FPGA Implementation of PoolFormer Network using Python-Driven High-Level Synthesis Framework for Edge-AIoT Speech Recognition

Tiancheng Cao, *Member, IEEE,* Zhongyi Zhang, Wei Soon Ng, *Student Member, IEEE,* Wang Ling Goh, *Senior Member, IEEE,* and Yuan Gao, *Member, IEEE*

***Abstract*—This paper presents an edge-AIoT speech recognition system which is based on a new spiking feature extraction method and a PoolFormer neural network optimized for implementation on FPGA hardware. A Python-driven High-Level Synthesis (HLS) flow is adopted to accelerate software-to-hardware conversion for fast validation, demonstrating the potential of FPGA-based solutions in edge applications. This work provides a holistic end-to-end solution for ultra-low-power speech recognition, leveraging HLS to bridge the gap between software and hardware development. Implemented in a Xilinx PYNQ-Z2 FPGA board, this optimized PoolFormer model achieved an speech recognition accuracy rate of 95.41% on the 35-class Google Commands dataset with a parameter count of 39k.**

***Index Terms*—High-level synthesis, PYNQ, Speech recognition, PoolFormer, Edge AIoT**

I. INTRODUCTION

Deep neural networks (DNNs) have been widely adopted in Artificial Intelligence of Things (AIoT) applications [1-3]. The growing demand for efficient, high-performance speech interfaces in applications ranging from home automation to smart cities has led developers to explore various hardware platforms, including System-on-Chip (SoC) [4] and hardware accelerator [5]. However, challenges such as high cost, power consumption, and design complexity limit the adoption of these solutions. In contrast, Field-Programmable Gate Arrays (FPGAs) offer high parallelism, low power and hardware reconfigurability, making them a promising alternative solution for edge AIoT tasks [6-7].

Recent research focused on optimizing Long Short-Term Memory (LSTM) networks for FPGA deployment by addressing the issues related to computational complexity, memory footprint, and power consumption [8]. Meanwhile, Transformer-based models—which provide superior parallel computation, better capture long-range dependencies, and enhanced interpretability—are gaining favor over LSTMs [9]. Despite these advantages, traditional Verilog development faces challenges when compressing extensive Transformer architectures for software-hardware co-design, highlighting the need for a high-level synthesis (HLS) framework tailored for edge AIoT speech recognition.

Motivated by the success of PoolFormer in edge computing, this study introduces a Python-driven software-hardware co-design framework that integrates a spiking feature extraction module with a PoolFormer model for speech recognition on FPGA hardware, specifically targeting the 35-class Google Commands dataset [10]. This work demonstrates not only the viability of FPGAs for efficient edge speech recognition and the advantage of HLS in accelerating the validation of software algorithm in hardware, but more importantly, the main novelty of this approach lies in the co-design and integration of an ultra-low-power analog spiking feature extraction module with a highly quantized PoolFormer model. This tightly coupled system achieved a superior tradeoff between power consumption, hardware resource utilization, and recognition accuracy, making it well-suited for edge-AIoT speech systems.

The rest of the paper is organized as follows. Section II introduce the structure of proposed edge speech recognition system with a novel spiking feature extraction. Section III presents FPGA implementation of PoolFormer with PYNQ platform. Section IV shows the simulation results and Section V concludes the paper.

