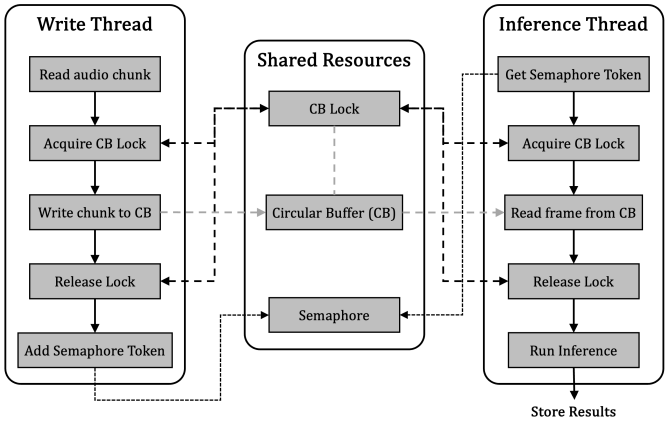

Fig. 2. E2PANNs RT inference threading system running on Raspberry Pi 5.

the foundation of our system due to their favourable trade-off between computational cost and baseline performance across general-purpose audio tagging tasks.

Our contribution begins with a thorough assessment of EPANNs in the context of Emergency Vehicle (EV) siren detection.

A. Motivations for Fine-tuning and Dataset Curation

As a preliminary step, we tested the pre-trained EPANNs model on an AudioSet-derived subset composed of one positive partition (clips annotated with Emergency Vehicle label) and three negative partitions (Traffic, Vehicles, and non-EV Alarms classes). This subset was designed to reflect a realistic and acoustically challenging deployment environment. Evaluation results revealed that EPANNs failed to discriminate between classes in this configuration, yielding a ROC-AUC close to 0.5 and effectively behaving as a degenerate constant classifier (always predicting absence of sirens) (see Fig. 1) [7].

This observation prompted the need for a domain-adapted fine-tuning strategy and a more specialized dataset. To this end, we developed AudioSet-Tools [11], a Pytorch framework designed for precise label-based filtering, rebalancing, and reproducible dataset curation from AudioSet.

B. AudioSet-EV and Dataset Variants

Using the AudioSet-Tools framework [11], we curated the following:

- **AudioSet-EV:** a filtered subset consisting exclusively of EV siren and carefully selected negative samples, enforcing label disjointness [11].
- **AudioSet-EV Augmented:** an expanded version of the previous set with time-domain data augmentation applied online during training, including random noise injection, polarity inversion and temporal shifts.
- **Unified-EV:** a merged dataset combining the previous two with *ESC-50* [16], *SireNNet* [19], *LSSiren* [23], *UrbanSound8K* [24], *FSD50K* [18] for generalization testing.

These datasets underpin all subsequent training and evaluation phases.

C. From EPANNs to E2PANNs

We fine-tuned the EPANNs model on AudioSet-EV and AudioSet-EV Augmented using supervised training under a binary classification regime (EV vs. non-EV). Hyperparameter search experiments were conducted to identify optimal configurations for learning rate, dropout, and batch size [7].

Training inputs were 64-bin log-Mel spectrograms extracted from 10-second mono audio clips sampled at 32 kHz. Data augmentation included background mixing, time masking, and volume jitter.

Model evaluation was carried out using stratified training/validation splits of Unified-Ev dataset. The fine-tuned model, Efficient/Emergency Pre-trained Audio Neural Networks (E2PANNs), was exported as three best-performing checkpoints for subsequent validation on edge platform:

- `Baseline_EV`: top performing checkpoint trained on Audioset-EV without augmentations and with exponentially decaying learning rate.
- `Augmented_EV`: top performing checkpoint trained on Audioset-EV with augmentations and fixed learning rate.
- `Unified_EV`: top performing checkpoint trained on the Unified-EV dataset III-B.

IV. Real-Time Edge Computing Implementation

Before deploying our E2PANNs model on the experimental target platform (Raspberry Pi 5) for real-time EV-SED experiments, we evaluated its ability to perform inference on small input sizes. The goal was to reduce computational complexity by leveraging the finetuned EPANNs to perform periodic inference on short, variable-length segments of streaming audio data. This approach aims to avoid recurrent layers — common in SoA SED systems — which introduce substantial computational and memory overhead.

To determine the minimum viable input size, we implemented a binary search algorithm [25] to identify the smallest input tensor that produces a valid model output. The minimum valid input size was found to be 9919 samples (approximately 310ms at 32kHz).

To support continuous EV siren detection on edge hardware, we designed a RT inference system tailored for the Raspberry Pi 5. The system (Figure 2) adopts a multi-threaded architecture with explicit concurrency control, ensuring non-blocking and low-latency operation. Its core component is a `CircularBuffer` acting as shared memory between a *producer* thread (writing audio data) and a *consumer* thread (performing inference). Synchronization is enforced using Python's `threading.Lock` and `threading.Semaphore` primitives.

The audio `Write Thread` (producer) emulates RT audio input by periodically writing short chunks from a source file or input buffer to the `CircularBuffer`. Thread-safe access is guaranteed by acquiring a `Lock` during buffer updates. Once data is written, a `Semaphore` token is released, signaling that new content is available for reading.

