Reducing AI complexity using ear bio-inspired primitives

Adrien Deverin*†‡, Valentin Gies*‡, Sebastián Marzetti*‡, Valentin Barchasz*‡, Hervé Glotin†‡,
*Aix Marseille Univ, Université de Toulon, CNRS, IM2NP, Marseille, France
†Aix Marseille Univ, Université de Toulon, CNRS, LIS, Toulon, France
‡Centre d'Intelligence Artificielle en Acoustique Naturelle, cian.lis-lab.fr, Université de Toulon, France

***Abstract*—The complexity of artificial intelligence (AI) raises significant challenges in developing embedded detection systems, particularly in terms of power consumption. In contrast, biological auditory perception addresses these issues efficiently. Drawing inspiration from biological primitive extraction in the auditory system, this article presents a new method for drastically reducing energy required for acoustic signal processing and classification. This method could also be applied to more general problems.**

To assess the efficiency of the proposed algorithm, experiments were conducted using the Google Speech Command Dataset (GSCD), focusing on 4 and 8 classes with added noise. Mimicking the structure of the cochlea, system training starts with 64 analog primitives, which are pruned sequentially, retaining only the most relevant ones for classification. This pruning relies on a novel neural network layer called "Line Gain." Results demonstrate that the proposed algorithm significantly reduces total energy consumption by 82%, while maintaining comparable accuracy levels (greater than 90%).

***Index Terms*—Embedded Artificial intelligence, Primitives Neural Network, Ultra low power, Speech Detection, Inputs Learning.**

I. Introduction

Over the past decade, neural networks have played a crucial role in driving the evolution of artificial intelligence (AI), particularly in the domain of high-performance detection and classification. Convolutional neural networks (CNNs) have emerged as pivotal components, enabling tasks such as image and sound classification [1]. Inspired by biomimetic analogies with human vision and brain functions, CNNs have effectively narrowed the gap between human and AI performance levels. However, a significant disparity in energy consumption between human and AI systems remains evident. Extensive research has been conducted to optimize CNN structures, with studies focusing on reducing operation consumption for embedded applications through quantization methods [2], [3] and optimizing hyperparameters for enhanced accuracy [4].

Despite these advances, the gap in energy consumption between digital and biological architectures raises significant challenges, especially for embedded systems. This energy consumption gap is caused by the analog nature of the human brain's operations, which consume significantly less energy compared to digital processing tasks, particularly for middle to high-frequency processes [5]. Additionally, previous studies have underscored that feature extraction and high sampling rate Analog-to-Digital Converters (ADCs) rank among the most energy-intensive tasks in fully digital embedded machine learning applications [6]–[9].

Drawing insights from biology, particularly from specialized organs such as the ears and eyes in our brain, which serve as dedicated sensors filtering information at low energy cost [10], presents promising ways for reducing overall energy consumption. Efforts to mimic these organs have been explored, aiming to address the energy consumption challenges in AI systems [11], [12]. Furthermore, proposals for optimizing CNN structures by reducing input selections using genetic algorithms (GA) or generalized regression neural networks (GRNN) have been investigated [13], [14]. However, it is essential to recognize that nature has undergone billions of years of optimization, making it challenging to determine which primitives are truly useful in specific cases.

In previous research in sound detection for biodiversity monitoring [8], [15], always on features extraction using analog primitives have been investigated. This contribution goes further by selecting only relevant analog primitives in order minimize overall power consumption while maintaining satisfactory accuracy levels.

Although this approach is demonstrated here for sound classification, it is based on a novel layer named "Line Gain" that can be used for other types of applications based on different neural structures.

II. Proposed architecture and algorithm

A. Principles

Operating within the constraints of ultra-low power embedded AI, a mixed analog-numeric architecture like the Primitive Neural Network (PNN) appears to be the solution to our problem. For the primitives extraction part of our architecture, we utilize a system that captures sound spectrograms through bandpass filters (BP) followed by a low-pass peak detector (LPPD) in analog form (Fig. 1). This setup closely mimics the passive functionality of the human cochlea, as indicated in previous research [10]. Similarly, using Mel-frequency cepstral coefficients through the numerical Fast Fourier Transform (FFT) is a conventional method in fully digital voice detection devices to obtain inputs features for a neural network [12], [16]. However, performing the FFT requires recording the entire signal at a high frequency (> 16 kHz) and involves numerous operations ($n \cdot \log_2(n)$). As explained in the introduction, analog computing is approximately 10^2 times more energy-efficient [5]. This efficiency is why our approach consumes significantly less energy while yielding similar results. The analog features are subsequently captured and recorded by an Analog-to-Digital Converter (ADC) at a low frequency before being analyzed by a Convolutional Neural Network (CNN).