II. EDGE SPEECH RECOGNITION SYSTEM

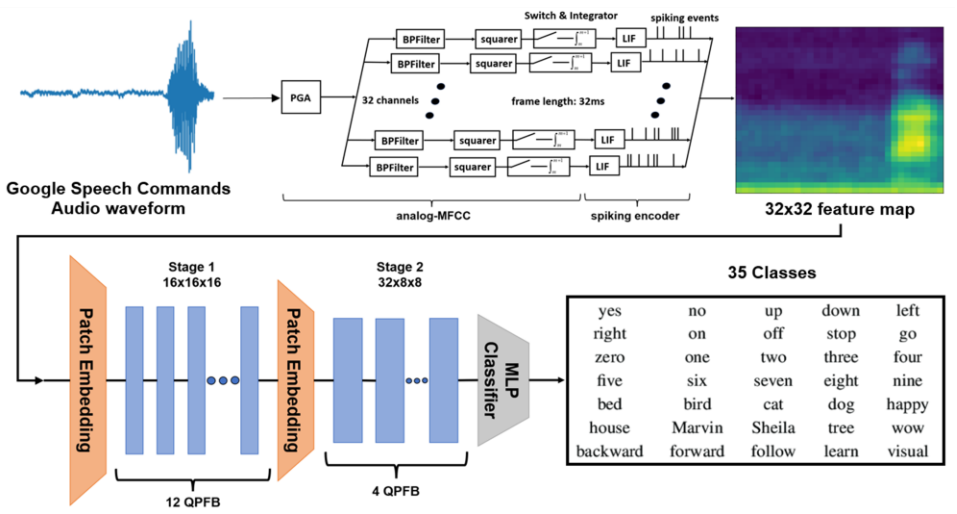
Fig. 1 shows the overall block diagram of the proposed PoolFormer edge speech recognition system. Firstly, the time-domain signal undergoes a novel spiking feature extraction module to extract a frequency domain feature map. Subsequently, this feature map will be processed by a PoolFormer neural network for classification.

A. Spiking Feature Extraction

A novel Spiking Feature Extraction (SFE) module is proposed, combining analog MFCC computation with a spiking encoder to

This research is supported by Agency for Science, Technology and Research (A*STAR), Singapore under the High Linearity Silicon Germanium Photonic Modulator for 6G Analog Radio over Fiber Project, Grant No. M24M8b0004 and the Eric and Wendy Schmidt AI in Science Postdoctoral Fellowship, a Schmidt Futures program. (Corresponding author: Yuan Gao)

Tiancheng Cao, Zhongyi Zhang, Wei Soon Ng, and Goh Wang Ling are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, 639798 (e-mail: tiancheng.cao@ntu.edu.sg; zhongyi001@e.ntu.edu.sg; weisoon001@e.ntu.edu.sg; ewlgoh@ntu.edu.sg).
Yuan Gao is with the Institute of Microelectronics (IME), Agency for Science, Technology and Research (A*STAR), 2 Fusionopolis Way, Innovis #08-02, Singapore 138634, (e-mail: gaoy@a-star.edu.sg).



Stage	Layer Specification	Quantized PoolFormer
Patch Embedding	Patch Size	3×3, stride 2
	Embed. Dim.	16
1	Pooling Size	3×3, stride 1
	MLP Ratio	4
	# Block	12
Patch Embedding	Patch Size	3×3, stride 2
	Embed. Dim.	32
2	Pooling Size	3×3, stride 1
	MLP Ratio	4
	# Block	4
Classifier	MLP	128×35

Fig. 1. The overall structure of the Edge-AIoT speech recognition framework with a 2-stage PoolFormer structure including 12 and 4 Quantized PoolFormer Blocks (QPFB). The layer specifications of PoolFormer are shown on the right.

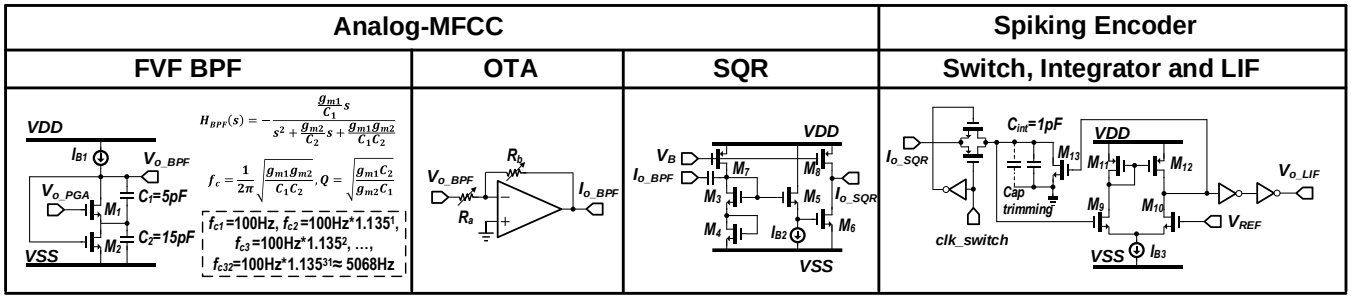


Fig. 2. Detailed circuit design of Spiking Feature Extraction based on analog-MFCC.