The `CircularBuffer` is a pre-allocated, fixed-size memory structure supporting wrapping reads and writes. It maintains a sliding window of the most recent samples, enabling continuous streaming behavior. Read/write operations are guarded by a `Lock`, while the `Semaphore` ensures reads occur only after new data has arrived.

The input `Frame Provider` module, embedded in the inference thread, manages frame extraction. Before reading, it acquires a `Semaphore` token (blocking if necessary), then locks the buffer to extract a valid audio frame. Frame length is dynamically adjusted based on the most recent model output: starting from the minimum valid size, the duration is increased whenever the output probability exceeds a predefined threshold. This adaptive mechanism progressively expands the temporal context during likely positive detections, enhancing robustness while limiting computational overhead when EV-like characteristics are absent. The frame duration is bounded by a configurable maximum.

The `Inference Thread` (consumer) runs concurrently, querying the `Frame Provider` for the next input and passing it to E2PANNs. The resulting probability is appended to a shared output list for decision logic. All inference times and timestamps are logged using a custom profiler. The inference loop terminates upon receiving a shutdown signal via a `threading.Event`.

This implementation achieves responsive and low-overhead inference while preserving data consistency. The adaptive frame-length mechanism complements the model's efficiency, allowing longer analysis windows only when acoustically justified.

A. Full-Stack Live System with Remote Interface

To transition from simulated real-time to a live operational setup, we integrated a RaspiAudio Ultra++ DAC+microphone audio board into the Raspberry Pi 5 platform. This board provides high-fidelity audio input with low-latency access to ALSA streams and onboard microphone array support (Fig. 3).

The real-time engine is extended with a WebSocket-based interface for remote monitoring and control. The system

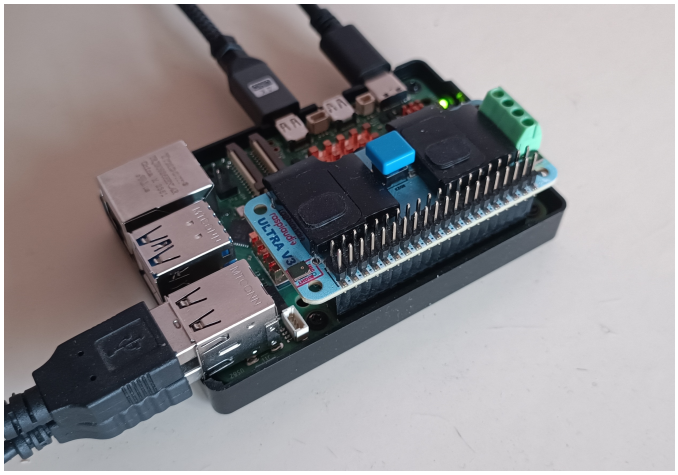

Fig. 3. The Raspberry Pi 5 with RaspiAudio UltraV3 audio board on top

exposes a live stream of classification probabilities, detection flags, and diagnostic metadata (e.g., current frame width, inference time per frame). The frontend, accessible via browser or terminal client, allows operators to inspect system behavior during field deployment or controlled playback scenarios (Fig. 10).

A core component of the postprocessing stage is the event decision state machine, which determines whether a positive siren detection event should be issued. This logic builds upon two key mechanisms that can be enabled and configured to meet specific needs:

- **Moving average smoothing filter:** the sequence of raw inference probabilities is smoothed using a fixed-size moving average window to suppress transient noise and reduce the likelihood of spurious spikes triggering false positives.
- **Consecutive frame validation:** a detection event is confirmed only if a minimum number of consecutive frames (each exceeding a configurable probability threshold) are observed. This adds temporal consistency and reduces instability in event onset detection.

The state machine tracks ongoing classification outputs and triggers or resets detection flags accordingly, ensuring robust decision-making under noisy or ambiguous acoustic conditions. These design choices are critical for minimizing false alarms in edge deployment scenarios, particularly in urban environments.

This full-stack implementation supports live demonstrations, reproducibility experiments, and real-world data acquisition campaigns. It ensures all components — model, dataflow, post-processing, and human interaction — are co-located on a low-cost, power-efficient embedded platform.

Results from the real-time test scenarios and their SED performance analysis are detailed in section V.

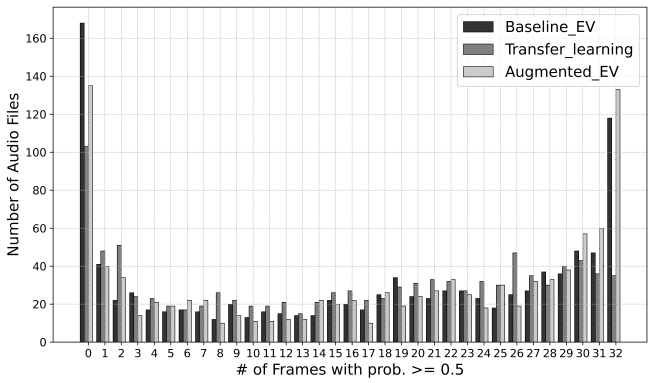

Fig. 4. Distribution of samples based on the number of frames with probability $>= 0.5$

