EMBEDDED ACOUSTIC SCENE CLASSIFICATION FOR LOW POWER MICROCONTROLLER DEVICES

Filippo Naccari

STMicroelectronics
System Research and Applications
Stradale Primosole 50, 95121 Catania, Italy
filippo.naccari@st.com

Ivana Guarneri

STMicroelectronics
System Research and Applications
Stradale Primosole 50, 95121 Catania, Italy
ivana.guarneri@st.com

Salvatore Curti

STMicroelectronics
System Research and Applications
Stradale Primosole 50, 95121 Catania, Italy
salvatore.curti@st.com

Alberto Amilcare Savi

STMicroelectronics
System Research and Applications
Via C. Olivetti 2, 20864 Agrate Brianza, Italy
alberto.savi@st.com

ABSTRACT

Automatic sound understanding tasks have been very popular within research community during the last years. The success of deep learning data driven applications in many signal understanding fields is now moving from centralized cloud services to the edge of the network, close to the nodes where raw data are generated from different type of sensors. In this paper we show a complete workflow for a context awareness acoustic scene classification (ASC) application and its effective embedding process into an ultra-low power microcontroller (MCU). It can widen the capabilities of edge AI applications, from environmental and inertial sensors up to acoustic signals, which require more bandwidth and generate more data. In the paper the entire workflow of such development is described in terms of dataset collection, selection and annotations, acoustic features representation, neural net modeling and optimization as well as the efficient embedding step of the whole application into the target low power 32-bit microcontroller device. Moreover, the overall accuracy of the proposed model and the capability to be real time executed together with an audio feature extraction process shows that such kind of audio understanding application can be efficiently deployed on power constrained battery-operated devices.

Index Terms**—** acoustic scene, deep learning, convolutional neural net, real-time, microcontroller

1. INTRODUCTION

Acoustic scene classification is one of the most common tasks in the field of sound understanding applications. It is the task of automatic labeling acoustic environments from the sound they produce [1]. During the last years several public challenges have been proposed in order to allow research community to improve automatic detections results on a common evaluation baseline [2][3][4]. Several public datasets have been also published with the goal to address sound understanding applications both for the environment classification and acoustic event detection. Some public datasets are composed of sounds produced in particular acoustic environments [5][6][7], whereas some address the complete ontology of sounds in real life conditions in order to enable a full sound based context awareness computing [8][9]. Public challenges based on such datasets drive the submitters through common baseline systems [10] and evaluation metrics [11], which usually lead submitters to propose data modeling solutions aimed to reach the best accuracy scores regardless the computational complexity aspects. More recently, some metrics related to the complexity of submitted machine learning models have been published [12], paving the way to a tradeoff oriented research in terms of accuracy vs complexity data modeling. Moreover, the upcoming diffusion of interconnected nodes, belonging to the Internet of Thing (IoT) ecosystem, leads to design low complexity footprint applications, since most of the IoT nodes are typically battery operated and require a small power consumption application footprint. In this paper we describe a research work related to finding the best trade off in terms of accuracy and complexity for such an edge AI application, aimed to classify on a real time system the acoustic scene as a main audio based subsystem of a multi sensing context awareness application. In order to limit the number of classes to a complete first level taxonomy of possible acoustic environments, we chose to label the sounds coming from three main scenes: indoor, outdoor and in vehicle. The remainder of this paper is organized as follows: section II describe the data collection and labeling process, section III the data modeling process in terms of audio features representation, neural net modeling and inference latency optimization, section IV section describe the whole real time application embedding on a development kit equipped with a digital microphone and low power 32 bit microcontroller unit. Finally, conclusions and potential further developments will be provided.

2. DATASET

The development workflow of deep learning edge AI applications is mainly composed of different steps: i.e. dataset collection, annotation and partitioning, signals feature extraction when not end-to-end approach is followed, machine learning (ML) engine modeling and hyper-parameters tuning on high level development framework, and finally integration and optimization into a selected target platform. In this section we describe the data set used for the before mentioned application. An internal data collection campaign has been done by using a firmware application on a commercially available multi sensor board, named sensorTile development kit [13], which allows to collect and store locally on a µSD mass storage the signals coming from different sensors on board: environmental, inertial, acoustic. For the goal of the ASC application, only data coming from the digital MEMS microphone were used. Acoustic data were collected by minimizing the audio front end processing steps: pdm2pcm conversion, which is present as a hardware feature of the selected recording device, followed by an high pass filter tuned to acoustic bandwidth. All recordings were done at a 16 KHz sampling rate. Different acoustic environments were considered as target for a first option, belonging to 12 different environments: *home, office, café-restaurant, shopping-center, park, city-center, residential-area, car, bus, subway, train, tramway*. The overall recordings were about 29 hours for the 12 different environments but not equalized in terms of sample per class, whereas for the 3 classes aggregation (*indoor, outdoor, in-vehicle*) an equalized training set partitioning was possible, leading to the same length of recordings for each class in the training set. Moreover, in order to avoid an excessive correlation between training, validation and test sets, a manual partitioning has been performed, thus avoiding that audio segments coming from either the same recording or similar acoustic environments could be present in different sets, which could lead to modeling overfit effects. Table I summarizes the recordings distribution and the dataset partitioning into training, validation and test sets, used to develop a neural net model able to classify the selected acoustic scenes.

