A Real-Time Audio Monitoring Framework with Limited Data for Constrained Devices

Asif Salekin[1], Shabnam Ghaffarzadegan[2], Zhe Feng[2], John A. Stankovic[1]
University of Virginia[1], VA, USA and Bosch Research and Technology Center[2], CA, USA

***Abstract*—An effective and non-invasive audio monitoring system needs to be capable of simultaneous real-time detection of multiple audio events in many different environments, and locally executable on resource constrained devices, such as, smart microphones. A major challenge in this research domain is having limited available annotated data. This paper presents a novel framework to generate robust detection models of environmental and human audio events with limited available data. The framework presents the generation of a large synthetic dataset using limited data for any audio event, a novel computationally efficient feature modeling technique, named Audio2Vec, that is robust against environmental variations, and identifies and exploits the syntactic relation between audio states represented by the features and the targeted audio events. The presented framework achieves 10.3% higher F_1 scores compared to the best baseline approaches. To demonstrate the effectiveness of the framework we implemented a real-time audio monitoring system that simultaneously detects 10 audio events on a Raspberry Pi 3B and evaluate it in real home and in-car settings, that achieve F_1 scores of 0.96 and 0.956, respectively.**

I. Introduction

Automated event detection (AED) systems are becoming increasingly popular and important both in private and public environments. Most of the existing monitoring systems work based on video information. Video monitoring systems are not robust against conditions, such as, fog or low visible conditions, or obstacles. Moreover, due to theie invasiveness they are often not appropriate in private settings. While thermal infrared sensors can be a less invasive alternative, this technology is highly dependent on temperature, and the separation between background and foreground objects can be problematic. In contrast, audio as a monitoring or event detection modality has the following advantages: (1) needs fewer memory storage and computation requirements compared to video streams, hence it is more appropriate for executing on resource constrained devices; (2) unlike cameras, microphones are omnidirectional with no angular limitations; (3) audio event detection (AED) is robust against many environmental obstacles; (4) AED is robust against illumination and temperature; (5) many events have distinctive audio signatures, but little or no video counterpart; and (6) audio-based monitoring systems that perform all the computations locally are potentially more privacy friendly.

In recent years smart technologies such as smart homes, smart cars, home health monitoring and surveillance systems, have become popular among consumers. Smart speakers, such as, alexa, google home, come with built in microphones. In most modern cars a microphone already exists in the cabin. Hence, a real-time AED system, capable of running locally on resource constrained devices, such as, Raspberry Pi, can be a great real-time monitoring solution, and can be added to already existing smart home and smart car devices.

Fig. 1: *Framework for real-time AED with limited data.*

Though there are some available datasets [4], [20], [21] which contain event level annotation for automated monitoring systems, the amount of labeled event data is very small. Hence, the majority of AED studies [6], [27] perform their evaluation on small datasets. A limited dataset in training leads to lack of robustness of the AED approach as they are used in different environments (with noise and a large variety of extraneous sound events).

This paper presents a novel framework for AED, which generates robust models for audio monitoring applications with limited available data. Moreover, the generated AED systems are real-time executable on resource constrained devices. The main characteristics and contributions of the framework and its evaluation are:

- As shown in figure 1, for each of the audio events with limited data, the framework generates a large synthetic dataset with a large variation of background environmental sounds, signal to noise ratios (SNRs), and reverberation effects. Theoretically, the presented **automated audio mixture synthesizer** (section III) can generate an infinite number of variations.
- To extract meaningful and effective feature representation from the raw audio data, this paper presents a novel **computationally effective feature modeling technique, named Audio2Vec** (section IV-A & IV-B). The generated representations by Audio2Vec are robust against environmental noise, reverberation, and de-amplification of sound due to distance. Moreover, it identifies and exploits the inherent relation between audio states and targeted audio events. As a result, Audio2Vec features can be used with much shallower (less layers & network parameters) neural network classifiers, and achieves significantly higher accuracy compared to the baseline feature representations typically used with much deeper neural networks. Also, shallow networks (for classification) have less execution time which makes them more suitable for real-time AED systems on resource constrained devices.
- To demonstrate the extensive applicability of the presented AED framework, we applied the framework (figure 1) to develop and evaluate audio detection models for ten environmental and human audio events: crying, laughter, screaming, coughing, snoring, brushing teeth, sneezing, baby crying, glass breaking, and gunshots. One example of the value of detecting multiple audion events is that

according to the Cohen-Mansfield Agitation Inventory [3], detection of crying, laughter, and screaming are important for monitoring agitation in dementia patients. Also automated detection of coughing, snoring, brushing teeth, sneezing have application in home healthcare and hygiene monitoring. Detection of baby crying is important for infant monitoring systems, and gunshots and glass breaking have application in automated surveillance and security systems. Our AED approach using the Audio2Vec feature representation achieved on average 10.3% higher F_1 score compared to the best baseline approach for 10 targeted audio events (section V-B).

- To evaluate the applicability of our approach in realistic scenarios, running on resource constrained devices (on-device or local computations), we implemented an **real-time AED system** (that simultaneously detects the 10 targeted audio events) on a Raspberry Pi 3B with a MATRIX Creator development board (section V-C). We evaluated the implemented system for two realistic applications: real homes and inside car monitoring. According to the evaluation we achieved average F_1 scores of 0.96 and 0.956 for AED in real-home and in-car settings, respectively.
- An effective monitoring system needs to be real-time executable. We experimentally evaluated the CPU runtime of each component of our AED system and demonstrated its real-time capability for a constrained device: Raspberry Pi (section V-D).

II. Related Works

Several combinations of features and classifiers have been investigated for AED tasks. From low level audio features, such as Zero Crossing Rate (ZCR) [1], to middle level features, such as Mel-Frequency Cepstral Coefficients (MFCC) and Perceptual Linear Predictive (PLP) [17], [22], and to high level feature descriptors, such as MPEG-7 [17], these different features and their combination have been used to represent acoustic events.