V. Evaluation of Real Time Sound Events Detection

To assess the real-time performance of the E2PANNs model, we deployed it on a Raspberry Pi 5 8 GB (Broadcom BCM2712 2.4GHz quad-core 64-bit Arm Cortex-A76) within a custom multi-threaded inference pipeline, as seen in IV. A ground truth evaluation was conducted using three checkpoints of the model (see III-C) and the 1,025 emergency vehicle (EV)-labeled samples from the AudioSet-Strong dataset [26], a subset of AudioSet with temporally-strong labels, suitable for SED. Each sample underwent frame-wise simulated real-time inference with detailed logging of model outputs, frame durations, frame sizes, and system-level performance metrics including CPU load and memory usage. Three experimental conditions were evaluated across all three model checkpoints: one with fixed frame width, and two with variable frame width increasing at rates of 0.2 and 0.4 seconds per second of sustained over-threshold confidence predictions. A statistical analysis of the number of frames exceeding a 0.5 probability threshold (Figure 4) revealed a subset of 287 audio files with zero confident detections. These files were manually audited, and 182 (63.41%) of them were confirmed to contain no emergency vehicle events despite their positive ground-truth labels. This human post-validation process exposed a significant presence of mislabelled positives in the original dataset and led to the creation of a corrected subset, thereby enhancing the reliability of the test data and strengthening the credibility of subsequent performance evaluations. This finding underscores the importance of the availability of reliable datasets, particularly in safety-critical applications where false positives are more disruptive than missed detections.

To further assess the behavior of the E2PANNs model during real-time operation on the Raspberry Pi 5, we conducted a thorough evaluation of the inference outputs using standard Sound Event Detection (SED) metrics. The evaluation pipeline processed the frame-wise probability scores generated by the model on 1025 audio clips, each corresponding to a 10-second excerpt from YouTube videos containing emergency vehicle sounds. To ensure compatibility with the input formats expected by the `torchmetrics` and `sed_eval` libraries, the outputs were transformed into fixed-resolution binary sequences. This involved discretizing the fixed and variable-length inference windows (which depended on the adaptive frame duration logic) into uniform time steps and aligning them with the corresponding ground truth annotations. The annotations were drawn from a merged metadata file, which included manual validation of each event, and optionally filtered using a list of *false* ground truth labels (i.e., predictions originally annotated as positives but subsequently determined to be incorrect). This allowed the evaluation to reflect a stricter reference standard when required.

Both frame-wise and event-based metrics were computed across all experiments and checkpoints. Frame-wise metrics — computed via `torchmetrics` [27] — included precision, recall, F1-score, accuracy, specificity, balanced accuracy, and error rate. Event-based metrics — obtained using `sed_eval` [12]— included event-level F-measure (and its precision and recall components) as well as error rate (and its insertion and deletion components). All metrics were aggregated and reported both at the individual checkpoint level and in a combined summary (Table I) across the full evaluation set.

To more rigorously evaluate the model's resilience against spurious activations in real-world scenarios — where false alarms may lead to undesirable system behavior or user fatigue — we performed an in-depth false positive (FP) analysis grounded in the model's inference frame structure, following the methodology proposed in previous large-scale SED benchmarks [28] [29]. In this context, a false positive frame is defined as an output frame whose predicted probability exceeds the classification threshold but does not temporally overlap with any annotated ground truth event.

The computed metrics, summarized in Table II, offer a multifaceted view of false positive behavior for each checkpoint over the three experiments. We report: (col. 3) the average number of false positive inference frames per sample (FP_FW_AFPS); (col. 4) the average proportion of these frames relative to the total number of inference frames (FP_FW_AFPSP); (col. 6) the total number of false positive events (FP_EB_T), defined as contiguous sequences of at least N consecutive FP inference frames (with $N = 3$); (col. 7) the average model confidence associated with these FP events; (col. 8) the maximum observed run length of consecutive FP frames; and (col. 9) the mean run length over all events. Additionally, we computed (col. 5) the average model confidence over all inference frames (FP_FW_AC), providing a baseline against which the confidence of spurious activations can be contrasted.

To complement these aggregate statistics, we generated per-experiment histograms 5 6 7 showing the frequency of FP event durations (in inference frames), stratified by checkpoint. These visualizations highlight the distribution and persistence of false positive bursts, with each bar annotated by the corresponding average model confidence. The resulting figures

TABLE I
EVALUATION SUMMARY FOR FRAME-WISE AND EVENT-BASED METRICS (EBM)