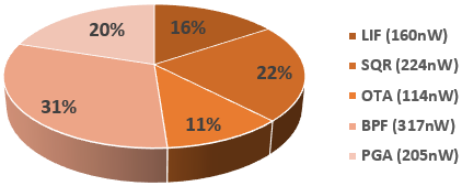


Fig. 3. The power breakdown of proposed Spiking Feature Extraction.

achieve high accuracy, robustness, and ultra-low power. The circuit, designed and validated at schematic level in TSMC 0.13 µm CMOS with all transistors in subthreshold, provides valuable insights into spectral fidelity and energy efficiency, though it does not fully capture real silicon effects such as layout parasitics and PVT variations. While our analog approach exhibits lower stopband attenuation and frequency selectivity than digital filters, it is still adopted here for its significant power savings and effective feature representation. The speech signal is amplified by a programmable gain amplifier (PGA) [11], then decomposed into Mel-scaled spectral bands by a 32-channel bandpass filter (BPF) bank—serving as an analog alternative to FFT. Squarer, switch, and integrator circuits then produce 32-dimensional analog MFCC-like features over 32 ms frames, and a leaky integrate-and-fire (LIF) block encodes these as asynchronous spikes.

The analog Spiking Feature Extraction circuit, shown in Fig.2. A flipped voltage follower (FVF) BPF structure is selected because it inherent current-reuse capability to conserve power consumption [13]. Additionally, its single-branch biasing and identical nMOS transistors enhance matching characteristics, ensuring robustness against process variations. In this work, the Q-factor is set to 1.7. The BPF voltage output is converted to current via a simple OTA, then squared in the SQR block, which employs a current-reuse Stacked Translinear Loop – a nanowatt-level power-efficient solution [14]. Switching, integration, and LIF functions are realized through a comparator-driven charge/discharge mechanism.

Overall, an acoustic signal undergoes the following feature extraction through the aforementioned circuit to obtain the energy:

$$E(i) = \frac{1}{T_{INT}} \int_{(i-1)T_{INT}}^{iT_{INT}} |y(t)|^2 \, dt \qquad (1)$$

where $y(t)$ is the input acoustic signal, T_{INT} is the frame length, and $E(i)$ represents the energy extraction over the i^{th} frame. This equation effectively computes the average energy of the input signal over each frame by integrating the square of the signal's amplitude over the given time interval T_{INT}.

The total power consumption of the proposed analog spiking feature extraction is 1.02µW during simulation as shown in Fig. 3. Fig. 4 (a) shows its frequency response from the PGA input to the LIF output, exhibiting the expected bandpass characteristics across 32 channels. For comparison, Fig. 4 (b) illustrates the frequency response of a digital Mel filterbank. As observed, the proposed analog feature extraction demonstrates inferior performance in the stopband attenuation compared to

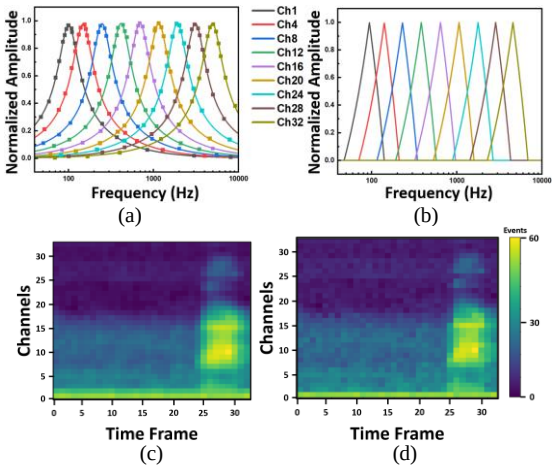


Fig. 4. (a) frequency response of analog spiking feature extraction. (b) frequency response of digital Mel bandpass filterbank. (c) spectrogram of keyword "Bed" over different channel with analog feature extraction. (d) spectrogram of keyword "Bed" over different channel with digital Mel-bandpass filterbank.

the digital filter. As shown in Fig. 4 (c) and Fig. 4 (d), using the keyword example "Bed" which contains most of its energy in the high-frequency range, the analog feature extraction shows a wider response and lower resolution in the high-frequency region. However, for speech recognition applications, the proposed PoolFormer network model effectively mitigates the impact of this difference. Accuracy drops by only 0.2% compared to digital processing. Yet the analog extraction consumes just 1.02 µW, much less than digital MFCC, which also requires ADC power [15]. Although schematic-level simulation does not capture all layout parasitics or PVT variations, our results still demonstrate a strong advantage in accuracy, robustness, and ultra-low power, supporting PoolFormer's efficient architecture.