Clavel et al. [2] propose a Gaussian Mixture Model (GMM) and Maximum A Posteriori (MAP) decision rule-based gunshot detection approach using short-time energy, Mel-Frequency Cepstral Coefficients (MFCC) and spectral statistical moments as features. Vacher et al. [28] also adopt a GMM classifier, with wavelet-based cepstral coefficients as features, for the detection of screams and broken glass. Rouas et al. [24] use MFCC features and a combination of the GMM and Support Vector Machine (SVM) classifiers for detecting screams in outdoor environments. Their method uses an adaptive thresholding on sound intensity for limiting the number of false detections. Sharan et al. [25] evaluates the audio event detection performance of classification techniques for multi-class support vector machines in various noise conditions.

Peter et al. [27] present a car AED approach using MFCC features that was evaluated on 100 audio clips for each of the events with different signal to noise ratios. Hussein et al. [6] introduce an AED for smart homes exploring energy, pitch, ZCR, spectral features, MFCC features and K-nearest neighbor (KNN) and SVM classifiers. Nandwana et al. [16], utilized GMM, GMM mean super-vectors and the I-vector framework for an in car AED. This work was evaluated on 100 audio clips for each of the audio events. Our evaluation considers this I-vector representation as one of the baseline feature representation approaches to which we compare. All these works evaluate on limited acted datasets. Morfi et al. [15] used weakly supervised learning to detect audio events from weakly labeled training data. In contrast, the focus of this paper is to use available strong labeled audio data to generate a robust real-time AED solution.

Recently, research on audio event detection studies are shifting from conventional methods to modern deep neural network approaches [9], [13]. Several studies [8], [18], [26] have used Convolutional Neural Networks (CNN) with a large input field to identify the presence of an audio event in a large audio clip. CNN learns filters that are shifted in both time and frequency, hence can cover a large input field. Our evaluation considers CNN as one of the baseline classifier approaches.

In recurrent neural networks (RNNs), information from previous time steps can in principle circulate indefinitely inside the network through the directed cycles, where the hidden layers also act as memory. Hence, an RNN can capture comparatively long temporal context information. Parascandolo et al. [19] propose an audio tagging approach based on bi-directional long short term memory (BLSTM) and evaluate 60 different sound events detection in 103 real life recording clips. Marchi et al. [11] and Wang et al. [29] also apply an LSTM classifier on limited audio data for AED. Our evaluation considers a bi-directional LSTM as one of the baseline classifier approaches.

III. Synthetic Dataset

A robust audio event monitoring system needs models that perform well in various environments not introduced in the training phase. Also, in real event monitoring scenarios input audio signal to noise ratio (SNR) can be very low due to variable source to microphone distances and presence of other ambient noise sources. Additionally, in indoor settings, audio data suffers from reverberation effects. Training a supervised model robust against unknown environments, reverberation, and low SNR requires sufficiently large dataset with variation of environmental sounds, reverberation, and SNRs. One of the challenges this paper addresses is having small available audio event datasets. But there are larger environmental scene datasets, where, background sound is collected from many different environments, such as, cafes, train stations, or parks. These sounds are easy to collect and do not need any labeling of the data. This paper presents an audio mixture synthesizer, that generates a large synthetic mixture of labeled isolated audio event clips and various environmental audio clips. Using this automated generalized approach, it is possible to generate any number of well labeled positive and negative synthetic data samples (this can be applied to any audio event with a small available dataset).

Solution: Audio mixture synthesizer: We mix isolated audio clips (from available small datasets) with environmental background sounds to generate synthetic data samples. For the

mixture synthesizer we used the pydub python toolkit. Synthetic audio samples are generated in the following manner.

We randomly select 10s audio from a randomly selected environmental background audio clip. Additionally, we randomly select one or more isolated (targeted and/or non-targeted) audio event clips. Both the environmental background audio and the isolated event audio clips are amplified or de-amplified to generate a random event-to-background ratios (EBR) between -6 to 6 dB. Then isolated audio event clips are overlaid on the 10s background audio clips at randomly selected positions. Random numbers are drawn from a uniform distribution, to achieve maximum variation in background sounds, EBR, and event positions.

In a 10s synthetic positive audio sample of a targeted audio event (e.g., screaming), at least one (or more) isolated targeted audio clips are overlaid/placed, and zero or more non-targeted audio clips are overlaid/placed on the same 10s environmental sound. In a 10s synthetic negative sample of a targeted audio event, any other (except the targeted event) or none of the isolated audio clips are overlaid/placed in the 10s audio.

There are different artificial reverberation effect parameters to model how sound waves reflect from various types of room size and characteristics. Our synthesizer introduces different combinations of reverberation parameters: wet/dry ratio, diffusion, and decay factor in the generated audio samples. These parameters generate different reverberated signal to the original signal ratio, discrete echo effects, and reverberation tails (decay factor of reverberation).

The generated synthetic data is highly imbalanced (due to significantly larger number of non-targeted audio clips), that can make the binary AED classifiers biased toward the majority (i.e., negative) class. To address the data imbalance issue, we perform cluster-based under-sampling [30] on the negative (non-targeted) 10s audio clips to generate an equal number of positive and negative synthetic data samples for each of the targeted audio events, as well as to accommodate the variations of the negative or non-targeted audio samples.

IV. Audio Event Detection

Our presented approach performs the AED task on 10s audio clips. We developed a novel feature modeling approach named Audio2Vec (section IV-B) to extract robust and effective feature representations of audio data. Binary classifiers for each of the targeted events (section IV-C) take the Audio2Vec features as input and perform the classification task.