FGT	experiment	checkpoint	Frame-wise Metrics						EBM f-measure			EBM error rate		
			precision	recall	f1	accuracy	specificity	bal_accuracy	f1	precision	recall	error_rate	deletion_rate	insertion_rate
True	const_fr	(1)	**88%**	68%	72%	69%	24%	69%	14%	9%	26%	3.30	0.74	2.56
True	const_fr	(2)	87%	64%	69%	65%	25%	65%	12%	8%	24%	3.36	0.76	2.60
True	const_fr	(3)	86%	60%	66%	61%	26%	61%	5%	3%	12%	4.37	0.88	3.49
True	var_fr_02	(1)	87%	74%	76%	73%	20%	73%	26%	21%	35%	2.00	0.65	1.35
True	var_fr_02	(2)	87%	71%	73%	71%	21%	71%	24%	19%	32%	2.04	0.68	1.36
True	var_fr_02	(3)	86%	61%	67%	62%	24%	62%	12%	9%	19%	2.82	0.81	2.01
True	var_fr_04	(1)	87%	**77%**	**78%**	**76%**	19%	**76%**	**30%**	**25%**	**38%**	**1.74**	**0.62**	**1.12**
True	var_fr_04	(2)	87%	74%	75%	73%	19%	73%	27%	22%	34%	1.88	0.66	1.22
True	var_fr_04	(3)	86%	63%	68%	64%	24%	64%	14%	10%	21%	2.60	0.79	1.81
False	const_fr	(1)	76%	57%	60%	63%	31%	63%	13%	9%	22%	2.97	0.78	2.18
False	const_fr	(2)	73%	53%	57%	60%	**32%**	60%	12%	8%	20%	2.98	0.80	2.18
False	const_fr	(3)	77%	50%	56%	57%	**32%**	57%	5%	3%	10%	3.99	0.90	3.09
False	var_fr_02	(1)	76%	62%	64%	67%	27%	67%	24%	20%	29%	1.89	0.71	1.18
False	var_fr_02	(2)	73%	59%	61%	64%	28%	64%	22%	19%	27%	1.88	0.73	1.15
False	var_fr_02	(3)	77%	52%	57%	58%	31%	58%	11%	8%	16%	2.65	0.84	1.81
False	var_fr_04	(1)	76%	64%	66%	69%	26%	69%	27%	24%	32%	1.68	0.68	1.00
False	var_fr_04	(2)	73%	61%	63%	66%	27%	66%	25%	22%	28%	1.75	0.72	1.04
False	var_fr_04	(3)	77%	53%	58%	59%	30%	59%	12%	10%	17%	2.48	0.83	1.65

Column `FGT` refers to checking against false ground truth labelled list.
Checkpoint (1): audioset_ev_augmented; Checkpoint (2): audioset_ev; Checkpoint (3): unified_EV.
In bold the best result for each metric.

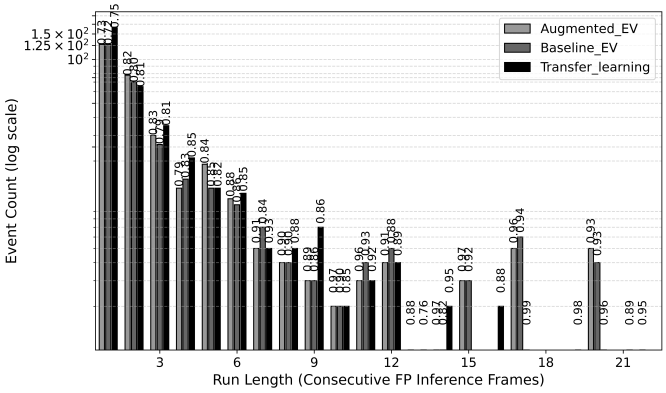

Fig. 5. False Positives frequency distribution over events length for constant frame inference. Average confidence is reported above each bar.

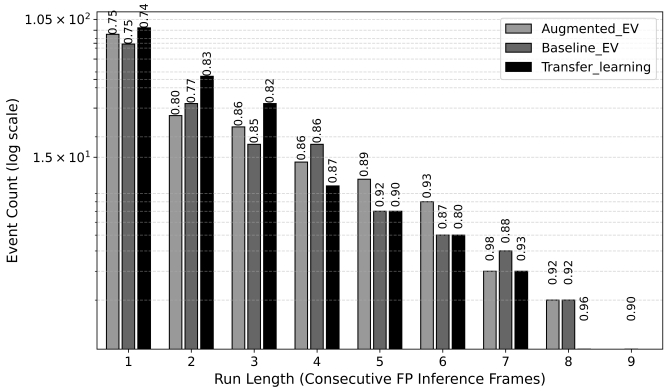

Fig. 6. False Positives frequency distribution over events length for variable frame inference with `increment_speed` $= 0.2$. Average confidence is reported above each bar.

clearly indicate that the `constant_frame` configuration is more prone to sustained and high-confidence spurious activations, whereas both `variable_frame` configurations produce fewer and shorter false positive sequences. This supports the design hypothesis that adaptive windowing contributes to suppressing erroneous activations by leveraging temporally extended input segments, thus enhancing prediction stability in ambiguous regions.

Finally, to verify the feasibility of running the system in real time on the Raspberry Pi 5, we conducted a two-fold analysis. One one side we collected and analyzed CPU and memory load data during inference. From the corresponding log files, we generated greyscale density plots (Figures 8, 9) showing the distribution of resource usage over time. On the other side, to verify the feasibility of live deployment under operational conditions, we conducted two one-hour real-time experiments on the embedded system. The first was performed under an active soundscape containing siren and non-siren events ("multiple detections"), while the second

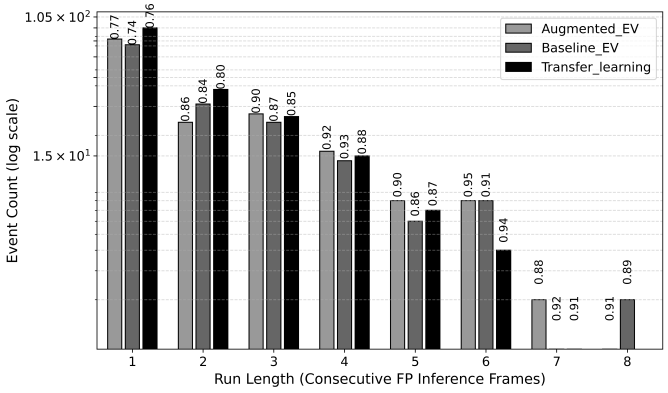

Fig. 7. False Positives frequency distribution over events length for variable frame inference with `increment_speed` $= 0.4$. Average confidence is reported above each bar.

