A Sound Events Detection and Localization System based on YAMNet Model and BLE Beacons

Carlos M. Mesa-Cantillo[†], Itziar Alonso-González[†‡], Miguel A. Quintana-Suárez[‡], Carlos Ley-Bosch[†‡], Carlos Ramírez-Casañas[†‡], Javier J. Sánchez-Medina[*], David Sánchez-Rodríguez[†‡]

[†]Institute for Technological Development and Innovation in Communications, University of Las Palmas de Gran Canaria, Las Palmas de Gran Canaria, Spain, email: {carlos.mesa, itziar.alonso, carlos.ley, carlos.ramirez, david.sanchez}@ulpgc.es

[‡]Telematic Engineering Department, University of Las Palmas de Gran Canaria, Las Palmas de Gran Canaria, Spain, email: mangel.quintana@ulpgc.es

[*]CICEI, Innovation Center for the Information Society, University of Las Palmas de Gran Canaria, Las Palmas de Gran Canaria, Spain, email: javier.sanchez@ulpgc.es

Abstract—**Automatic sound event detection is a must-have feature for several emerging applications, such as surveillance, automatic listening, and noise source identification. Acoustic Event Detection (AED) aims to know the sounds' identity and temporal position in signals captured by one or several microphones. In this work, we use a pre-trained Yet Another Mobile Network (YAMNet) model to perform real-time audio classification. That audio event classifier model takes the audio waveform as an input and makes independent predictions for each of the 521 audio events in the AudioSet ontology. The model used the MobileNet v1 architecture and was trained using the AudioSet corpus. By means of a Raspberry Pi 3, a commercial microphone, and a set of Bluetooth Low Energy (BLE) beacons, this system is able to detect potentially harmful events. Thus, the system can detect where and what event has been detected and send it to a database. After that, the database is updated, and a notification can be sent to the users of a specific application. This information may be helpful for people with disabilities so they can be warned of danger in the nearby areas.**

Index Terms—**sound event classification, beacons, machine learning, yamnet.**

I. Introduction

The task of detecting the onsets and offsets of target class activities in general audio signals is known as Sound Event Detection (SED) [1]. It is a technique that accepts an audio signal as input and produces temporal activity for target classes like ”vehicle passing by,” ”footsteps,” ”people chatting,” ”gunshot,” etc. The time resolution of class activities might vary between techniques and datasets.

The use of audio sensors in surveillance and monitoring applications is especially useful for event detection. Detection systems can effectively notify when an event occurs while enabling further processing, such as notifying the system with information about the event and its position. In recent years, Acoustic Event Detection (AED) [2], [3] and Acoustic Event Classification (AEC) have been essential for many applications such as security surveillance [4], human-computer interaction [5], and ”machine hearing” [6].

In most surveillance systems, locating the position of the acoustic source over a topological grid is the final objective of sound localization. In environments with little reverberation time, like a typical public square, the Time Difference of Arrivals (TDOA) of the signal at an array of microphones is the most used technique for source localization. These time delays are further analyzed in [7] to determine the source location.

On the other hand, indoor positioning technology is being commercialized in different technologies and qualities. Unlike the Global Positioning System (GPS), which is used for outdoor positioning, there is no proven method that can be used for all purposes in indoor positioning, at the moment of writing this paper. The available technologies employed nowadays may vary in terms of cost, accuracy, and maintenance requirements. Some of the most common may be the following:

- Bluetooth Low Energy (BLE): Signals from battery-powered beacons are the core of indoor location technology. It is one of the most remarkable technologies that have emerged for indoor location. It uses BLE beacons or iBeacons, which are cheap, small, have long battery life, and do not require an external power source. A receptor device can detect the beacon signal and can roughly calculate the distance to the beacon.
- Wi-Fi: it can be used similarly to BLE beacons but requires an external power source and, higher setup and hardware costs. The signal tends to be stronger than BLE, with a larger range.
- Ultra-Wideband: it is the most precise method for indoor positioning available. Nevertheless, compared to its alternatives, it has more hardware requirements, as well as higher costs.