A. Audio Features

Our approach segments the generated 10s audio clips into overlapping windows (200ms with 20% overlap in our evaluation), and extracts a feature set from each window. The extracted feature set, represents the inherent state of audio from that window. Based on the previous studies on acoustic features associated with targeted audio events (i.e., baby crying, gun shots, glass breaking, screaming) we considered low-level descriptor (LLD) features shown in left column of Table I, as well as their delta and delta-delta coefficients. Each window is segmented into overlapping 25ms frames with 10 ms overlap, from which LLD features are extracted. Next the 8 functionals: minimum, maximum, mean, median, standard deviation, variance, skew and kurtosis, are applied to extract the audio window representation. In total 272 raw features are extracted from each of the 200ms windows, where 10s audio clips consist of 62 overlapping windows (200ms).

TABLE I: *Raw audio features*

LLD Features	High level features (functionals)
Zero crossing rate & Δ (2-dim)	Min, Max
Energy & Δ (2-dim)	std, skew, var
Spectral centroid & Δ (2-dim)	kurtosis, mean
Pitch & Δ (2-dim)	median
MFCC & Δ (26-dim)	

B. Feature Modeling

The modeling stage of an audio analysis system develops a representation that reflects the audio information for that specific task. Each segment of audio represents a state, and an audio event is represented by the progression of audio signals through various states. The following sections introduce a robust novel representation of the audio state from a 200ms window that takes into account the inherent notion of that state with a particular targeted audio event.

1) Audio to Word: We use the Audio-Codebook model [23] to represent the audio signal from windows with 'audio words'. The 'audio words' represent the state of audio in each 200ms window. In our context the audio-codebook words are fragments of audio signal represented by features. We need a robust feature descriptors to represent the audio state in an audio window. Inspired by [7], we use a GMM based clustering method to generate the audio codebook from the functional representations mentioned in section IV-A.

In the codebook generation step, a GMM based model is trained on randomly sampled data from the training set. The resulting clusters form our codebook audio words. Once the codebook has been generated, acoustic features within a certain range of the audio signal are assigned to the closest audio words (cluster centers) in the codebook. In the experiments, we have evaluated with different sizes of codebooks.

The raw audio features from section IV-A distort up to a certain level with variance of environmental noise, audio de-amplification, and reverberation. Our trained audio-codebook places similar points in the feature space into the same code words, which make the feature representation robust against different environments, noise, and distance to the microphone.

2) Audio2Vec Approach: Audio words extracted from overlapping 200ms windows represent the states of the audio. Since, audio signals from a particular audio event are different from others, audio states representing that event would be different from others. Also, some states occur more frequently in a targeted audio event signal compared to other events. Typical audio codebook word methods fails to convey this state to an audio event representation.

To identify and exploit the inherent relation between audio states and audio events, we developed a novel audio word to vector conversion (Audio2vec) approach, that generates an N dimensional vector representation for each of the audio words (from the Audio codebook). Position of a generated vector in

the N dimensional vector space, depicts the relation between the state it represents and the targeted audio event.

Algorithm 1 shows our Audio2Vec conversion approach. In the initialization stage audio words unique for a targeted audio event are randomly assigned vectors near the $Positive_{centre}$ vector in an N dimensional vector space and words which never occur in the targeted events are assigned random vectors near $Negative_{centre}$. Other audio words are randomly assigned vectors between them ($Positive_{centre}$ and $Negative_{centre}$). In the iteration stage, every time an audio word w_j occurs in the targeted event, the vector representation of that audio word v_j is modified according to equation 1, which makes v_j move closer to $Positive_{centre}$ in the N dimensional vector space. Otherwise v_j is modified according to equation 2, which makes v_j move closer to $Negative_{centre}$ (line 18-24, Algorithm 1).

$$v_j \leftarrow v_j + (Positive_{centre} - v_j) \times \delta_p \quad (1)$$

$$v_j \leftarrow v_j + (Negative_{centre} - v_j) \times \delta_n \quad (2)$$

Algorithm 1 Audio2Vec Algorithm

w: audio word
v: Audio2Vec vector
C: audio codebook
A: audio clip represented as a sequence of n audio words; $A = \{w_1, w_2, \ldots, w_n\}$
T: set of audio clips for training
δ_p: small constant used as parameter
δ_n: small constant used as parameter
m: dimension of generated Audio2Vec vector
$iter$: number of iterations

1: **procedure** AUDIO2VEC ALGORITHM(T, C, δ_p, δ_n, $iter$)
2: **#Initialization:**
3: $Positive_{centre} \leftarrow [k_1 k_2 \ldots k_m]$ where $k \in \{0.8, 1\}$
4: $Negative_{centre} \leftarrow [k_1 k_2 \ldots k_m]$ where $k \in \{0, 0.2\}$
5: **for** all audio word $w_i \in C$ **do**
6: **if** w_i only appears in targeted audio event **then**
7: $v_i \leftarrow [k_1 k_2 \ldots k_m]$ where $k \in \{0.8, 1\}$
8: **end if**
9: **if** w_i never appears in targeted audio event **then**
10: $v_i \leftarrow [k_1 k_2 \ldots k_m]$ where $k \in \{0, 0.2\}$
11: **end if**
12: **if** w_i appears in targeted and other events **then**
13: $v_i \leftarrow [k_1 k_2 \ldots k_m]$ where $k \in \{0.4, 0.6\}$
14: **end if**
15: **end for**
16: **loop**: $iter$ times
17: **for** all audio clip $A_i \in T$ **do**
18: **for** audio word $w_j \in A_i$ **do**
19: **if** $w_j \in$ targeted audio event **then**
20: $difference \leftarrow Positive_{centre} - v_j$
21: $v_j \leftarrow v_j + difference \times \delta_p$
22: **else**
23: $difference \leftarrow Negative_{centre} - v_j$
24: $v_j \leftarrow v_j + difference \times \delta_n$
25: **end if**
26: **end for**
27: **end for**
28: **end loop**
29: **end procedure**