TABLE II
FALSE POSITIVES INFERENCE ANALYSIS

experiment	checkpoint	Frame-wise Based Metrics			Event Based Metrics			
		FP_FW_AFPS	FP_FW_AFPSP	FP_FW_AC	FP_EB_T	FP_EB_AC	FP_EB_MRL	FP_EB_ARL
const_fr	(1)	1.64	5.11%	68.47%	110	86.55%	22	7.06
const_fr	(2)	1.55	4.84%	65.17%	103	85.13%	20	7.15
const_fr	(3)	1.56	4.88%	58.73%	116	84.76%	21	6.16
var_fr_02	(1)	0.62	4.21%	62.69%	61	88.17%	8	4.34
var_fr_02	(2)	0.59	**3.86%**	58.77%	55	87.06%	9	4.44
var_fr_02	(3)	0.64	3.94%	**50.01%**	58	**84.36%**	8	3.97
var_fr_04	(1)	0.59	4.45%	64.78%	62	90.92%	8	4.11
var_fr_04	(2)	**0.56**	4.11%	60.47%	55	89.26%	8	4.16
var_fr_04	(3)	0.57	3.97%	50.45%	**53**	86.75%	**7**	**3.85**

Checkpoint (1): audioset_ev_augmented; Checkpoint (2): audioset_ev; Checkpoint (3): unified_EV
FP_FW_AFPS: False Positives, frame-wise, average number of frames per sample
FP_FW_AFPSP: False Positives, frame-wise, percentage of average number of frames per sample w.r.t. total frames
FP_FW_AC: False Positives, frame-wise, average confidence
FP_EB_T: False Positives, event based, total number of events
FP_EB_AC: False Positives, event based, average confidence
FP_EB_MRL: False Positives, event based, max run length
FP_EB_ARL: False Positives, event based, average run length

was executed in an urban soundscape devoid of EV signals ("no detections"). The system statistics collected during both sessions are summarized in Table III.

In terms of real-time processing performance, the system maintained a processing rate of 1.35x relative to frame duration under active detection, and a perfect 1.00x in the idle case. Maximum observed latency remained below 400 ms in both scenarios. The strong difference of *total inferences* between the two experiments depends on the adaptive frame width mechanism.

From the standpoint of stability, the system showed no runtime interruptions in either session. In the idle case, frame success was near perfect (99.53%), with low CPU (30.3%) and memory (15.5%) usage. In contrast, the detection session demonstrated slightly higher load, but within acceptable thresholds.

No false detections were triggered during the baseline run (as expected, since no EV signals were present), while the active session reported 334 detection events, corresponding to a sustained detection rate of approximately 334 events/hour.

Taken together, these findings show that the system operates at the edge of real-time viability. Importantly, all measurements were obtained without applying low-level optimizations such as model quantization or hardware-accelerated inference backends. These results confirm the viability of a full-stack SED pipeline on embedded ARM platforms and lay the foundation for further latency and efficiency improvements.

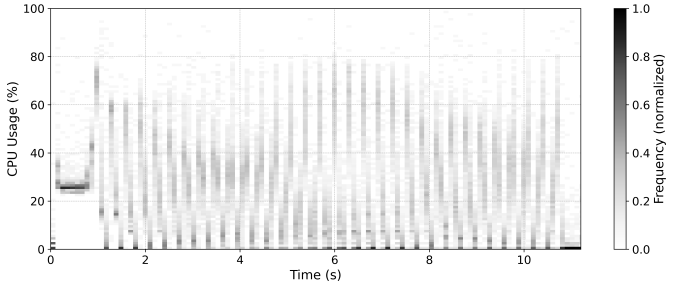

Fig. 8. Greyscale density plot of CPU load for constant frame experiment. Monitoring period $\sim 100ms$. Each frame displays the distribution of CPU load among test samples

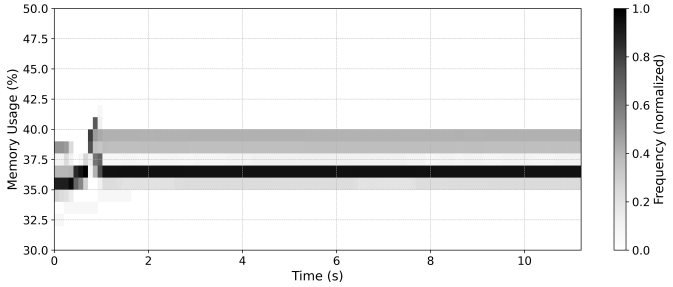

Fig. 9. Greyscale density plot of memory usage for constant frame experiment. Monitoring period $\sim 100ms$. Each frame displays the distribution of memory usage among test samples

VI. LIVE DEMONSTRATION SETUP

The system described in the previous sections has been deployed as a fully operational demonstrator, showcasing real-time emergency siren detection capabilities on a Raspberry Pi 5. The live demo environment replicates a realistic acoustic scenario by incorporating both audio playback of curated soundscapes and live microphone capture.

