A REAL-TIME ENVIRONMENTAL SOUND RECOGNITION SYSTEM FOR THE ANDROID OS

Angelos Pillos [†], *Khalid Alghamidi* [†], *Noura Alzamel* [†], *Veselin Pavlov* [†], *Swetha Machanavajhala* [‡]

[†] Computer Science Department, University College London, UK
angelos.pillos.15, khalid.alghamidi.15, noura.alzamel.15, veselin.pavlov.15@ucl.ac.uk
[‡] Microsoft, Redmond, USA, swmachan@microsoft.com

ABSTRACT

Sounds around us convey the context of daily life activities. There are 360 million individuals [1] worldwide who experience some form of deafness. For them, missing these contexts such as fire alarm can not only be inconvenient but also life threatening. In this paper, we explore a combination of different audio feature extraction algorithms that would aid in increasing the accuracy of identifying environmental sounds and also reduce power consumption. We also design a simple approach that alleviates some of the privacy concerns, and evaluate the implemented real-time environmental sound recognition system on Android mobile devices. Our solution works in embedded mode where sound processing and recognition are performed directly on a mobile device in a way that conserves battery power. Sound signals were detected using standard deviation of normalized power sequences. Multiple feature extraction techniques like zero crossing rate, Mel-frequency cepstral coefficient (MFCC), spectral flatness, and spectral centroid were applied on the raw sound signal. Multi-layer perceptron classifier was used to identify the sound. Experimental results show improvement over state-of-the-art.

***Index Terms*—** Environmental sound recognition, signal processing, machine learning, Android OS

1. INTRODUCTION

Understanding or recognizing context of sounds in environmental surroundings is very important in terms of making the next move based on a sound that occurred. Examples include evacuating a building on hearing a fire alarm or attending to a baby on hearing the cries. Such activities depend on human hearing which intelligently filters out sounds and quickly recognizes it thereby signaling the brain to take the next step. According to the World Health Organization [1], over 360 million individuals worldwide suffer from some form of disabling hearing loss. Deaf individuals can neither hear sounds nor distinguish between many sounds. Such situations could be life threatening at times and also inconvenient.

Providing access to sound in some other form of communication are in demand for its practical applications towards assisting deaf individuals in their daily activities. Capturing, detecting, identifying sounds and alerting users in the form of visual notifications, and vibrations is a concept known as Environmental sound recognition. Although auditory recognition is an active research area where extensive efforts have been made over fifty years, the focus has largely been on recognizing human speech or music, and much less efforts have been directed towards acoustic event recognition [2, 3]. Over the past decade, environmental sound recognition has become popular leading to a considerable amount of research.

Technology has progressed to a stage where a powerful computational device can be easily carried in your pocket. Therefore, one of the emerging trends is the increased demand for sound recognition applications to be available on these portable devices. Despite the existence of non-portable applications that are capable of acoustic environmental recognition, it decreases the practical application of being able to move anywhere thereby hindering the freedom of deaf individuals. Additionally, reproduction of non-portable systems is considered either not suitable or leads to low performance on mobile devices [4, 5].

In this paper, we carry out investigations, experiments and propose a simple but effective approach to recognizing everyday environmental sounds using mobile devices, in particular on the Android platform. Our goal is to notify the user about an acoustic event that just occurred in the users surroundings.

Our main contribution is to deliver a real-time sound recognition system on Android devices following several principles. First, the fundamental nature of our system is a sound processing system and its key performance metric is the relative accuracy of correctly recognized sounds. With systems which provide only conceptually similar features [6], we managed to get similar and even a higher level of accuracy. Second, we designed a battery friendly approach and as a result of various tests, our application managed to be approximately two times more battery efficient than other similar applications. Third, it was proved that mobile devices were hardware efficient by optimizing the computational resources. Fourth, our sound recognition system is network independent indicating that it will be always available. Fifth, the machine learning algorithm that we used for the application has the capability to learn new types of sounds and update the model on Android platform. Last, we provide a simple design that alleviates privacy concerns over audio.

The rest of this paper is organized as follows. Section 2 motivates the problem of classifying environmental sounds. Section 3 provides a system overview. Section 4 discusses the sound datasets used in experiments. Section 5 describes our approach in designing the sound recognition system and explains in detail about the system functionality such as sound detection, feature extraction and classification techniques. Section 6 provides an evaluation of approaches used in related work and our approach. Section 7 concludes the research and proposes future work.