In this paper, a system for sound event detection and localization is proposed. It is based on the Yet Another Mobile Network (YAMNet) model for sound detection and BLE beacons to infer the location. YAMNet is based on the Visual Geometry Group (VGG) architecture and employs the Mobilenet v1 depthwise-separable convolution architecture. The classifier has 3.7M weights and performs 69.2M multiplies for each 960 ms input frame. Although the YAMNet model has a total of 521 classes, only the ones that may be relevant for notification will be used. In [8], the accuracy of this model is tested with only 6 out of the 527 classes of the Audioset database due to the simplification of the experiment, as well as time and computational constraints. YAMNet can classify single fixed-size audio samples with 92.7% accuracy and 68.75% precision.

In addition to the use of microphones for the SED, in this system, BLE beacons are used to associate each microphone with a specific location. This location is previously set in the database to correspond to the Media Access Control (MAC) of the beacon, so when the Raspberry Pi code scans for beacons, the beacons are filtered and only those beacons that have been configured in the database with a location associated with them will appear. This information associated with an application that can notify and warn about the environment can be helpful for people with disabilities. For example, in the case of someone who is deaf, a notification will appear on their phone screen and warn them that an event has occurred nearby and that they should be alert. This could help them avoid any possible danger. Also, since cell phones can read notifications, if the person is blind, an audio notification about a dangerous event that has occurred in a nearby area may be helpful for them.

This article is organized as follows. Section II summarizes the related work about sound event detection and indoor positioning systems. Section III briefly describes the hardware and software employed. Section IV introduces the methodology to start the real-time audio classification. Next, in Section V, we analyse the results of the working system. Finally, in Section VI, the conclusion and future work are presented.

II. Related Work

Audio tagging is the task of detecting the presence or absence of a certain sound event in a recording. This has several applications, such as surveillance, monitoring, and health care [9].

In many papers such as [10], [11], the input is a polyphonic sound in an undefined context, and the output is one identified sound event at each instance of the polyphonic sound. The sound event is determined by the most prominent one in that instance.

Polyphonic events are the main error source of AED. This problem is usually solved by treating the AED task as a multi-label classification problem. In [12], to better handle polyphonic mixtures, the authors propose to consider each possible label combination as one class. In other works, to handle event overlaps, the AED task is usually framed as a multi-label classification problem. This is solved by a deep neuronal network with multi-label output [13], [14].

In [15], a solution for the polyphonic SED task on mobile devices is presented. Its architecture includes offline training and online detection. The offline training involves model training and compression, and the online detection process consists of acquiring sensor data, audio processing, and detecting sound events.

Indoor positioning systems already have broad applications for providing localized information and directions. In [16], an indoor positioning system with room-level accuracy is proposed. It focuses on an installation procedure that non-technical staff can easily follow. In addition, it has a low cost.

Most of the existing solutions, employing beacons and smartphones, require the floorplan of the indoor environment and many beacons. These applications usually try to increase the accuracy of the coordinate estimation on Line Of Sight (LOS) environments using a large number of beacons [17] and large training datasets.

The most common methods used in indoor positioning systems are:

- Trilateration: the distance from the source to the receiver is used to estimate the user's location. A more detailed description of this method is presented in [18].
- Triangulation: a method for calculating a position that relies on a known distance between two measuring apparatuses and the measured angles from those two points to an object [19].
- Fingerprinting: this method consists of two stages. First, Received Signal Strength (RSS) measurements are captured for multiple points in the indoor environment. Then, these measurements are used to determine the user's location.

III. Materials

The hardware and software used to operate this system are as follows:

A. Raspberry Pi 3

A computer is required to run the classification script and send the information to the database. Raspberry Pi was selected because it is an inexpensive and compact computer that meets the software requirements for classification. This computer runs Raspberry Pi OS as its operating system and requires an Internet connection to communicate with the database. Also, Python 3.9 needs to be installed to execute the scripts with the required libraries.