A. Hardware Configuration

The demonstration platform consists of:

- A Raspberry Pi 5 (8 GB RAM) running a custom Linux-based OS image optimized for low-latency audio processing.
- A RaspiAudio Ultra++ DAC+mic board, providing high-quality microphone input and full ALSA compatibility (Fig. 3).
- External speaker(s) for playback of controlled siren/non-siren audio samples.

The embedded model is executed in a multi-threaded inference pipeline (as detailed in Section IV), enabling real-time

TABLE III
REAL-TIME PERFORMANCE AND DETECTION ANALYSIS: COMPARATIVE EVALUATION

Category	Metric	Multiple Detections	No Detections
Real-Time Processing Performance			
	Normal Avg Frame Duration (ms)	310.0	310.0
	Adaptive Avg Frame Duration (ms)	848.5	-
	Avg. Processing Time (ms)	318.1	309.9
	Overall Real-Time Factor	1.35x	1.00x
	Normal Avg RT Factor	1.00x	1.00x
	Adaptive Avg RT Factor	2.48	-
	P95 Processing Time (ms)	355.2	320.8
	P99 Processing Time (ms)	370.6	328.3
	Max Processing Time (ms)	442.7	370.0
System Stability Analysis			
	Session Duration (h)	1.0	1.0
	Total Inferences	7,249	11,553
	System Interruptions	0	0
	Interruption Rate (per h)	0.00	0.00
	Avg. Processing Rate (fps)	2.0	3.2
Resource Usage Analysis			
	Avg. CPU Usage	29.7%	30.3%
	Peak CPU Usage	34.5%	37.3%
	Avg. Memory Usage	16.0%	15.5%
	Peak Memory Usage	16.5%	16.1%
Detection System Analysis			
	Total Detection Events	334	0
	Detection Rate (events/h)	333.9	0.0

Real-Time Factor is defined as `Frame_Duration / Actual_Processing_Time`
P95 Processing Time is the maximum duration of overall processing of 95% of inference operations
P99 Processing Time is the maximum duration of overall processing of 99% of inference operations

classification on streamed audio without additional compute resources.

B. Web-Based Remote Interface

The system features a WebSocket-based remote interface for live monitoring and operator feedback. Built on a lightweight Python server backend, the interface exposes classification probabilities, detection triggers, and performance diagnostics (e.g., latency per frame, inference confidence trends).

The interface is accessible via browser and supports responsive updates every 100 ms, with visual elements including:

- Real-time probability plot for the 'Emergency Vehicle' class.
- Event detection flags based on state machine logic.
- Diagnostic display for current audio frame parameters (length, inference time).

This interface is essential for demonstration control and transparency, enabling reproducible SED testing, adjustable playback scenarios, and public engagement during the IEEE Internet of Sounds conference.

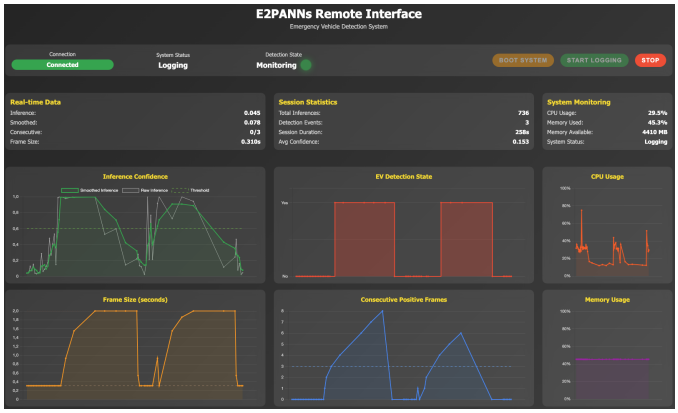

Fig. 10. Layout of the graphic HTML/Javascript interface.

VII. DISCUSSION, CONCLUSION, AND FUTURE WORK

A. Label Reliability and Dataset Curation

One of the most impactful bottlenecks encountered throughout the project was the poor reliability of the original tagging in Google's AudioSet corpus. Despite being one of the most comprehensive open-source audio datasets available, AudioSet suffers from weak labeling practices — many audio segments are annotated based on video-level metadata rather than precise acoustic content.

As a response, we introduced a curation pipeline using the AudioSet-Tools framework and developed the AudioSet-EV and AudioSet-EV Augmented datasets. These filtered and semantically structured distributions aim to mitigate label noise and enforce taxonomic clarity.

However, a problem became particularly evident during the evaluation of the AudioSet-Strong [26] subset, which was manually inspected to verify labeling quality. Numerous mislabelled positives and label inconsistencies were found in clips labeled as containing 'Emergency Vehicle' sirens, significantly affecting both training and testing phases.

Taking all this into account, we believe that the broader issue of unreliable ground truth in large-scale audio corpora remains a fundamental limitation for training models in safety-critical domains.

B. System Limitations and Deployment Challenges

Another concern is the potential overfitting of the fine-tuned E2PANNs model to curated subsets. While evaluation on Unified-EV helped to assess generalization across diverse urban acoustic conditions, the performance may still degrade when exposed to highly variable environmental factors such as occlusions, reverberation, and background chatter. Further integration of data from public datasets and in-the-wild recordings may be necessary.

Although the embedded system performs within acceptable latency bounds, the need for multiple consecutive positive frames before declaring a detection introduces a trade-off between robustness and speed. In scenarios where siren cues are short-lived or partially occluded, delayed response could be critical.