Since, the targeted audio events are only few seconds in the 10s audio samples, the total number of audio words that appear in the targeted audio event segments are significantly lower compare to total number of audio words in other audio event segments. To mitigate this bias, we calculate the addition fraction parameter for negative samples δ_n according to equation 3, where N_p and N_n are the total number of audio words in the targeted and other events, respectively.

$$\delta_n \leftarrow \frac{N_p}{N_n} \times \delta_p \quad (3)$$

Figure 2 shows an example of the Audio2Vec approach, where the vector dimension is N=2. In Figure 2 (a), black points are the vectors (audio words) unique for targeted events, white points are ones that never occur in the targeted events, and the grey ones are common between two classes. Later, in the iterative stage of Audio2vec, every time an audio word occurs in the targeted event in training set, the vector representation of that audio word is moved closer to the targeted event clusters in the vector space, as shown in Figure 2 (b). Similarly, if an audio word occurs for any other events, the vector representation of that audio word is moved further from the targeted event clusters (figure 2 (c)). As shown in Figure 2 (d), the Audio2vec approach brings frequently occurring words in the targeted events closer in the vector space compared to others.

The advantages of the Audio2Vec approach:

Representational efficiency: Audio2Vec learns to map audio data from each 200ms windows into a fixed-length low-dimensional continuous vector space from their distributional properties observed in training. Our evaluation has found that the best Audio2Vec dimension is N=30. Hence, Audio2Vec approach represents the audio windows by a significantly low-dimensional distributed representation. Classifiers that take lower dimensional input data can optimize their parameters more effectively when training data is limited.

Mapping Efficiency: An interesting property of Audio2Vec vectors is that they encode the syntactic relationships between audio states and targeted audio events (classes). Audio2Vec vectors are similar for audio states with similar probability of occurring in targeted audio events. These characteristics are similar to the output of convolution layers in a CNN. Each convolution layer generates a successively higher level abstraction of the input data, which preserves essential yet unique information. Deep CNNs extract meaningful feature representations (also in the form of vectors) from input data by employing a deep hierarchy of layers. For example, the best baseline CNN classifier (section V-B) has 4 convolution layers, with, in total, 140501 network parameters. Training such a high number of parameters needs a large training set. One of the challenges of this study is having a small amount of training data. The advantage of Audio2Vec approach is, its vector generation process involves vector addition and subtractions observing the training examples and is not constrained by

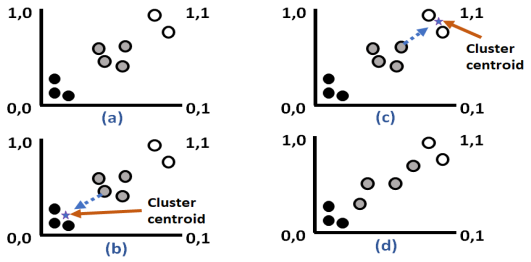


Fig. 2: Example of Audio2vec approach in 2 dimensional space.

number of training instances. Hence, the Audio2Vec approach can generate effective feature representations for our targeted application with limited training examples.

Execution efficiency: The Audio2Vec approach performs the clustering operation (to calculate Audio-Codebook) during training and stores the audio-words to Audio2Vec vectors conversion maps in a dictionary. Hence, during execution, converting raw audio from 200ms windows to Audio2Vec vectors involves finding the nearest cluster centroid and a dictionary lookup, which are linear in complexity and require significantly fewer computations compared to the baseline deep learning approaches.

C. Classifier

In this study we evaluated with Convolutional Neural Network (CNN), Bi-directional Long Short-Term Memory (BLSTM), and Deep Neural Network (DNN) (i.e., 'vanilla' Neural Network) as classifiers. For each of these classifiers we performed a grid search on the network parameter values. For each of the classifiers, our evaluation iterate on 1 to 10 layers, with 50 to 500 neurons/filters (for convolution layers) for each of the layers. Due to the limitation of space, each of the classifiers with only the best iterated parameter combinations on the training set is presented in the evaluation section.

V. Evaluation

Our evaluation consists of three parts. First, section V-A & V-B discuss our synthetic dataset generation for 10 targeted audio events, and the performance of our Audio2Vec AED approach on the generated data. Later in section V-C, we evaluate our AED approach for two realistic applications: in-home and in-car audio monitoring systems. And, in section V-D, we evaluate the CPU runtime of each component of our system to demonstrate it's real-time capability.

A. Synthetic Data Generation

Our presented system detects ten audio events. We collected 160 isolated audio clips for each of these targeted events (100 for training, 30 for testing, and 30 for validation) from the freesound dataset [5], the ESC environmental sound Dataset [21], and the MIVIA audio event dataset [4].

To train a robust audio detection classifier, we need a training dataset that contains a large variation of non-targeted events that may occur in the real scenarios. Hence, we collected isolated audio clips of 80 environmental and human events (40 clips for each category) from the freesound dataset [5] and the ESC Dataset [21]. These events include environmental events such as, rain, sea waves, birds, water drops, wind, pouring water, car horn, helicopter, siren, engine, train, bells, fireworks, and human sounds such as, clapping, breathing, footsteps, drinking, sipping, dish washing, and animal sounds such as a dog, rooster, cow, cat, insects, crow, etc. Additionally, we collected 400 clips of human speech with different emotions (happy, angry, sad) from the EMA speech dataset [10]. We use this large variation of non-targeted isolated audio clips to generate negative training samples.

Collected isolated sound clips (targeted and non-targeted) have exact labels with sampling frequency 44.1 kHz or higher. The duration of the audio clips varied from 0.5 to 4.3 seconds.

To introduce a large variety of environmental background sounds we collected 1121 background audio clips from the TUT Acoustic Scenes development dataset [14]. This environmental scene dataset contains 15 acoustic scenes, including audio clips recorded from bus, train, cafe, car, city center, forest, store, home, beach, metro stations, office, park.