B. iBKS 105

iBKS 105 is the BLE beacon that was employed. Its specifications are as described in Table I. These beacons are configured to work with the iBeacon protocol, with a transmission (Tx) Power of 0 dBm and an advertising interval of 1 second.

C. Krom Kimu Pro

The microphone used to capture the audio in real time was a commercial microphone, in this case, a Krom Kimu Pro. The specifications are as shown in Table II. This microphone was selected since it is a commercial microphone whose sample rate and the number of channels can be configured in the PyAudio library to correspond to the audio input of the YAMNet model.

D. Software

The code for the real-time classification was done in Python 3.9. The most relevant libraries that were employed are the following ones:

TABLE I
IBKS 105 SPECIFICATIONS

Size	11,3 x Ø52,6 mm
Battery lifetime	30-40 months (Tx power at 1s interval)
Protocols	iBeacon and Eddystone

TABLE II
KROM KIMU PRO SPECIFICATIONS

Sensitivity	$-35dB \pm 3dB$
Impedance	2.2K ohms
Frequency Response	20 Hz - 20 KHz
Sample Rate	48 KHz at 16 bits

1) PyAudio: To collect the real-time audio, the library PyAudio was selected since it allows configuring the sampling rate, the number of channels, the number of frames per buffer, and which microphone will capture the audio. The audio input must be configured as 16 kHz mono audio. This setting is required because it is the one supported by the YAMNet model.

2) TensorFlow Lite: TensorFlow Lite library was employed since the YAMNet model is a pre-trained model given by them. This model was trained with audio features computed as follows:

- All audio is resampled to 16 kHz mono.
- A spectrogram is computed using the Short-Time Fourier Transform magnitudes with a window size of 25 ms, a window hop of 10 ms, and a periodic Hann window.
- A mel spectrogram is computed by mapping the spectrogram to 64 mel bins covering the range 125-7500 Hz.
- A stabilized log mel spectrogram is computed by applying log(mel-spectrum + 0.001), where the offset is used to avoid taking a logarithm of zero.
- These features are then framed into 50%-overlapping examples of 0.96 seconds, where each example covers 64 mel bands and 96 frames of 10 ms each.

Additionally, since the computer that will support the real-time classification is a Raspberry Pi 3, and it cannot support some of the libraries that can be used on a desktop computer, it was necessary to use other libraries which require less processing power. In this case, the library employed was TensorFlow lite. The TensorFlow lite version of the YAMNet model has some additional changes:

- The model is quantized. The model is retrained with Relu6 non-linearities instead of Relu to limit the activation ranges.
- The inference signature is simpler. It takes a fixed-length audio frame and returns a single vector of scores for 521 audio event classes.

3) Bluepy: This is a project to provide an API to allow access to Bluetooth Low Energy devices from Python. At the moment, it runs only on Linux. This library has been used to scan the BLE beacons near our Raspberry Pi to speed up the assignment of the corresponding zone.

IV. METHODOLOGY

The methodology followed in this research work is presented in Fig. 1, which development is described as follows:

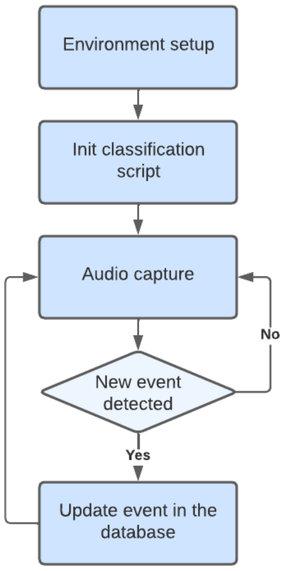
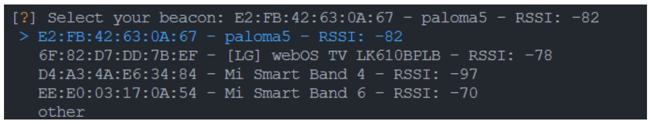


Fig. 1. The proposed methodology.