2. ACOUSTIC EVENT RECOGNITION OVERVIEW

Natural sounds (dog barking, rain, rooster, sea waves) and artificial sounds (helicopter, siren etc.,) that might be heard in a given environment [2] are often referred to as acoustic events. Unlike music signals, the characteristics of environmental sounds do not exhibit meaningful stationary patterns such as melody and rhythm [7]. Moreover, acoustic events differ from human speech because there is no existing sub-word dictionary for sounds in the same way that is possible to decompose words into their constituent phonemes [2]. Usually, environmental sound recognition systems have three main components: acoustic events detection as they occur, processing of these events in real-time by extracting useful information to create an acoustic feature for classification and then the determination of the most suitable category for that event, based on training carried out with similar samples [2].

3. SYSTEM OVERVIEW

Different mobile recognition system design choices were explored in prior related works [4, 8]. For example, feature extraction and sound recognition could be done directly on the phone and this mode is called embedded sound recognition. Another architecture is where both the feature extraction and recognition are done in cloud which is synoymous with a back-end server. [8]. In addition, we can have a combination of both approaches, distributed sound recognition, where the recognition is split across the mobile device and the server.

In our case, due to the high requirement of system availability we have decided to use the embedded mobile sound recognition architecture where both feature extraction and recognition are implemented on the mobile device alone. The main advantage of this mode is that the application will work in conditions where there is no Wi-Fi connection [4]. Mobile internet will not be sufficient since we are using 44100 Hz and 16 bits per sample. Mathematically, it is equivalent to 705600 bits per second which averages to 5 MB per minute and that requires a lot of internet bandwidth.

Our system consists of two main deliverables: A desktop application that considers a set of environmental sounds [10] as training data and extracts features, then trains the model to classify sounds based on the features and an Android application component that loads the machine learning model generated by the desktop application. The goal is to provide a single application that identifies multiple sounds such that the user can use the application like a baby monitor or a fire alarm alerter. To achieve this, the android application first detects the sound activity in the environment and captures the sound using a microphone, then extracts the features from the captured sound. Finally, it classifies the sound into its correspondent class using Multi-Layer Perceptron classifier.

4. SOUND DATASET

The dataset used in this paper is the ESC-10 [10] which represents three general groups of sounds. It includes transient/percussive sounds, sometimes with very meaningful temporal patterns (sneezing, dog barking, clock ticking) and sound events with strong harmonic content (crying baby, crowing rooster). Additionally, it also contains more or less structured noise/soundscapes (rain, sea waves, fire crackling, helicopter, chainsaw). This dataset was available under a Creative Commons non-commercial license through the Harvard Dataverse project [9]. Furthermore, the recordings are available in a unified format (5- second-long clips, sampled at 44.1kHz, 16 bits resolution, single channel) [10].

5. SYSTEM APPROACH

Most sound recognition systems follow either the Simple Classifier approach, Gaussian Mixture Model approach or HMM Model approach [11]. In our system we followed the approach with the lowest energy consumption and relatively high accuracy where the system detection component will continuously record the environmental sounds in chunks of one second each. Then, each captured sound signal will be processed by splitting it into frames (windows) of 1024 samples (around 25 millisecond (ms) considering the 44.1kHz sampling rate). In the case of training, splitting is done with an overlap of 50%, whereas in the testing there is no overlap in the splitting process. Each frame is smoothed with a Hanning window which is used with gated continuous signals and long transients to give them a slow onset and cutoff in order to reduce the generation of side lobes in their frequency spectrum.

Next, each frame is passed to the MFCC, ZCR, Spectral Centroid and Spectral Flatness feature extractors. Then, statistical values such as mean and standard deviation of the extracted features are calculated. The feature extraction stage plays an important part for the recognition process (explained in Subsection 5.2). At the end of this stage, there is a constant size feature vector for the whole one second sound signal. By analyzing each frame individually, the spectrogram type B solution technique is applied for selecting the features set [11]. We have considered this solution since from an implementation point of view, this technique does not need much memory [11]. Finally, the feature vector will be passed to the classification model to get the best match. The described recognition approach is shown in Fig. 1.

5.1. Sound Detection

The detection technique implemented in the system is based on the standard deviation of normalized power sequences. This method immediately signals a possible impulsive sound [11]. The signal is based on the normalization of successive windowed power sequences where the detection flag is signaled when the power exceeds the threshold value, which in our case is 0.015 [11] This technique aims to be simple yet an efficient detection scheme with a low computational load since it intends to be running all the time. Moreover, the performance turns out to be relatively the same as other more energy consuming advanced methods as described in [11].