Using our mixture synthesizer (section III), we generate 2000, 10s audio clips (1000 for training, 500 for validation and 500 for testing), for each of the targeted audio events. Isolated audio event clips and environmental background audio data for training, validation, and testing datasets were disjoint. We perform 10-fold cross-validations on the training and testing sets to select the best models and model parameters that fit the data. We use the validation set to report the AED performance on the synthetic dataset.

Due to the highly imbalanced data in the evaluation phase of the study, using accuracy as the evaluation metrics can introduce an accuracy paradox. Hence, we used class-wise F_1 score, the harmonic average of precision, and recall as the evaluation metrics.

$$F_1 = \frac{2 \times precision \times recall}{precision + recall} \tag{4}$$

B. Evaluation Results: synthetic data

In this section, we describe the binary audio event detection evaluation results on generated synthetic data (section V-A). As mentioned in section IV-A, each 10s audio clips is segmented into 62, 200ms windows and 272-dimensional feature-set is extracted from each of these windows. Our Audio2Vec feature modeling approach converts these 272 dimensional raw feature-sets to N dimensional vectors. As the first step of Audio2Vec approach, we generate 1000 GMM clusters from randomly selected audio samples for each of the targeted audio events. The resulting clusters form the generated audio words (section IV-B1). The proposed Audio2Vec solution generates 30-dimensional vector representation for each of the extracted audio words as described in Algorithm 1. We performed a grid search over 500 to 3000 cluster sizes and 10 to 100 dimensions to find the best values of cluster size (1000) and Audio2Vec vector dimension (N=30) for our AED task.

Figure 3 illustrates the change of generated Audio2Vec feature vector space, for the audio event: gunshot with the increase of iteration number of Algorithm 1. PCA based visualization approach is used here to visualize the 30-dimensional feature space in 2-dimensions. In this figure, the blue points represent the audio words unique for gunshot signals, red points represent the ones that never occur in gunshot signals, and green points represent the ones which are common. At the initial step of Algorithm 1, green points are clustered in the middle between red and blue points (Figure 3(a)). With the increase of iterations, these points spread out, and as points occur more frequently in gunshot signals they move toward blue points' cluster and the points that occur frequently in other audio signals move towards the red points' cluster. This visualization demonstrates how Audio2vec approach brings

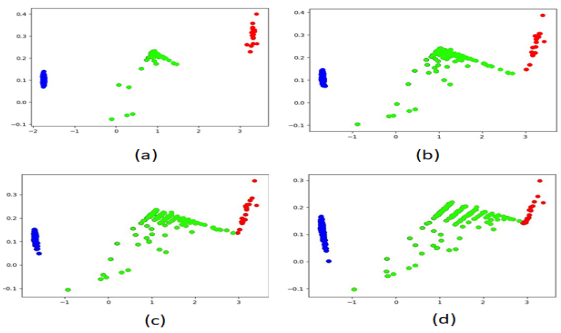
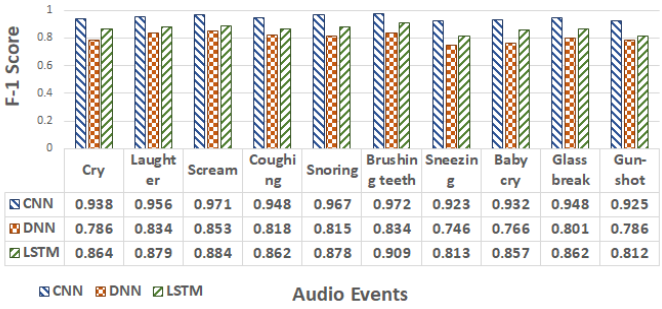


Fig. 3: *Iterations of Audio2Vec vector generation approach for gun shots. Figure (a), (b), (c), and (d) show the generated vector feature space after initial, 5,10 and 20 iterations of Algorithm 1.*



	Cry	Laughter	Scream	Coughing	Snoring	Brushing teeth	Sneezing	Baby cry	Glass break	Gun-shot
CNN	0.938	0.956	0.971	0.948	0.967	0.972	0.923	0.932	0.948	0.925
DNN	0.786	0.834	0.853	0.818	0.815	0.834	0.746	0.766	0.801	0.786
LSTM	0.864	0.879	0.884	0.862	0.878	0.909	0.813	0.857	0.862	0.812

Fig. 4: *Evaluation of classifiers with Audio2Vec features.*

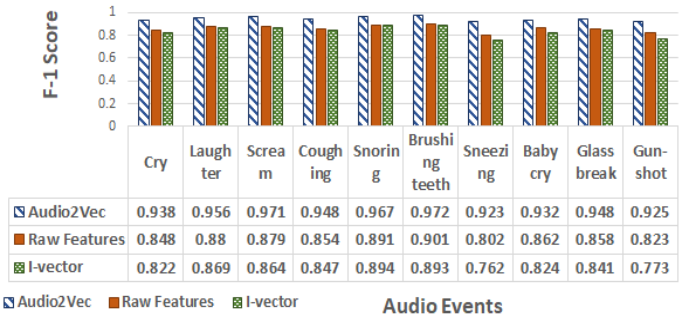
frequently occurring audio words in the targeted event (gun-shot) closer in the vector space compared to others.

Figure 4 shows the evaluation (class-wise F_1 scores) on different deep learning classifiers for our AED task. In this evaluation all three classifiers take extracted 62×30 dimensional Audio2Vec features (from 10s audio) as input. We performed a grid search on the network parameters to identify the best combination. The CNN implementation had 2 convolution layers [60,60], each with 60 convolution kernels (temporal extension of each filter 2), a ReLU activation function, 20% dropout rate and max pooling (window size 2 and down-scaling factor 2). Two fully connected dense layers [20,1] with sigmoid activation function, were attached, which make binary event presence decision. The CNN classifier achieved an average F_1 score of 0.948 for the ten targeted events.