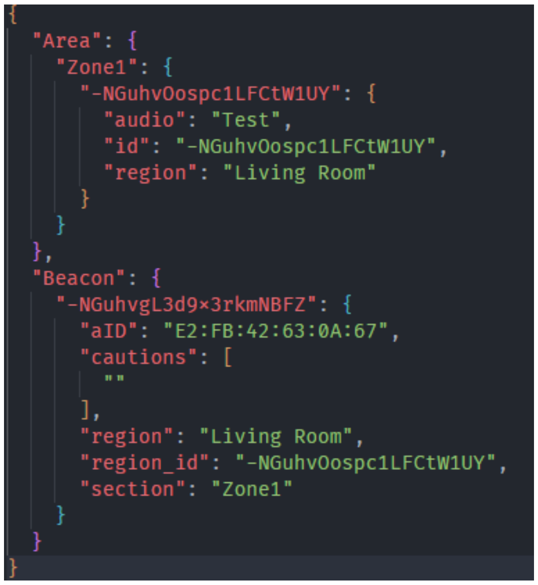
First, a previous environment setup is required. The beacons must be registered in the database and associated with a known location. This is made by designating each MAC of the beacons that will be employed to a known location of the indoor environment. Also, the iBKS 105 beacons need to be configured with their own application. In this application, the BLE service that will be used can be chosen between, Eddystone or iBeacon. The beacons were configured as iBeacon with a 1000 ms advertising interval and radio Tx Power of 0 dBm.

To register the beacons to the database, for 10 seconds a script developed in python scans for the nearest beacons. Then, it displays a list of the MAC addresses that were discovered with the name and the Received Signal Strength Indicator (RSSI) of the device as shown in Fig. 2. After that, the user can choose the desired MAC and select the region and the section where the beacon is displayed. The database structure will look as in Fig. 3. The audio is associated with an area or section instead of the beacon because multiple beacons or microphones can be associated with the same area. After the beacons are in their determined location, the Raspberry Pi and microphone pair can be placed near the desired beacon.



```
[?] Select your beacon: E2:FB:42:63:0A:67 - paloma5 - RSSI: -82
 > E2:FB:42:63:0A:67 - paloma5 - RSSI: -82
   6F:82:D7:DD:7B:EF - [LG] webOS TV LK610BPLB - RSSI: -78
   D4:A3:4A:E6:34:84 - Mi Smart Band 4 - RSSI: -97
   EE:E0:03:17:0A:54 - Mi Smart Band 6 - RSSI: -70
   other
```

Fig. 2. Adding beacon to the database.



```
{
  "Area": {
    "Zone1": {
      "-NGuhvOospc1LFCtW1UY": {
        "audio": "Test",
        "id": "-NGuhvOospc1LFCtW1UY",
        "region": "Living Room"
      }
    }
  },
  "Beacon": {
    "-NGuhvgL3d9×3rkmNBFZ": {
      "aID": "E2:FB:42:63:0A:67",
      "cautions": [
        ""
      ],
      "region": "Living Room",
      "region_id": "-NGuhvOospc1LFCtW1UY",
      "section": "Zone1"
    }
  }
}
```

Fig. 3. Database structure.

Once the Raspberry has been initialized, and the script runs, it will scan for nearby beacons. A list of the detected beacons that are registered in the database will be displayed, showing their MAC address, region, section, and RSSI. The user can choose from all the possible options for sound event detection to update the information in the database associated with that location. Next, the user can select the microphone to capture the audio in real-time. Once this configuration has been made, the script will start capturing audio. Whenever it detects a new event happening in relation to the previous one, it will update the database with the new event. The audio captured with the microphone must be 16 kHz mono audio to analyze sound events. This audio configuration is necessary for the YAMNet model to perform a correct audio classification. This adjustment was made when the microphone was selected.

V. Results

As mentioned in the previous section, the used components were configured to test the system and verify that it works correctly. First, several iBKS beacons were configured with the iBKS config tool application. Then, they were placed in different locations and registered in the database with the python script, so there were multiple options to set the microphone and Raspberry Pi.

Firebase was implemented as the back-end of the application, and the Firebase real-time database was used. This database is structured like a JavaScript Object Notation (JSON) file. In this database, the beacons were registered, as shown in Fig. 3.