5.2. Feature Extraction

As most recent research propose, feature analysis is considered the most crucial and important part in building a robust and effective recognition system [11, 12]. The aim of feature extraction is to convert the sound signal into a sequence of feature vectors in order to produce a set of characteristic features that describe the sound signal [13, 14].

From a physical point of view, signals can be represented in different domains: time or frequency domain. Therefore, acoustic features can be grouped into two groups, temporal and spectral features that can be extracted using time and frequency domain respectively [12]. Due to the nature of the environmental sound, which is considered as unstructured data and no assumptions can be made about detectable repetitions or harmonic structure in the signal, many features are needed to describe the audio signals [12].

In this work, several analysis methods were considered, inspired from the speech recognition community. As mentioned in

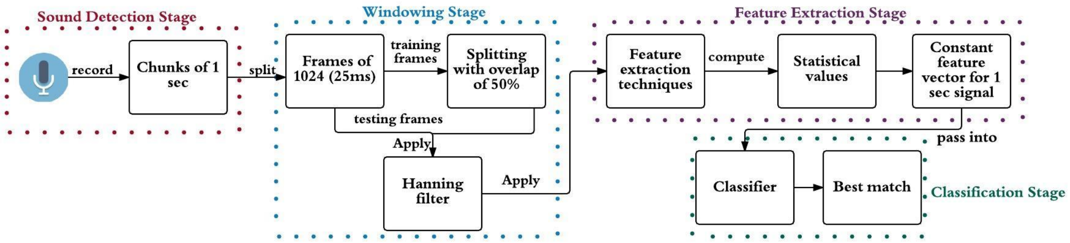

Figure 1: System Architecture.

Section 5, the feature extraction is applied on frame level. Time domain features such as Zero Crossing Rate, Spectral Centroid and Flatness were considered. Zero crossing rate (ZCR) is a measure of the number of times the signal value crosses the zero axe [14]. It is considered a very simple, yet useful feature. ZCR is defined formally as illustrated in (1)

$$ZCR = \frac{1}{T-1}\sum_{t=1}^{T-1}\prod S_t S_{t-1} < 0 \tag{1}$$

Where s is a signal of length T and the indicator function PA is 1 if its argument A is true and 0 otherwise. After the ZCR is applied for each frame of the 1 second signal, the mean and standard deviation were computed for all of the ZCR coefficients.

In order to obtain frequency domain features, we first applied the Fast Fourier Transform (FFT) algorithm to convert the sound signal from its original time domain into magnitude spectrum. Three types of features were extracted from each frame: Mel-frequency cepstral coefficient (MFCC), spectral centroid, and spectral flatness.

MFCC is the most common feature used in audio classification [15, 10, 11, 8]. The idea of the MFCC technique is to distribute the cepstral coefficients according to the critical bands, instead of the traditional linear distribution. Usually, it is mentioned as calculation of 13 Mel-frequency cepstral coefficients and discarding 0th order coefficient for each of the 25 ms frames. After that, the mean and standard deviation of each frame element are computed which results in 12 values representing means and 12 values representing standard deviation.

Another widely used feature is spectral centroid, which measures the brightness of a sound. Brightness is especially relevant for continuous instrumental sounds and is calculated as the amplitude-weighted average of all partials for the whole duration of the event [16]. In spectral centroid, the higher the centroid, the brighter the sound is [12]. It is calculated as the weighted mean of the frequencies present in the signal, determined using a Fourier transform, with their magnitudes as weights [14].

$$Centroid = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)} \tag{2}$$

where $x(n)$ represents the weighted frequency value, or magnitude, of bin number n, and $f(n)$ represents the center frequency of that bin. Similarly, spectral flatness has been useful in audio signal processing which quantifies the tonal quality; namely, how much tone-like the sound is as opposed to being noise-like [12]. It is calculated by dividing the geometric mean of the power spectrum by the arithmetic mean of the power spectrum [14]:

$$Flatness = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{\sum_{n=0}^{N-1} x(n)}{N}} = \frac{\exp(\frac{1}{N}\sum_{n=0}^{N-1}\ln x(n))}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)} \tag{3}$$

where x(n) represents the magnitude of bin number n. Note that a single (or more) empty bin yields a flatness of 0, so this measure is most useful when bins are generally not empty.