While false positive suppression naturally improves standard accuracy metrics, in safety-critical contexts such as emergency siren detection false positives are disproportionately disruptive compared to false negatives. We therefore treat FP analysis as a dedicated design target, not just a derivative of global metrics.

Lastly, the effectiveness of detection is sensitive to the placement and quality of the microphone. In outdoor deployments or mobile installations, wind noise, vibration, and signal clipping can introduce errors. Future versions of the system could integrate beamforming or spatial filtering techniques.

C. Summary of Contributions

In this work, we presented a complete pipeline for emergency vehicle siren detection on embedded hardware, addressing both training and deployment challenges:

- We evaluated and fine-tuned the EPANNs architecture for EV detection, obtaining the specialized E2PANNs model.
- We designed and released curated datasets (AudioSet-EV, AudioSet-EV Augmented, Unified-EV) using a reproducible, taxonomy-aware Python toolkit.
- We implemented a full real-time pipeline on Raspberry Pi 5 using multithreaded inference, adaptive frame logic, and post-processing filters.
- We evaluated the whole model on a temporally-strong labeled subset of AudioSet, finding, by human inspection, a relevant portion of mislabelled positives.
- We implemented a live system that integrates a high-quality DAC + mic board and a WebSocket remote interface for interactive monitoring.

D. Future Work and Research Directions

Several enhancements are currently under consideration:

- Extension to multi-microphone setups enabling Direction-of-Arrival estimation and spatial filtering.
- Exploration of model distillation and quantization techniques to further reduce inference latency.
- Deployment in moving vehicles and noisy environments to validate system resilience in the wild.
- Broader class coverage for urban sound events, moving beyond binary classification.

The IoS-ready architecture opens pathways for advanced distributed applications:

- **Collaborative Detection Networks**: Multiple E2PANNs nodes can share acoustic fingerprints and detection confidence scores, enabling robust multi-point triangulation of emergency vehicles. Using edge computing federation, nodes can collaboratively reduce false positives through consensus mechanisms.
- **Smart City Integration**: The WebSocket interface enables direct integration with existing smart city platforms (FIWARE, Azure IoT Hub, AWS IoT Core), allowing emergency vehicle detections to trigger automated responses such as adaptive traffic signal control, public announcement systems, and emergency route clearance.
- **Acoustic Event Streaming**: By implementing Apache Kafka or similar event streaming platforms, our system can contribute to real-time acoustic maps of urban environments, where emergency vehicle movements are tracked alongside other sound events for comprehensive situational awareness.
- **Edge-Cloud Hybrid Processing**: While maintaining real-time edge detection, the system can selectively stream audio features to cloud services for advanced analytics, model updates, and long-term pattern analysis of emergency response times across the city.

These developments aim to support long-term goals of scalable, interpretable, and adaptive sound event detection in smart city infrastructures.

REFERENCES

[1] C. Castorena, M. Cobos, J. Lopez-Ballester, and F. J. Ferri, "A safety-oriented framework for sound event detection in driving scenarios," *Applied Acoustics*, vol. 215, p. 109719, Jan. 2024. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0003682X23005170

[2] A. E. Ramirez, E. Donati, and C. Chousidis, "A siren identification system using deep learning to aid hearing-impaired people," *Engineering Applications of Artificial Intelligence*, vol. 114, p. 105000, Sep. 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0952197622001890

[3] D. V K, A. P S, P. Jayakumar, and P. S, "A comprehensive review of smart emergency vehicle detection and response systems," in *2025 3rd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*, April 2025, pp. 1–5.

[4] U. Mittal and P. Chawla, "Acoustic Based Emergency Vehicle Detection Using Ensemble of deep Learning Models," *Procedia Computer Science*, vol. 218, pp. 227–234, 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1877050923000054

[5] T. Miyazaki, Y. Kitazono, and M. Shimakawa, "Ambulance Siren Detector using FFT on dsPIC," in *Proceedings of the 1st IEEE/IIAE International Conference on Intelligent Systems and Image Processing 2013*. The Institute of Industrial Applications Engineers, 2013, pp. 266–269. [Online]. Available: https://www2.ia-engineers.org/conference/index.php/icisip/icisip2013/paper/view/247

[6] F. Meucci, L. Pierucci, E. Del Re, L. Lastrucci, and P. Desii, "A real-time siren detector to improve safety of guide in traffic environment," in *2008 16th European Signal Processing Conference*, 2008, pp. 1–5.

[7] S. Giacomelli, M. Giordano, C. Rinaldi, and F. Graziosi, "From large-scale audio tagging to real-time explainable emergency vehicle sirens detection," version: 1. [Online]. Available: http://arxiv.org/abs/2506.23437

[8] A. Singh, H. Liu, and M. D. Plumbley, "E-PANNs: Sound Recognition Using Efficient Pre-trained Audio Neural Networks," May 2023, arXiv:2305.18665 [cs]. [Online]. Available: http://arxiv.org/abs/2305.18665

[9] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020, conference Name: IEEE/ACM Transactions on Audio, Speech, and Language Processing. [Online]. Available: https://ieeexplore.ieee.org/document/9229505

[10] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. New Orleans, LA: IEEE, Mar. 2017, pp. 776–780. [Online]. Available: http://ieeexplore.ieee.org/document/7952261/