The BLSTM implementation had two layers with [100, 100] nodes, a 20% dropout rate, and two fully connected dense layers [20, 1] with sigmoid activation function. The BLSTM classifier achieved an average F_1 score of 0.862 for the ten targeted events. DNN implementation had four fully connected layers with [500, 500, 300, 100] nodes and a ReLU activation function, a 20% dropout rate, and one fully connected dense layer [1] with a sigmoid activation function to make binary decisions. The DNN classifier achieved an average F_1 score of 0.8039 for the ten targeted events.

Our targeted audio event duration varied between 0.5s to 4.3s. Hence, the BLSTMs ability to convey contextual information in long audio sequence was not very advantageous. In CNNs, the convolutional filters not only can generate meaningful feature representations, moreover, they are translation invariant. That means, during training the convolution filters are being applied in a sliding window fashion on the entire 10s audio. Hence, no matter where an audio event occurs in a 10s audio clip, CNNs can detect it with limited training examples.

The CNN classifier achieved 9.9% and 17.9%, higher F_1 scores compared to the BLSTM and DNN classifiers. According to this evaluation we achieved the highest F_1 scores of 0.971 and 0.972, for events screaming and brushing teeth. The detection of gun shots and sneezing were most difficult since, different environmental sounds (from TUT Acoustic Scenes) are very similar to them, especially after the de-amplification and reverberation effects.



	Cry	Laughter	Scream	Coughing	Snoring	Brushing teeth	Sneezing	Baby cry	Glass break	Gun-shot
Audio2Vec	0.938	0.956	0.971	0.948	0.967	0.972	0.923	0.932	0.948	0.925
Raw Features	0.848	0.88	0.879	0.854	0.891	0.901	0.802	0.862	0.858	0.823
I-vector	0.822	0.869	0.864	0.847	0.894	0.893	0.762	0.824	0.841	0.773

Fig. 5: *Evaluation on features.*

To evaluate the effectiveness of Audio2Vec representation, we analyze the performance of AED with three different feature representations: 1) Audio2Vec, 2) I-vector, and 3) raw acoustic feature-set discussed in Section IV-A. We performed a grid search on the three classifiers (CNN, BLSTM, and DNN) parameters to identify the best classifiers. As shown in figure 5, raw feature-set with a CNN implementation achieved an average F_1 score of 0.8598. The CNN implementation had 4 convolution layers, each with 100 convolution kernels (temporal extension of each filter 4), a ReLU activation function, a 20% dropout rate and max pooling (window size 2 and down-scaling factor 2). Two fully connected dense layers [100,1] with a sigmoid activation function, are attached, which makes the binary event presence decision.

The I-vector system is a technique to map the high-dimensional GMM super vector space to low-dimensional space called total variability space. The basic idea of using I-vector representation is to represent each 200ms windows using concatenated I-vector feature vectors extracted based on audio event-specific GMM super vectors, and then to use these in the classifier. However, the existence of noise and channel variation can substantially affect the performance of i-vector representations. Since, environmental background sounds for training and validation data in our generated synthetic dataset are disjoint and the EBR varied between -6 to 6 dB, the i-vector representation fails to achieve a better AED performance. I-vector features with a CNN implementation achieved an average F_1 score of 0.839.

According to this evaluation the AED with our Audio2Vec features achieves 10.3% higher F_1 score compared to the best baseline features.

C. Evaluation on Realistic Applications

The MATRIX Creator [12] is an all-inclusive development board that connects to the GPIO pins on the Raspberry Pi. It has an 8-microphone MEMS array and an ARM Cortex

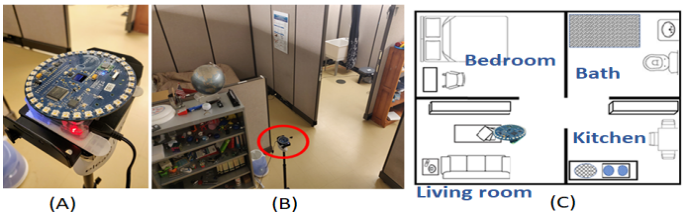


Fig. 6: *Realistic data collection.*

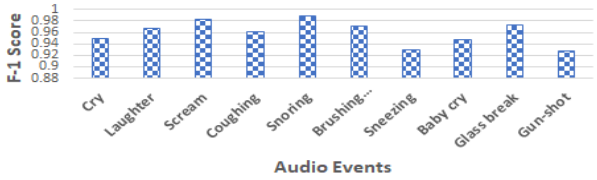


Fig. 7: *Evaluation in real home scenario.*

M3 microcontroller and features built-in noise cancellation and beamforming. We used the MATRIX Creator with Raspberry Pi 3B, as our scalable AED device (shown in figure 6-A).

We evaluate our AED approach for two realistic applications: inside car and real home audio event monitoring. For home event monitoring evaluation, we collected audio data from a pseudo smart home (figure 6-B,C) setup in the Smart Home Lab at the University of Virginia, and from a real one-bedroom apartment. In both settings we placed the AED device in center of the room. Since performing some events in real home or car settings were not feasible (such as for gun shots), we played sounds of targeted events through a Sony SRS-XB10 Bluetooth speaker. The speaker was placed in different places of the bedroom, bath, kitchen and living room. We collected data for a single day where a single occupant performed daily in-home activities, and different events were played at random times from random places. In total, we collected 50 audio examples for each of the 10 targeted audio events from each of the home settings.