Acknowledging that significant detections are infrequent, a forthcoming article will delve into analog criteria for triggering the ADC and subsequently the CNN. This strategy aims to minimize their substantial energy consumption, ensuring their activation only when necessary. Although not elaborated here, the ADC is currently triggered at 100Hz for 90 samples every time. Given that all data samples are sounds with durations of less than 1 second, this sampling rate is sufficient to gather the required information. Additionally, because the ADC operates at a low frequency, ultra-low power requirements can be still maintained.

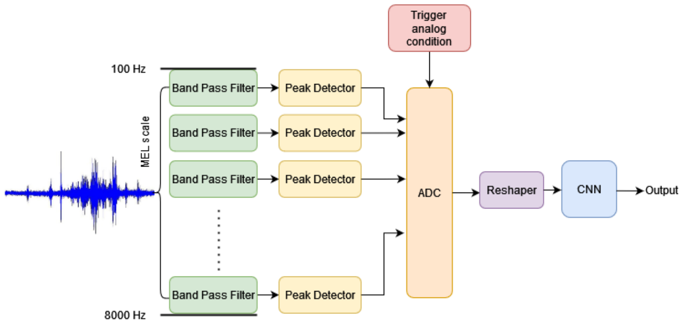


Fig. 1: Structure of the System

This process is becoming increasingly recognized and has already been employed in recent articles [17], [18] in speech recognition task.

However, cochlea sensitivity in terms of frequency differs between species, reflecting their specific detection needs. In this article, the identification of filter relevance is crucial for customizing detection, which may not be human, and for eliminating unnecessary filters. Starting with 64 distinct frequency bands, CNN is trained on these spectrograms. Then, systematically eliminating one band at a time, the dense layers only are retrained while retaining the weights of convolutional layers at each iteration. At each stage, the model is saved, and the algorithm halts either when accuracy significantly decreases or when the number of layers becomes insufficient for convolution operations. Authors' objective is to develop a method that allows pruning frequency band filters according to their relevance in the optimal way.

B. Implementation

To demonstrate our example, we chose to utilize the Google Speech Command Dataset (GSCD) [19], focusing on the classification of four distinct words: ”Yes”, ”No”, ”Left”, and ”Right”. Within this dataset, all recordings are subjected to noise and spoken by various individuals. Each word class is represented 2368 times, resulting in a dataset comprising a total of 9472 one-second-long elements. Furthermore, at the end of this article, another experiment as been conducted by adding 4 new words: ”five”, ”six”, ”seven”, and ”eight”, resulting in a total of 8 classes. For a correct learning, the dataset has been divided into three parts:

- 10% of the data were reserved for the evaluation of our final performance. This ensures that our results in Section III are not affected by overfitting.
- The rest of the data was subdivided into two parts for our training. The validation split has been set at 20%.

The starting architecture of our CNN (Table I) and its corresponding hyperparameters (Table II) are outlined below. The ”Line Gain” is a specific layer that will be explained in the next section. The convolutional kernel size is 3×3 in ”valid” mode. The memory value for each layer is estimated in bytes, considering float-type variables, and represents the memory footprint during inference. This estimate includes the number of variables in the layer as well as the residual values required for the subsequent layer. FLOPs (Floating Point Operations) indicate the computational load for each inference, with each multiplication and addition counted as one operation.

TABLE I: CNN Architecture

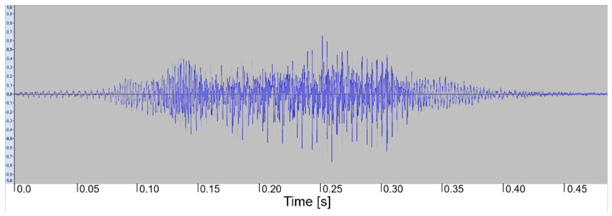
Layer	**Output Shape**	**Activation**	**Memory**	**FLOPs**
Input	(-, 64, 90, 1)	-	23 040	0
Line Gain	(-, 64, 90, 1)	ReLU	23 296	5 760
Conv2D (kernel 3x3)	(-, 62, 88, 8)	ReLU	174 912	436 480
Conv2D (kernel 3x3)	(-, 60, 86, 8)	ReLU	167 456	3 302 400
Conv2D (kernel 3x3)	(-, 58, 84, 8)	ReLU	158 240	3 118 080
Flatten	(-, 38976)	-	0	0
Dense	(-, 128)	ReLU	19 956 736	4 989 056
Dropout (0.2)	(-, 128)	-	512	0
Dense	(-, 32)	ReLU	16 640	4 128
Dense	(-, 4)	Softmax	544	132
Total			20.5 MB	11 856 036