[11] S. Giacomelli, M. Giordano, C. Rinaldi, and F. Graziosi, "AudioSet-tools: A python framework for taxonomy-aware AudioSet curation and reproducible audio research," ISSN: 2693-5015. [Online]. Available: https://www.researchsquare.com/article/rs-6957428/v1

[12] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound Event Detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, Sep. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9524590/

[13] M. Y. Shams, T. Abd El-Hafeez, and E. Hassan, "Acoustic data detection in large-scale emergency vehicle sirens and road noise dataset," *Expert Systems with Applications*, vol. 249, p. 123608, Sep. 2024. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417424004731

[14] S. Damiano, T. Dietzen, and T. van Waterschoot, "Frequency tracking features for data-efficient deep siren identification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE2024)*, Tokyo, Japan, Oct. 2024, pp. 36–40.

[15] F. Beritelli, S. Casale, A. Russo, and S. Serrano, "An Automatic Emergency Signal Recognition System for the Hearing Impaired," in *2006 IEEE 12th Digital Signal Processing Workshop & 4th IEEE Signal Processing Education Workshop*. Teton National Park, WY, USA: IEEE, Sep. 2006, pp. 179–182. [Online]. Available: http://ieeexplore.ieee.org/document/4041054/

[16] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proceedings of the 23rd ACM international conference on Multimedia*, ser. MM '15. New York, NY, USA: Association for Computing Machinery, Oct. 2015, pp. 1015–1018. [Online]. Available: https://doi.org/10.1145/2733373.2806390

[17] M. Fuentes, B. Steers, P. Zinemanas, M. Rocamora, L. Bondi, J. Wilkins, Q. Shi, Y. Hou, S. Das, X. Serra, and J. P. Bello, "Urban Sound & Sight: Dataset And Benchmark For Audio-Visual Urban Scene Understanding," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2022, pp. 141–145, iSSN: 2379-190X. [Online]. Available: https://ieeexplore.ieee.org/document/9747644

[18] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: An Open Dataset of Human-Labeled Sound Events," Apr. 2022, arXiv:2010.00475 [cs]. [Online]. Available: http://arxiv.org/abs/2010.00475

[19] A. Shah and A. Singh, "sireNNet-Emergency Vehicle Siren Classification Dataset For Urban Applications," Feb. 2023, publisher: Mendeley Data. [Online]. Available: https://data.mendeley.com/datasets/j4ydzzv4kb/1

[20] M. Usaid, M. Asif, t. rajab, P. D. E. S. Hussain, P. D. s. M. munaf, and S. Wasi, "Large-Scale Audio Dataset for Emergency Vehicle Sirens and Road Noises," 2022. [Online]. Available: https://figshare.com/articles/media/Large-Scale_Audio_Dataset_for_Emergency_Vehicle_Sirens_and_Road_Noises/19291472/2

[21] A. Shah, A. Singh, and A. Singh, "Audio Classification of Emergency Vehicle Sirens Using Recurrent Neural Network Architectures," in *Proceedings of International Conference on Paradigms of Communication, Computing and Data Analytics*, A. Yadav, S. J. Nanda, and M.-H. Lim, Eds. Singapore: Springer Nature Singapore, 2023, pp. 71–83, series Title: Algorithms for Intelligent Systems. [Online]. Available: https://link.springer.com/10.1007/978-981-99-4626-6_6

[22] S. Damiano, B. Cramer, A. Guntoro, and T. van Waterschoot, "Synthetic data generation techniques for training deep acoustic siren identification networks," *Frontiers in Signal Processing*, vol. 4, p. 1358532, 2024.

[23] M. Asif, M. Usaid, M. Rashid, T. Rajab, S. Hussain, and S. Wasi, "Large-scale audio dataset for emergency vehicle sirens and road noises," *Scientific Data*, vol. 9, no. 1, p. 599, Oct. 2022. [Online]. Available: https://www.nature.com/articles/s41597-022-01727-2

[24] J. Salamon, C. Jacoby, and J. P. Bello, "A Dataset and Taxonomy for Urban Sound Research," in *Proceedings of the 22nd ACM international conference on Multimedia*. Orlando Florida USA: ACM, Nov. 2014, pp. 1041–1044. [Online]. Available: https://dl.acm.org/doi/10.1145/2647868.2655045

[25] A. R. Chadha, R. Misal, and T. Mokashi, "Modified binary search algorithm," *CoRR*, vol. abs/1406.1677, 2014. [Online]. Available: http://arxiv.org/abs/1406.1677

[26] S. Hershey, D. P. W. Ellis, E. Fonseca, A. Jansen, C. Liu, C. Moore, and M. Plakal, "The benefit of temporally-strong labels in audio event classification," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 366–370, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:234742594

[27] N. S. Detlefsen, J. Borovec, J. Schock, A. H. Jha, T. Koker, L. D. Liello, D. Stancl, C. Quan, M. Grechkin, and W. Falcon, "TorchMetrics - Measuring Reproducibility in PyTorch," *Journal of Open Source Software*, vol. 7, no. 70, p. 4101, Feb. 2022. [Online]. Available: https://joss.theoj.org/papers/10.21105/joss.04101

[28] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification: An overview of dcase 2017 challenge entries," in *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2018, pp. 411–415.

[29] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.