Once the beacons were registered in the database, the script was run on the computer. Then one of the locations associated with the previously set beacons and the Krom Kimu Pro microphone was selected, after which the audio was captured.

YAMNet was re-trained in a transfer learning approach, for 5 out of the 50 categories in the ESC-50 dataset [20], to later test the accuracy of the model. Each of these categories consists of 40 sounds. The selected categories were the sounds of dogs, knocking on wooden doors, coughing, sirens, and vacuum cleaners. A total of 200 sounds were selected and divided into five different folds. Defining a very simple sequential model with one hidden layer and five outputs to recognize the sounds described before. For the new model, 60% of the samples were employed for testing, 20% were employed for validation, and the remaining samples were employed for testing. The model was configured to train for 20 epochs in the training phase. The accuracy obtained was around 87% in the 20 rounds that were done. In addition, a confusion matrix with 20 different sound samples was generated to visualize the results. The results obtained for these 20 samples are shown in Fig. 4.

To test the real-time update of the database, some sounds were emulated, such as knocking on a door, the sound of a siren, or dog sounds. These sounds were classified as intended, and the database was updated after a few seconds each time the sound changed. This update only rewrites the event of the previously selected location; if the event being played is the same, the database will not be updated.

VI. Conclusion and future work

In this work, it is described a system that is easy to implement and can classify real-time audio events and update a database with the detected location and associated information of such events. This system could be implemented in different environments to help people with disabilities. This assistance could take the form of an audible notification whenever a potentially dangerous event occurs near the user, requiring his or her attention.

In our ongoing work, we are planning to implement the system in a mobile phone application that detects where the user is located and warns them there might be a danger for a user with a disability. Also, the application could send push notifications so the user could be aware of the location

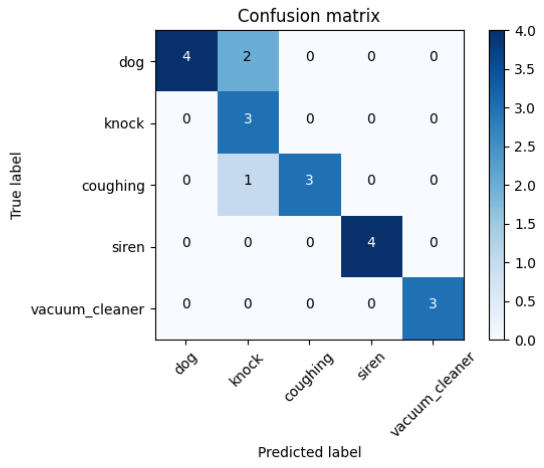


Fig. 4. Confusion matrix of the five classes for 20 different samples.

and the detected event. Additionally, with the implemented beacons, an indoor positioning system could be implemented in the application, enabling more accurate indications about the events that are being captured.

ACKNOWLEDGMENT

This research has been partially founded by the Consejería de Economía, Conocimiento y Empleo del Gobierno de Canarias, Agencia Canaria de Investigación, Innovación y Sociedad de la Información under projects ProID2020010009 and CEI2020-08, Spain.

REFERENCES

[1] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen and T. Virtanen, ”Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection,” in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no. 6, pp. 1291-1303, June 2017, doi: 10.1109/TASLP.2017.2690575.

[2] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange and M. D. Plumbley, ”Detection and Classification of Acoustic Scenes and Events,” in IEEE Transactions on Multimedia, vol. 17, no. 10, pp. 1733-1746, Oct. 2015, doi: 10.1109/TMM.2015.2428998.

[3] H. Phan, M. Maaß, R. Mazur and A. Mertins, ”Random Regression Forests for Acoustic Event Detection and Classification,” in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 23, no. 1, pp. 20-31, Jan. 2015, doi: 10.1109/TASLP.2014.2367814.

[4] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci and A. Sarti, ”Scream and gunshot detection and localization for audio-surveillance systems,” 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, 2007, pp. 21-26, doi: 10.1109/AVSS.2007.4425280.