TABLE II: CNN Training Hyperparameters

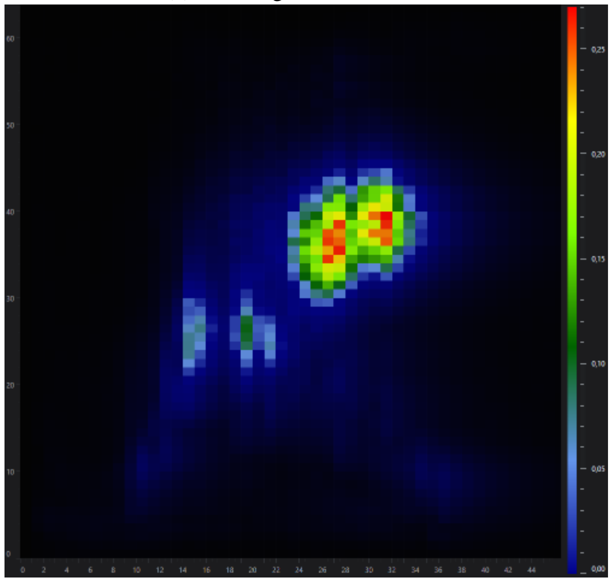
Epoch	15
Validation Split	0.2
Batch Size	32
Optimizer	Adam
Loss	CategoricalCrossentropy
Learning Rate	0.001

Alternative structures exist in the state of the art. Recurrent Neural Networks (RNNs) are sometimes used instead of, or in conjunction with, CNNs, but they generally incur higher operational costs [20]. Although this is not developed in this article, keep in mind that RCNNs could be a variant structure for performance enhancement and are compatible with our main idea. The same applies to all quantization methods that work on reducing operation costs; while not covered in this article, they could further reduce the energy consumption of the CNN.

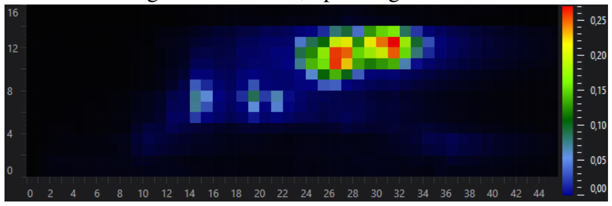
During the process, we start by segmenting our data into 64 frequency bands distributed along the Mel scale, spanning from 100 to 8000Hz, like in human cochlea. Each band undergoes processing with a peak detector set at cutting frequencies of 20Hz. Our ADC records 90 samples per second at 100Hz, generating images sized 64 x 90 pixels for analysis by our CNN (Fig. 2 B).



(a) The original audio sound



(b) Spectrogram of the sound divided into 64 frequency bands distributed along the Mel scale, spanning from 100 to 8000Hz.



(c) Spectrogram with the 48 initial bands suppressed.

Fig. 2: Example of spectrogram collected analogically and processed by the CNN depending on the number of inputs

After each training session on specified frequency bands, referred to as a "generation" in this article, the deletion of the less significant input is performed. Subsequently, a new CNN is constructed with the reduced input size, and it is trained again for 15 epochs. The convolutional weights are inherited from the previous generation, while the dense layers are initialized randomly. This methodology is grounded in empirical findings. It has been observed that retraining the old dense weights (for the first dense layer, the part corresponding to the deleted band in its weight matrix is suppressed) tends to trap the model in local minima, significantly compromising accuracy across generations. Conversely, retraining the convolutional layer is unnecessary as the spectrogram remains relatively consistent between generations. This aligns with biological phenomena, where convolutional layers act akin to type I spiral ganglion neurons (SGN I), passively aggregating signals from the cochlea before forwarding them to the brain without significant analysis [10]. These neurons exhibit limited flexibility. In contrast, dense layers resemble brain neurons, which are subject to cerebral plasticity.

The pseudo code of our algorithm is described below (Algorithm 1).