Table I Audio Recordings (hh:mm:ss)

Class	*Training Set*	*Validation Set*	*Test Set*
Indoor	3:28:01	0:51:17	6:49:37
Outdoor	3:30:23	0:56:32	4:48:23
In-vehicle	3:30:34	0:53:50	4:26:00
Total	**10:28:58**	**2:41:39**	**16:04:00**

3. DATA MODELING

3.1. Audio Features

Differently from the computer vision machine learning tasks, audio signals are rarely fed into ML models as raw data. Therefore, an audio signal feature extraction process is typically performed on raw PCM audio samples in order to extract significant features from signals, before feeding them into a ML model for the training and inference processes [14]. Hence, we adopted a 2D time-frequency audio features representation by calculating log-mel energies from audio PCM sample frames, segmented as follows: every 1024 ms of new samples is calculated a brand-new feature set without time overlap. Each column of the 2D matrix is calculated every 32ms of new samples (512 new samples at 16KHz) with a frame overlap of 50%. This leads to an audio features set, by considering 30 log-mel coefficients for each frame, represented by a 30x32 matrix, which is the input to the convolutional neural net.

3.2. Neural Net Modeling

Different topologies, mostly related to public challenge submissions, were analyzed for dataset modeling. According to our goal of realizing a whole embedded audio based edge AI application, we selected few candidates in terms of neural nets topologies [15][16], which showed the better trade off in terms of complexity scalability, i.e. number of trainable parameters vs accuracy. Therefore, a two-dimensional Convolutional Neural Net (CNN) was selected in order to model the classification task on the collected dataset. To the goal of classifying three different acoustic environments: *indoor, outdoor and in-vehicle,* the model topology which showed best trade off in terms of accuracy vs complexity is composed of two convolutional layers followed by two dense layers, with 7785 trainable parameters. In Table II we report the complete topology summary of the adopted CNN model, trained by using Keras framework. In Fig. 1 is reported the confusion matrix obtained by the trained CNN on the test set, where a percentage average accuracy in 90.16% was obtained.

Table II 2D CNN Topology

Layer	*(type)*	*Out Shape*	*Param #*
conv2d_1	(Conv2D)	(, 28, 30, 16)	160
max_pooling2d_1	(Max-Pool2D)	(, 14, 15, 16)	0
batch_norm_1	(BatchNorm)	(, 14, 15, 16)	64
dropout_1	(Dropout)	(, 14, 15, 16)	0
conv2d_2	(Conv2D)	(, 12, 13, 16)	2320
max_pooling2d_2	(Max-Pool2D)	(, 6, 6, 16)	0
batch_norm_2	(BatchNorm)	(, 6, 6, 16)	64
dropout_2	(Dropout)	(, 6, 6, 16)	0
flatten_1	(Flatten)	(, 576)	0
dense_1	(Dense)	(, 9)	5193
batch_norm_3	(BatchNorm)	(, 9)	36
activation_1	(Activation)	(, 9)	0
dropout_3	(Dropout)	(, 9)	0
dense_2	(Dense)	(, 3)	30
Total params			**7867**
Trainable params			**7785**
Non-trainable params			**82**

3.3. Latency Optimization

Acoustic scene classification does not require a low latency response, since it is devoted to monitoring the environment acoustic conditions which typically change slowly over time. Therefore, some tests were performed by modulating the model latency in order to optimize the classification average accuracy at an acceptable cost in terms of latency, i.e. the classification system time response to a sudden acoustic scene change, and computational power. Hence, a low pass filter has been applied to the inference engine softmax output vector in order to mitigate the effects of mis-classifications. A set of different filter lengths was tested, going from 2 up to 28 consecutive inferences outputs. The average accuracy

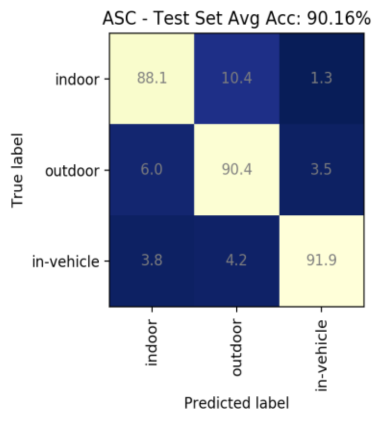

Figure 1: Average Accuracy on Test Set

results obtained by applying such filtering strategy to the inference engine are reported in Table III. The results show that the best average accuracy was obtained by filtering 14 successive inference softmax arrays. The added latency can be considered a more than acceptable cost for most context awareness audio-based applications.