As in MFCC, mean and standard deviation were calculated for spectral centroid. Moreover, min and max were calculated for spectral flatness. In total we obtained a vector of features with 30 elements which describe the signal to help the classifier provide the best match.

5.3. Classification

For the classification stage, we worked with the Weka Classification Library [17]. It is a collection of machine learning algorithms for data mining tasks which provides a number of classification models. We used an unofficial stripped down version of the library [18] which was compatible on the Android platform. The author of the project claims that this is the same Weka project with the GUI components removed so that it can work on Android.

As mentioned previously, a desktop application has been developed to extract the features from the datasets mentioned in Section 4 and then saves the training and test data into separate ARFF files (Attribute-Relation File Format) for training the model and then evaluating it. Another option is to save the extracted features from both datasets and then save them into one file for cross validation.

The extracted features were input to five classifiers: Multi-layer Perceptron, SVM (SMO in Weka), RandomForest, BayesNet and NaiveBayes. Weka uses Sequential Minimal Optimization (SMO) to solve the SVM training problem by using heuristics to partition the training problem into smaller problems that can be solved analytically. Training and testing were performed on ESC-10 dataset, using 2, 3, 4 and 5 fold cross validation. The results for the 10 class datasets are shown in figure 2.

5.4. Design to Alleviate Privacy Concerns

Gray [19] gives a nice explanation on privacy implications of having always-on microphone enabled devices. We utilized some of the design approaches in our application to alleviate privacy concerns. For

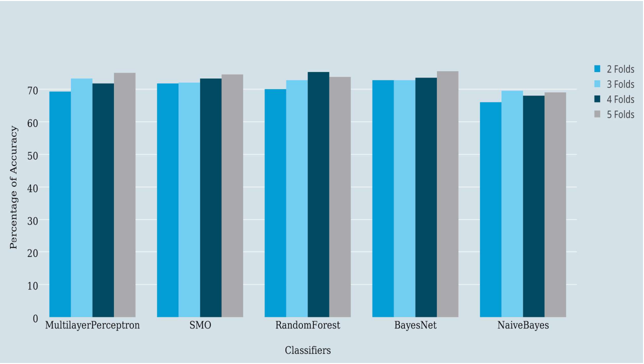

Figure 2: Cross Validation of 10 class dataset.

example, the user would know when the application is listening to a sound, when it has detected a sound and finally when it's classifying the sound; all of which are provided by flashing lights. Additionally, we also give the user an option to turn the sound recognition on or off. This makes it clear that user has given consent for the application to be always-on listening.

6. EVALUATION

We evaluate our proposed system's accuracy by comparing the results obtained using a combination of feature extraction algorithms and Multi-Layer perceptron classifier with the results obtained from previous studies [10]. The same dataset [10] was used in the evaluation.

Additionally, one of the main factors of a successful application is the battery consumption. We evaluate the battery efficiency of our proposed application with another similar application.

6.1. Multi-layer Perceptron Accuracy

Fig. 2 lists the accuracy of each classifier on 10 classes dataset. As a result of training the models, Multi-layer Perceptron was chosen as it performed with the best overall accuracy with 74.5% for the 10 class dataset and also due to the ability to derive meaningful patterns from unstructured data.

We perform multiple versions of cross validation to ensure comparability with ESC paper results [10]. Though a large number of folds is more expensive, it does however give a less biased estimate of the model performance. Additionally, the variance of the resulting estimate for different samples or partitions of data to form training and test sets is reduced as the number of folds are increased. As a result, cross validating the classifiers with more folds provide more accurate results. Table (1) shows an example of this.

6.2. Accuracy Comparision

The results of the proposed system were compared to results in the research paper considering environmental sound recognition [9] using Random Forest and Multi-layer perceptron classifiers. The results were acquired by using 5-fold cross validation. For the purpose of objective results the same dataset was used: ESC-10 dataset [10]. From Table (2), we can see that our system performs with a better accuracy using Multi-Layer Perceptron classifier.

Class-Fold	10 Classes
2 folds	69.25%
3 folds	73.25%
4 folds	71.75%
5 folds	74.5%

Table 1: Multi-layer Perceptron Cross Validation Results

	ESC-10	
	Random forest	Multi-Layer Perceptron
Proposed system	73.75%	74.50%
ESC paper results [10]	73.70%	62.50%

Table 2: Comparison with previous research studies

6.3. Battery Consumption

We have compared battery consumption of our system with Otosense [6] which is also an environmental sound recognition system. Battery consumption was tested by measuring the mobile device runtime for 5 repetitions. The tests were held by using two smartphones: LG Nexus 5 and LG Nexus 4. During all of the tests the mobile devices were initially fully charged, the connection to internet was switched off and no background processes were stopped. The runtime was measured until the mobile devices switched off. Table (3) shows the average runtime for the both systems using the mobile devices stated above. Both of the mobile devices perform almost twice better when running our proposed approach.