Algorithm 1 Cnn inputs pruning

```
Require: cnn_structure, $x_{\text{values}}$, $y_{\text{values}}$, epochs
  Create the cnn
  Separate the dataset into two parts ($x_{\text{train}}$, $x_{\text{eval}}$)
  Train the cnn on $x_{\text{train}}$
  Evaluate the accuracy on $x_{\text{eval}}$
  while not termination condition do
      Save the cnn weights
      * Get the less significant input
      Delete the associated information in $x_{\text{train}}$ and $x_{\text{eval}}$
      Recreate the cnn with the changed input size
      Fill the cnn with the previous weights
      Train the cnn on $x_{\text{train}}$
      Evaluate the accuracy on $x_{\text{eval}}$
  end while
```

C. *Line Gain : a novel neural layer*

The process of determining the sequence for eliminating filters ((*) 7th line in algorithm 1) is a critical aspect and challenge discussed in this article. Initially, one approach considered prioritizing the deletion of filters based on the mean gain of data within their frequency band. However, this method encountered a significant issue: certain crucial aspects of the sounds, despite having less phonetic emphasis, should not be overlooked in favor of noise components, which might dominate in specific frequency bands. This naive method could lead to prioritizing noise over informative content.

Another approach for selecting the input is to choose it statically, as proposed by some articles, using various forms of genetic algorithm. However, in the majority of cases, this method presents significant time issues and may not converge correctly. Consequently, in this article, backpropagation is utilized, and a new type of layer named 'Line Gain' is introduced. This layer learns from the inputs and prioritizes their deletion accordingly.

This layer is incorporated into the CNN structure after the input layer. Its function is to amplify certain lines of the spectrogram with the objective of converging weights to an amplification of 0. At the end of the training, a smaller amplification indicates a lesser importance of the corresponding line, thereby increasing the priority for deletion. The different weights will be sorted, providing the correct order of the primitives to delete. To induce certain weights of the model to converge to zero, an L1 norm is included in the loss function of this layer (Equation (1)).

$$\text{New Loss} = \text{Original Loss} + \lambda \sum_{i=1}^{n} |w_i| \,. \quad (1)$$

Here, λ represents the regularization parameter, and w_i denotes the weights of the layer. The L1 regularizer term $\sum_{i=1}^{n} |w_i|$ encourages sparsity in the weights by penalizing large values, effectively promoting some weights to converge towards 0 during training. This regularization technique helps prevent overfitting and encourages simpler models. For this to be effective, setting a small regularization strength is recommended (λ is set at 0.01 in this article).

III. Results

Initially, around 89% accuracy is achieved by our model (88% for 8-class detection) with nearly 5 million parameters, stabilizing after 15 epochs. The original structure and comparison in energy consumption with other devices are more thoroughly described in our previous work [21].

Following each deletion, our model has been retrained for an additional 15 epochs, as detailed in Section II-B. Figure 3 depict in red and green the accuracy evolution with the reduction in the number of inputs from 64 to 7 (further reduction is unfeasible due to convolutions as shown in table I). Each deletion results in pruning 86,016 parameters. This accuracy is computed on a subdataset unseen during CNN training. Table III summarized the effect of filter deletion on energy consumption gain of the CNN and its equivalent accuracy. Note that due to the CNN structure, the number of operations (and thus the energy consumption) follows a linear relationship with the number of inputs, resulting in a proportional energy gain. This analysis has been performed for both 4- and 8-class classifications.