Table III Latency Optimization

Filter Length	-	2	4	7	14	28
Avg. Acc. (%)	90.16	91.80	92.80	93.30	94.14	93.92

4. REAL-TIME EMBEDDED APPLICATION

The reference CNN ASC model for indoor, outdoor, in-vehicle classification has been implemented as real time application on the sensorTile development kit [13], reported in Fig. 2, in order to implement a full proof of concept demo application in the field of sound understanding IoT applications and to prove also that benefits of the data driven deep learning can be extended onto an end-point device. The development kit is equipped with sensing, processing and connectivity components. To the goal of the ASC application the microphone and the microcontroller subsystems were used. The ultra-low-power MCU is the STM32L476JG (based on ARM® Cortex®-M4 RISC core with FPU operating @80 MHz). Its main features are: (100 DMIPS), 128 KB static ram (SRAM), and 1 MB of non volatile memory (NVM). It supports also the onboard digital micro Electro-Mechanical Systems (MEMS) microphone through a direct memory access controller and provides also the pulse density modulation (PDM) bitstream conversion into pulse code modulation (PCM) in hardware. In order to implement the whole application on the target platform three main software tasks were therefore implemented:

- PCM audio buffer samples managing every 32 ms, implemented by handling the audio 16 bit PCM samples coming from the digital MEMS Microphone through a DMA controller.
- Audio Feature Calculation of a single column of the 2D feature matrix every 32 ms (every 512 new samples @16 KHz)
- CNN inference engine, which is performed every 1024 ms of new samples on a 30x32 LogMel Spectrogram features matrix.

The tasks were configured in terms of priority and duty cycle management in order to avoid audio sample loss and allow the real time continuous application running

Figure 2: SensorTile Development Kit. size: 13.5 mm x 13.5 mm

4.1. Audio Buffer Managing

The audio subsystem of the target platform was configured to access by a DMA controller to the microphone samples buffer. Therefore, every 1 ms, 16 new PCM samples are added into a filling buffer, which every 32ms is swapped into a processing buffer, which is used as input to the feature calculation process. Each buffer is composed of 1024 16 bit PCM samples

4.2. Audio Signal Featurization

512 new PCM samples, collected every 32 ms, are moved to the processing buffer and processed on a window of 1024 PCM audio samples to generate a single feature column with a frame length configuration of 1024 samples with a 50% overlap. The process is repeated 32 times in order to generate a brand-new audio feature matrix, which has 30x32 size and is then processed by the 2D CNN inference engine. The steps to calculate a 30 log-mel coefficients from 1024 16 bit PCM samples are here reported:

- Asymmetric Hanning windowing
- 1024 samples fft f32 application
- Power spectrum calculation
- Mel filter banks look-up table application
- Mel energies log amplitude calculation
- Range clipping [-80.0, 0] dB

The output of the process is a single column of the 2D feature representation, composed by 30 LogMel coefficients.

4.3. Inference Engine

As mentioned above, the 2D CNN model was trained by using Python based framework Keras. In order to map the inference engine to C code, an AI extension pack for STM32CubeMX tool was used [17], able to generate an STM32-optimized library from neural network models pre-trained on most common high level deep learning frameworks. Post training quantization and a validation on target platform were also performed for the inference engine process related to the CNN selected model. Moreover, the post training model quantization process, optimized on a representative input/output subset allowed to significantly reduce the overall library complexity for the target platform at a minimum cost in terms of model accuracy. In Table IV the comparison, in terms of resources on the target platform of both 32 bit floating point and integer 8 bit quantized models, is reported.

Table IV CNN Complexity Figures

Model	*Avg. Acc. (%)*	*NVM (KB)*	*RAM (KB)*	*Inference Time (ms)*
ASC CNN f32	90.16	30.81	17.41	81.505
ASC CNN int8	89.17	7.71	4.94	36.022

4.4. Application Complexity

A complexity estimation process of the whole embedded ASC application has been also performed. We report in table V the requirements in terms of NVM, RAM and execution time on the target platform of the main software tasks described before: audio buffer managing, audio feature calculation and the CNN int8 fixed point post training quantized inference engine coupled with the post processing filter. According to the reported software tasks frequency distribution and their execution time on the reference platform, we then estimated a 13.6% MCU duty cycle, which shows that such kind of sound understanding application can effectively run onto an ultra-low power MCU.