Figure 7 shows our evaluation on real home data. In this evaluation the AED approach achieved an average F_1 score of 0.96 for the ten targeted events, that is 1.2% higher compared to our evaluation on the synthetic dataset (section V-B). This is due to the significantly less noise and variations of non-targeted audio events, compared to some challenging pseudo scenarios (section V-A), such as, trains, cafes, city center. According to the evaluation gun shot detection was most difficult since, different in-home environmental sounds, such as, knocking on the door, jumping, walking on a wooden floor, are very similar to de-amplified gunshot sounds.

For the inside car event monitoring evaluation, we collected audio data from a Toyota Corolla 2016 car for 3 different speed ranges (below 25, 25 to 45, and above 45 MPH) with 2 different conditions, AC on and AC off. The AED device was placed in the center of the car, and different sound events were played from a Sony speaker placed in four passenger seat positions. For each of the 6 conditions we collected 20 audio examples for each of the 10 targeted audio events from each passenger seat positions. Additionally, we collected audio samples without the presence of any of the targeted events.

All the audio examples of targeted and non-targeted events used for this evaluation (in-home & in-car) were not included in the synthetic dataset (section V-A).

TABLE II: *AED evaluation in real car scenario.*

Speed (MPH Condition)	0-25		25-45		45-65	
	AC on	AC off	AC on	AC off	AC on	AC off
Cry	0.954	0.946	0.954	0.948	0.94	0.938
Laughter	0.971	0.971	0.967	0.951	0.952	0.951
Scream	0.986	0.986	0.983	0.974	0.98	0.964
Coughing	0.968	0.968	0.964	0.961	0.948	0.942
Snoring	0.991	0.991	0.991	0.98	0.971	0.971
Brushing teeth	0.983	0.983	0.98	0.98	0.976	0.976
Sneezing	0.942	0.93	0.938	0.926	0.91	0.895
Baby cry	0.944	0.942	0.94	0.924	0.928	0.901
Glass break	0.989	0.982	0.971	0.971	0.961	0.942
Gunshot	0.957	0.957	0.942	0.928	0.901	0.896

Table II shows our evaluation on real car. These results are comparable to our evaluation in section V-B. At low car speeds, the AED approach achieved an average F_1 score of 0.9685. That is due to the significantly less noise and the absence of de-amplification on event sounds (All passenger seat positions are close to the AED device). Though at high speeds, the AED performance drops to 0.946 and 0.937 average F_1 scores on AC-on and AC-off conditions. The humming sound of AC, reduce the effect of noises in the car, hence, AC-on condition performed better for most of the cases.

D. CPU Time Benchmarking for Real Time execution

We did all our realistic application experiments on a Raspberry Pi 3B having a Quad Core 1.2GHz Broadcom BCM2837 64bit CPU and 1GB LPDDR2 (900 MHz) memory. Our program reads a 10s audio file at a time, and extracts 272-raw features. For each of the (10) targeted events a process reads the raw feature-set, converts it to an Audio2Vec representation, and performs classification through the CNN implementation described in section V-B. We performed the multiprocessing tasks through Python multiprocessing package.

We benchmark the computation time for (1) extracting 272-dimensional raw feature-set from 10s audio, (2) combined time for 10 processes to convert raw features to respective Audio2Vec representations, and (3) combined time for 10 processes to perform classification, as shown in table III. Moreover, we implemented the best baseline approach from section V-B: Raw audio features with a 4-convolutional layer CNN implementation in the Raspberry Pi 3B, with the similar multiprocessing approach. As shown in table III, for the baseline approach we benchmark the computation time for (1) extracting 272-dimensional raw feature-set, (2) combined time for 10 processes to perform classification taking raw features. Computation time is the time spent running the particular task plus running OS code on behalf of the task.

According to table III, reading audio files and extracting raw feature-set takes high CPU execution times, since they involve I/O operations. Audio2Vec vector conversion takes only 0.022s. The 2-convolutional layer CNN implementation used in our AED approach has $38,561$ parameters, and cumulative classification time takes 0.215s. Hence, the cumulative time taken by the 10 processes is 0.237s. And end-to-end total AED system time for one 10s audio is 4.557s. Given the AED window is 10s, the real-time system is extendable to including many more audio events. The baseline CNN implementation has $150,601$ parameters, and cumulative classification time takes 0.338s. Hence, the cumulative time taken by the 10

TABLE III: *Computation time for various system tasks*

Audio2Vec AED approach				
Task	Reading audio and extracting raw features	Audio2Vec conversion (cumulative)	Classification (cumulative)	Total time
Time (sec)	5.32	0.012	0.215	5.547
Baseline: Raw features+CNN				
Task	Reading audio and extracting raw features	Classification (cumulative)		Total time
Time (sec)	5.32	0.338		5.658

processes (one for each events) in the baseline implementation is 42.6% higher compared to the Audio2Vec AED approach.

According to this evaluation, our presented AED approach is capable of real time execution on a resource constrained device. Note that CPU times reported in table III are when only the audio event detection program is running. Running additional programs will effect/change these times.

VI. Conclusion

This paper presents a novel framework for robust AED models generation using limited available data. The framework uses a novel audio mixture synthesizer to generate a large synthetic dataset, that contains a large variation of background environmental sounds, noise, SNR, and reverberation effects; a novel robust and computationally effective feature representation technique, named, Audio2Vec. Due to the meaningful syntactic characteristics of the extracted feature representations, AED with Audio2Vec, performs significantly better with shallow network models, compared to much deeper models with baseline features. To demonstrate the applicability of the framework, we implemented a real-time AED system in a Raspberry Pi 3B and evaluated its performance in real home and in-car settings, that achieved F_1 scores of 0.96 and 0.956, respectively. Moreover, we experimentally evaluated the CPU runtime of the AED system to demonstrate its on-device real-time capability for a constrained device. Our framework is extendable to any other audio events and the real-time AED system is extendable to include many more audio events.

References

[1] Pradeep K Atrey, Namunu C Maddage, and Mohan S Kankanhalli. 2006. Audio based event detection for multimedia surveillance. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, Vol. 5. IEEE, V–V.