[5] J. Maxime, X. Alameda-Pineda, L. Girin, and R. Horaud, ”Sound representation and classification benchmark for domestic robots,” 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 6285-6292, doi: 10.1109/ICRA.2014.6907786.

[6] N. Ma, T. May, H. Wierstorf and G. J. Brown, ”A machine-hearing system exploiting head movements for binaural sound localisation in reverberant conditions,” 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 2699-2703, doi: 10.1109/ICASSP.2015.7178461.

[7] J. Chen, Y. Huang, and J. Benesty, Audio Signal Processing for Next Generation Multimedia Communication Systems. Kluwer, 2004, ch. 4-5.

[8] I. Kuzminykh, D. Shevchuk, S. Shiaeles, and B. Ghita, ”Audio interval retrieval using convolutional neural networks.” Internet of Things, Smart Spaces, and Next Generation Networks and Systems, pp. 229–240., 2020

[9] Y. Zhang and M. Martínez-García, ”Machine Hearing for Industrial Fault Diagnosis,” 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), 2020, pp. 849-854, doi: 10.1109/CASE48305.2020.9216787.

[10] A. Mesaros, T. Heittola and A. Klapuri, ”Latent semantic analysis in sound event detection,” 2011 19th European Signal Processing Conference, 2011, pp. 1307-1311.

[11] T. Heittola, A. Mesaros, T. Virtanen, and M. Gabbouj, ”Supervised model training for overlapping sound events based on unsupervised source separation,” 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 8677-8681, doi: 10.1109/ICASSP.2013.6639360.

[12] H. Phan, T. N. T. Nguyen, P. Koch, and A. Mertins, ”Polyphonic Audio Event Detection: Multi-Label or Multi-Class Multi-Task Classification Problem?,” ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022, pp. 8877-8881, doi: 10.1109/ICASSP43922.2022.9746402.

[13] E. Çakir, G. Parascandolo, T. Heittola, H. Huttunen and T. Virtanen, ”Convolutional recurrent neural networks for polyphonic sound event detection”, IEEE/ACM Trans. on Audio Speech and Language Processing, vol. 5, no. 6, pp. 1291-1303, 2017

[14] S. Jung, J. Park, and S. Lee, ”Polyphonic sound event detection using convolutional bidirectional LSTM and synthetic databased transfer learning”, Proc. IEEE Int. Conf. on Acoust. Speech Signal Process, pp. 885-889, 2019.

[15] Y. Fu, K. Xu, H. Mi, H. Wang, D. Wang, and B. Zhu, ‘A Mobile Application for Sound Event Detection’, in Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, 7 2019, pp. 6515–6517.

[16] T. Tegou, I. Kalamaras, K. Votis and D. Tzovaras, ”A low-cost room-level indoor localization system with easy setup for medical applications,” 2018 11th IFIP Wireless and Mobile Networking Conference (WMNC), 2018, pp. 1-7, doi: 10.23919/WMNC.2018.8480912.

[17] D. Čabarkapa, I. Grujić and P. Pavlović, ”Comparative analysis of the Bluetooth low-energy indoor positioning systems”, Telecommunication in Modern Satellite Cable and Broadcasting Services (TELSIKS) 2015 12th International Conference on, pp. 76-79, 2015.

[18] A. N. Raghavan, H. Ananthapadmanaban, M. S. Sivamurugan, and B. Ravindran, ”Accurate mobile robot localization in indoor environments using Bluetooth”, Robotics and Automation (ICRA) 2010 IEEE International Conference on, pp. 4391-4396, 2010.

[19] P. K. Yoon, S. Zihajehzadeh, B.-S. Kang and E. J. Park, ”Adaptive Kalman filter for indoor localization using Bluetooth low energy and inertial measurement unit”, Engineering in Medicine and Biology Society (EMBC) 2015 37th Annual International Conference of the IEEE, pp. 825-828, 2015.

[20] K. J. Piczak, ‘ESC: Dataset for Environmental Sound Classification’, in Proceedings of the 23rd Annual ACM Conference on Multimedia, 2015, pp. 1015–1018.