Table V Application Complexity Figures

Task	*Task period (ms)*	*NVM (KB)*	*RAM (KB)*	*Exec. Time (ms)*
PCM buffer managing	32	-	8.00	0.484
Feature Extraction	32	7.87	7.88	2.750
Inference Engine + PP	1024	7.71	5.03	36.072

5. CONCLUSIONS

In this paper we showed the results of the entire development workflow of a sound understanding deep learning application on an end-point microcontroller device. We showed that it is possible implementing data driven effective applications even if power-constrained devices, e.g. wearable and headsets devices. The whole application, composed of audio signal conditioning, feature extraction and 2D convolutional neural net inference engine, requires a fraction of a typical consumer ultra-low power 32 bit microcontroller computational power. Therefore, it can run continuously thus enabling advanced sounds recognition context awareness use cases. The low memory footprint as well as the duty cycle requirements allow also to address for future developments either multi inference engine applications or multi-modal sensors fusion based higher complexity applications.

6. REFERENCES

[1] Barchiesi, D., Giannoulis, D., Stowell, D., Plumbley, M. D. (2015). “Acoustic scene classification: Classifying environments from the sounds they produce”, *IEEE Signal Processing Magazine*, 32(3), 16-34.

[2] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, “Detection and classification of acoustic scenes and events,” *IEEE Trans. on Multimedia*, vol. 17, no. 10, pp. 1733–1746, October 2015.

[3] Mesaros, A., Heittola, T., Virtanen, T. , “Acoustic scene classification: an overview of DCASE 2017 challenge entries”, *In 2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)* (pp. 411-415). IEEE, Sept. 2018.

[4] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, “Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, Feb 2018

[5] Salamon, J., Jacoby, C., & Bello, J. P., “A dataset and taxonomy for urban sound research”, *in Proceedings of the 22nd ACM international conference on Multimedia* (pp. 1041-1044), November 2014.

[6] A. Mesaros, T. Heittola, and T. Virtanen, “A multi-device dataset for urban acoustic scene classification,” *in Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, Nov. 2018.

[7] Mesaros, A., Heittola, T., & Virtanen, T., “TUT database for acoustic scene classification and sound event detection”. *In 2016 24th European Signal Processing Conference (EUSIPCO)* (pp. 1128-1132). IEEE, August 2016.

[8] Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Ritter, M., “Audio set: An ontology and human-labeled dataset for audio events”. *In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 776-780). IEEE. August 2016.

[9] Fonseca E., Pons Puig J., Favory X., Font Corbera F., Bogdanov D., Ferraro A., and Serra, X. “Freesound datasets: a platform for the creation of open audio datasets”,. In Hu X, Cunningham SJ, Turnbull D, Duan Z, editors. *Proceedings of the 18th ISMIR Conference, Suzhou, China, International Society for Music Information Retrieval*. October 2017.

[10] Mesaros A., Heittola T., Diment A., Elizalde B., Shah A., Vincent E., and Virtanen, T. ,“DCASE 2017 challenge setup: Tasks, datasets and baseline system”, November 2017.

[11] Mesaros A., Heittola T., amd Virtanen, T., “Metrics for polyphonic sound event detection”, Applied Sciences, 6(6), 162, 2016.

[12] Mesaros, A., Heittola, T., Virtanen, T., “Acoustic scene classification in DCASE 2019 challenge: Closed and open set classification and data mismatch setups”, 2019.

[13] STEVAL-STLKT01V1, SensorTile board development kit https://www.st.com/en/evaluation-tools/steval-stlkt01v1.html

[14] Mesaros, A., Diment, A., Elizalde, B., Heittola, T., Vincent, E., Raj B., Virtanen, T. (2019). “Sound event detection in the DCASE 2017 challenge”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(6), 992-1006.

[15] Valenti M., Diment A., Parascandolo G., Squartini S., Virtanen, T. “DCASE 2016 acoustic scene classification using convolutional neural networks”, *in Proc. Workshop*

Detection Classif. Acoust. Scenes Events (pp. 95-99), September 2016.

[16] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, “A convolutional neural network approach for acoustic scene classification,” *in Proc. of 2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1547–1554, May 2017.

[17] X-CUBE-AI, AI expansion pack for STM32CubeMX https://www.st.com/en/embedded-software/x-cube-ai.html