[2] Chloé Clavel, Thibaut Ehrette, and Gaël Richard. 2005. Events detection for an audio-based surveillance system. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*. IEEE, 1306–1309.

[3] Jiska Cohen-Mansfield. 1991. Instruction Manual for the Cohen-Mansfield Agitation Inventory (CMAI). *Research Institute of the Hebrew Home of Greater Washington* (1991).

[4] Pasquale Foggia, Nicolai Petkov, Alessia Saggese, Nicola Strisciuglio, and Mario Vento. 2015. Reliable detection of audio events in highly noisy environments. *Pattern Recognition Letters* 65 (2015), 22–28.

[5] Frederic Font, Gerard Roma, and Xavier Serra. 2013. Freesound technical demo. In *Proceedings of the 21st acm international conference on multimedia*. ACM, 411–412.

[6] Hussein Hussein, Marc Ritter, Robert Manthey, Jan Schloßhauer, Etienne Fabian, and Manuel Heinzig. Acoustic Event Classification for Ambient Assisted Living and Healthcare Environments. In *Proceedings of the 27th Conference on Electronic Speech Signal Processing (ESSV)*, Vol. 81. 271–278.

[7] Anurag Kumar, Rajesh M Hegde, Rita Singh, and Bhiksha Raj. 2013. Event detection in short duration audio using gaussian mixture model and random forest classifier. In *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European*. IEEE, 1–5.

[8] Anurag Kumar and Bhiksha Raj. 2017. Deep cnn framework for audio event recognition using weakly labeled web data. *arXiv preprint arXiv:1707.02530* (2017).

[9] Yizhar Lavner, Rami Cohen, Dima Ruinskiy, and Hans IJzerman. 2016. Baby cry detection in domestic environment using deep learning. In *Science of Electrical Engineering (ICSEE), IEEE International Conference on the*. IEEE, 1–5.

[10] Sungbok Lee, Serdar Yildirim, Abe Kazemzadeh, and Shrikanth Narayanan. 2005. An articulatory study of emotional speech production. In *European Conference on Speech Communication and Technology*.

[11] Erik Marchi, Giacomo Ferroni, Florian Eyben, Leonardo Gabrielli, Stefano Squartini, and Bjorn Schuller. 2014. Multi-resolution linear prediction based features for audio onset detection with bidirectional LSTM neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2164–2168.

[12] MATRIX. 2019. MATRIX Creator. https://www.matrix.one/products/creator. (2019). [Online; accessed 28-Jan-2019].

[13] Ian McLoughlin, Haomin Zhang, Zhipeng Xie, Yan Song, and Wei Xiao. 2015. Robust sound event classification using deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23, 3 (2015), 540–552.

[14] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. 2016. TUT database for acoustic scene classification and sound event detection. In *Signal Processing Conference (EUSIPCO), 2016 24th European*. IEEE, 1128–1132.

[15] Veronica Morfi and Dan Stowell. 2018. Data-efficient weakly supervised learning for low-resource audio event detection using deep learning. *arXiv preprint arXiv:1807.06972* (2018).

[16] Mahesh Kumar Nandwana and Taufiq Hasan. 2016. Towards Smart-Cars That Can Listen: Abnormal Acoustic Event Detection on the Road.. In *INTERSPEECH*. 2968–2971.

[17] Stavros Ntalampiras, Ilyas Potamitis, and Nikos Fakotakis. 2009. An adaptive framework for acoustic monitoring of potential hazards. *EURASIP Journal on Audio, Speech, and Music Processing* 2009, 1 (2009), 594103.

[18] Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, Tuomas Virtanen, and others. 2017. Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25, 6 (2017), 1291–1303.

[19] Giambattista Parascandolo, Heikki Huttunen, and Tuomas Virtanen. 2016. Recurrent neural networks for polyphonic sound event detection in real life recordings. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 6440–6444.

[20] Karol J Piczak. 2015a. ESC: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 1015–1018.

[21] Karol J Piczak. 2015b. ESC: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 1015–1018.

[22] Asma Rabaoui, Manuel Davy, Stéphane Rossignol, and Noureddine Ellouze. 2008. Using one-class SVMs and wavelets for audio surveillance. *IEEE Transactions on information forensics and security* 3, 4 (2008).

[23] Shourabh Rawat, Peter F Schulam, Susanne Burger, Duo Ding, Yipei Wang, and Florian Metze. 2013. Robust audio-codebooks for large-scale event detection in consumer videos. (2013).

[24] J-L Rouas, Jérôme Louradour, and Sébastien Ambellouis. 2006. Audio events detection in public transport vehicle. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*. IEEE, 733–738.

[25] Roneel V Sharan and Tom J Moir. 2014. Comparison of multiclass SVM classification techniques in an audio surveillance application under mismatched conditions. In *Digital Signal Processing (DSP), 2014 19th International Conference on*. IEEE, 83–88.

[26] Naoya Takahashi, Michael Gygli, Beat Pfister, and Luc Van Gool. 2016. Deep convolutional neural networks and data augmentation for acoustic event detection. *arXiv preprint arXiv:1604.07160* (2016).

[27] Peter Transfeld, Simon Receveur, and Tim Fingscheidt. 2015. An acoustic event detection framework and evaluation metric for surveillance in cars. In *Sixteenth Annual Conference of the International Speech Communication Association*.

[28] Michel Vacher, Dan Istrate, Laurent Besacier, Jean-François Serignat, and Eric Castelli. 2004. Sound detection and classification for medical telesurvey. In *2nd Conference on Biomedical Engineering*. 395–398.

[29] Yun Wang, Leonardo Neves, and Florian Metze. 2016. Audio-based multimedia event detection using deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2742–2746.

[30] Show-Jane Yen and Yue-Shi Lee. 2009. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications* 36, 3 (2009), 5718–5727.