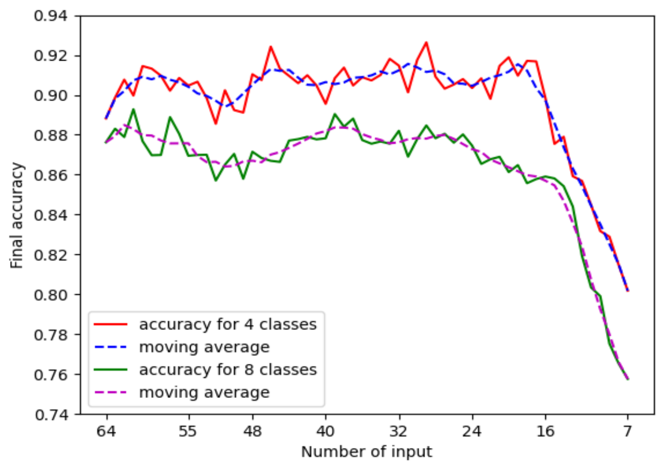


Fig. 3: Effect of filter deletion on CNN mean accuracy for 4- and 8-classes

TABLE III: Effect of Filter Deletion on CNN Consumption

Filters	Accuracy		Memory	MFLOPs	Energy Gain
64	0.89	0.88	20.5 MB	11.86	0.0%
56	0.90	0.88	17.6 MB	10.24	13.7%
48	0.89	0.77	14.8 MB	8.62	27.3%
40	0.90	0.88	12.0 MB	7.00	41.0%
32	0.92	0.88	9.2 MB	5.39	54.5%
24	0.90	**0.89**	6.4 MB	3.78	68.1%
16	**0.92**	0.86	3.5 MB	2.16	**81.7%**
10	0.85	0.80	1.4 MB	0.95	92.0%
7	0.80	0.76	0.38 MB	0.34	97.1%

The words used in this classification are 'yes', 'no', 'left', and 'right' for the 4-class classification. For the 8-class classification, the classes 'five', 'six', 'seven', and 'eight' are added.

During the initial generations, a stabilized improvement has been observed in accuracy but with a significant decrease in consumption. In our 4-class example, after eliminating 48 filter bands, which correspond to 3/4 of our initial inputs, and achieving an energy gain of 81.7%, a decline in performance has been observed. Specifically, only 8% of accuracy has been lost while halving the consumption. The compromise between accuracy and consumption will depend on your constraints, but in any case, the final consumption will drastically decrease compared to the initial setup.

Compared with other state-of-the-art results for embedded speech recognition on microcontrollers [1], [16]–[18], [20], [22], our approach maintains accuracy within the range of 89-94% while improving energy consumption. Additionally, our method is more versatile and can be applied to a broader range of subjects. This demonstrates that sometimes, irrelevant information can penalize our classification, and that highly optimized CNNs with intelligently selected information can outperform larger CNNs. A second experimentation confirmed these results with adding a fixed constant noise to virtuals new frequency bands. These new informations were not the first to be eliminated by our system. This consistent behavior led us to hypothesize that certain frequency bands are perceived as random by our system, resulting in a greater penalty for detection than constant information that essentially provides no features.

IV. Conclusion

In this article, the constraints regarding the energy efficiency of fully digital embedded machine learning applications are delineated. Taking cues from the energy efficiency of analog processing and the discerning feature extraction observed in biological systems such as the cochlea, a Primitive Neural Network (PNN) architecture has been implemented to achieve ultra-low-power embedded artificial intelligence for sound classification. Additionally, we present a general method to diminish the number of inputs and variables in a Convolutional Neural Network (CNN) while upholding satisfactory accuracy. A new neural layer, "Line Gain," is introduced, designed to prioritize the removal of frequency bands based on their significance, guided by backpropagation.

To assess the efficacy of our algorithm, the Google Speech Command Dataset (GSCD) is utilized. The results demonstrate the algorithm's effectiveness in substantially reducing total power consumption while maintaining accuracy. It is anticipated that this algorithm will not only enhance sound detection capabilities but also contribute to reducing the energy consumption of various other deep learning algorithms, potentially leading to the development of more efficient and accurate embedded machine learning systems across diverse applications.

V. Acknowledgments

This project is granted by Agence Nationale de la Recherche (ANR) SYLVANIA ANR-21-CE04-0019 towards environmental machine listening, and ANR-20-CHIA-0014 with DGA innovation Defense, towards AI for subsea machine listening (Pi H.G.). H.G. also received the European BIODIVERSA EUROPAM 2023-2026 and SeaStMar MARITTIMO 2024-2027 grants on acoustic megafauna surveys with low power listening machine. We thank Pole INPS of the University of Toulon for useful discussions.

References

[1] M. N. Miah and G. Wang, "Keyword spotting with deep neural network on edge devices," in *IEEE 12th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, 2022, pp. 98–102.

[2] M. Shah, S. Arunachalam, J. Wang, D. Blaauw, D. Sylvester, H.-S. Kim, J.-s. Seo, and C. Chakrabarti, "A fixed-point neural network architecture for speech applications on resource constrained hardware," *Journal of Signal Processing Systems*, vol. 90, 05 2018.

[3] B. Liu, Z. Wang, W. Zhu, Y. Sun, Z. Shen, L. Huang, Y. Li, Y. Gong, and W. Ge, "An ultra-low power always-on keyword spotting accelerator using quantized convolutional neural network and voltage-domain analog switching network-based approximate computing," *IEEE Access*, vol. 7, pp. 186 456–186 469, 2019.