7. CONCLUSION

In this paper, a viable real-time environmental sound recognition system for Android mobile devices was developed. The system uses a sound detection algorithm which helps reduce power consumption. In addition, a combination of several feature extracting algorithms were applied to accurately identify sounds. Finally, several classifiers were compared with described features and Multi-Layer Perceptron classifer performed best with 74.5% accuracy on 10-class dataset. The recognition accuracy of the proposed system exceeds the results of previous efforts using the same dataset [10].

The current dataset was chosen specifically for benchmarking purposes with prior papers. Looking into the future we propose to filter out sounds, just as fire crackling, clock-tickling, which are not really useful for people who are deaf or hard of hearing.

	LG Nexus 5	LG Nexus 4
Proposed System	5 hours 27 minutes	7 hours 15 minutes
Otosense [6]	3 hours 4 minutes	3 hours 50 minutes

Table 3: Average runtime duration

8. REFERENCES

[1] "World Health Organization", http://www.who.int/en/ [accessed on: 30 July 2016].

[2] J. Dennis, "Sound Event Recognition And Classification In Unstructured Environments", PhD thesis, Nanyang Technological University, 2011.

[3] R. Goldhor, "Recognition of environmental sounds", in *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference*, vol. 1, pp. 149–152, 1993.

[4] A. Kumar, A. Tewari, S. Horrigan, M. Kam, F. Metze and J. Canny, "Rethinking speech recognition on mobile devices", in *Proceedings of 2nd International Workshop on Intelligent User Interfaces for Developing Regions*, Palo Alto, CA, pp. 10-15, February 2011.

[5] A. Schmitt, D. Zaykovskiy and W. Minker, "Speech recognition for mobile devices", in *International Journal of Speech Technology 11*, no. 2, pp. 6372, 2008.

[6] "Otosense", http://www.otosense.com/ [accessed on: 30 July 2016].

[7] N. Scaringella, G. Zoia and D. Mlynek, "Automatic genre classification of music content: a survey", in *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 133141, March 2006. doi: 10.1109/MSP.2006.1598089.

[8] R. Mattia, S. Feese, O. Amft, N. Braune, S. Martis and G. Troster, "AmbientSense: A realtime ambient sound recognition system for smartphones", in *Pervasive Computing and Communications Workshops (PERCOM Workshops)* 2013 IEEE International Conference on. IEEE, 2013.

[9] "The Dataverse project", http://dataverse.org/ [accessed on: 30 July 2016].

[10] K. Piczak,"ESC: Dataset for Environmental Sound Classification", in *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 10151018, ACM, 2015.

[11] A. Dufaux, "Detection And Recognition Of Impulsive Sounds Signals", PhD thesis, University of Neuchtel, 2001.

[12] S. Chu, S. Narayanan and C. Kuo, "Environmental Sound Recognition With TimeFrequency Audio Features", in *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 11421158, Aug. 2009. doi: 10.1109/TASL.2009.2017438.

[13] M. Cowling and R. Sitte, "Comparison of techniques for environmental sound recognition", in *Pattern recognition letters*, vol. 24, no.15, pp. 2895-2907, 2003.

[14] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project", Technical report, IRCAM, 2004.

[15] X. Zhang and Y. Li, "Environmental Sound Recognition Using DoubleLevel Energy Detection", in *Journal of Signal and Information Processing*, Vol. 4 No. 3B, pp. 1924, 2013. doi: 10.4236/jsip.2013.43B004.

[16] D. Keller and J. Berger, "Everyday sounds: synthesis parameters and perceptual correlates", in *Proceedings of the VIII Brazilian Symposium of Computer Music*, 2001.

[17] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java", http://www.cs.waikato.ac.nz/ml/weka/ [accessed on: 30 July 2016].

[18] "Weka for Android", https://github.com/rjmarsan/WekaforAndroid [accessed on: 30 July 2016].

[19] S. Gray, "Always On: Privacy Implications of Microphone-Enabled Devices", in *Future of privacy forum*, April 2016.