[4] J.-H. Han, D.-J. Choi, S.-U. Park, and S.-K. Hong, "Hyperparameter optimization for multi-layer data input using genetic algorithm," in *IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA)*, 2020, pp. 701–704.

[5] S. Marzetti, V. Gies, V. Barchasz, H. Barthélemy, and H. Glotin, "Comparing analog and digital processing for ultra low-power embedded artificial intelligence," in *IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS)*, 2022, pp. 112–116.

[6] G. Cerutti, R. Prasad, and E. Farella, "Convolutional neural network on embedded platform for people presence detection in low resolution thermal images," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 7610–7614.

[7] V. Gies, S. Marzetti, V. Barchasz, H. Barthélemy, and H. Glotin, "Ultra-low power embedded unsupervised learning smart sensor for industrial fault classification," in *IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, 2021, pp. 181–187.

[8] S. Marzetti, V. Gies, V. Barchasz, P. Best, S. Paris, H. Barthelemy, and H. Glotin, "Ultra-low power wake-up for long-term biodiversity monitoring," 01 2021, pp. 188–193.

[9] S. Marzetti, V. Gies, V. Barchasz, H. Barthelemy, H. Glotin, E. Kussener, and R. Vauche, "Embedded learning for smart functional electrical stimulation," in *IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2020, pp. 1–6.

[10] B. J. Marin N, Lobo Cerna F, "Signatures of cochlear processing in neuronal coding of auditory information." *Mol Cell Neurosci.*, 2022.

[11] L. Cheng, L. Gao, X. Zhang, Z. Wu, J. Zhu, Z. Yu, Y. Yang, Y. Ding, C. Li, F. Zhu, G. Wu, K. Zhou, M. Wang, T. Shi, and Q. Liu, "A bioinspired configurable cochlea based on memristors." *Front. Neurosci.*, 2022.

[12] W. Shan, M. Yang, T. Wang, Y. Lu, H. Cai, L. Zhu, J. Xu, C. Wu, L. Shi, and J. Yang, "A 510-nw wake-up keyword-spotting chip using serial-fft-based mfcc and binarized depthwise separable cnn in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 151–164, 2021.

[13] P. Leahy, G. Kiely, and G. Corcoran, "Structural optimisation and input selection of an artificial neural network for river level prediction," *Journal of Hydrology*, vol. 355, no. 1, pp. 192–201, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022169408001522

[14] A. B. Dariane and M. M. Behbahani, "Development of an efficient input selection method for nn based streamflow model," *Journal of Applied Water Engineering and Research*, vol. 11, no. 1, pp. 127–140, 2023.

[15] S. Marzetti, V. Gies, P. Best, V. Barchasz, S. Paris, H. Barthélémy, and H. Glotin, "A 30 w embedded real-time cetacean smart detector," *Electronics*, vol. 10, no. 7, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/7/819

[16] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," 2018.

[17] K. Kim, C. Gao, R. Graça, I. Kiselev, H.-J. Yoo, T. Delbruck, and S.-C. Liu, "A 23-w keyword spotting ic with ring-oscillator-based time-domain feature extraction," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 11, pp. 3298–3311, 2022.

[18] F. Chen, K.-F. Un, W.-H. Yu, P.-I. Mak, and R. P. Martins, "A 108-nw 0.8-mm2 analog voice activity detector featuring a time-domain cnn with sparsity-aware computation and sparsified quantization in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 11, pp. 3288–3297, 2022.

[19] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv 1804.03209*, 2018. [Online]. Available: https://blog.research.google/2017/08/launching-speech-commands-dataset.html

[20] Y. Wei, Z. Gong, S. Yang, K. Ye, and Y. Wen, "EdgeCRNN: an edge-computing oriented model of acoustic feature enhancement for keyword spotting," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 3, pp. 1525–1535, Mar. 2022. [Online]. Available: https://doi.org/10.1007/s12652-021-03022-1

[21] S. Marzetti, V. Gies, V. Barchasz, H. Barthélémy, and H. Glotin, "Analog features extractor for ultra-low power embedded ai listening and keyword spotting," in *IEEE International Conference on Artificial Intelligence Circuits and Systems (IEEE AICAS)*, 2024.

[22] A. Coucke, M. Chlieh, T. Gisselbrecht, D. Leroy, M. Poumeyrol, and T. Lavril, "Efficient keyword spotting using dilated convolutions and gating," 2019.