B. Quantized PoolFormer Block (QPFB)

PoolFormer, which was previously studied in the context of edge computing, has demonstrated its good compatibility with FPGA hardware, primarily owing to its straightforward linear operations and reduced need for trainable parameters [16, 17]. In this work, the PoolFormer block is enhanced to fully leverage the advantages offered by FPGA.

Since the pooling operation is parameter-free, the majority of trainable weights are concentrated in the channel MLP. To optimize FPGA resource utilization, we employ quantization for the channel MLP weights, specifically utilizing an INT8 format with a symmetric quantization approach. The symmetric quantization process, depicted in Fig. 5, involves forcing the top 2% absolute weights to align with the 98% weight boundary, while the remaining weights are scaled to the nearest levels within a range spanning from -127 to 127. This weight coefficient is then integrated into the scaling layer.

$$weight\ coefficient = \frac{weight\ boundary}{127} \quad (2)$$

$$weight\ scaling_i = half(scaling_i \times weight\ coefficient) \quad (3)$$

where i is the channel number and $half(\cdot)$ denotes conversion

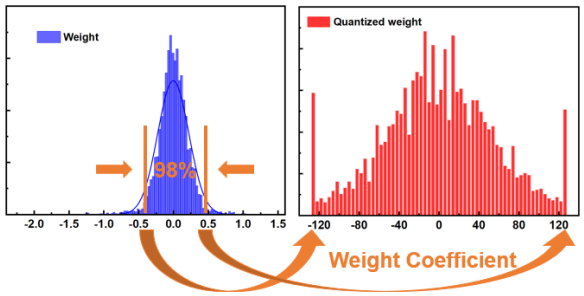


Fig. 5. The example of the proposed symmetric quantization approach.

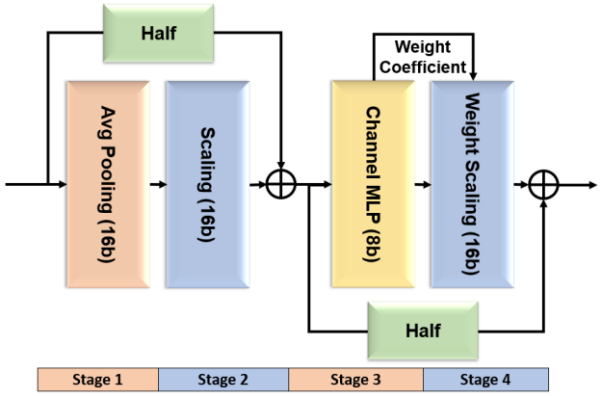


Fig. 6. The quantized PoolFormer block (QPFB) structure

of the output datatype to FP16 (16-bit floating-point), which reduces memory utilization and facilitates efficient data movement between layers in hardware. Additionally, the channel ratio has been decreased from 4 to 2 to further reduce the number of parameters within the PoolFormer block.

III. FPGA IMPLEMENTATION

A. Channel Pipeline for PoolFormer

High-Level Synthesis (HLS) is pivotal in modern electronic design, efficiently translating abstract algorithms into hardware. In this work, we simplify the process and reducing development time. In the context of neural network implementation in edge computing, HLS is crucial, enabling seamless software-hardware co-design for optimized system performance. This is especially advantageous in edge computing, where resource constraints require efficient hardware acceleration for neural networks, facilitating real-time processing at the edge.

Traditionally, each layer is processed sequentially, channel by channel, resulting in high time and memory costs. To address this, we introduce a channel pipeline that enables concurrent processing across channels and four pipeline stages (Fig. 6). During FPGA implementation, key HLS directives such as *#pragma HLS PIPELINE* and *#pragma HLS ARRAY_PARTITION* are used to further improve resource utilization and throughput.

The pooling layer operates on the padded feature map using 3×3 kernels and a stride of 1, performing average pooling. The sum of the values from the selected 9 cells is computed using an adder tree structure, and this sum is then added to the cached input after scaling multiplication. Following the caching of the output, the intermediate values are directed into the channel

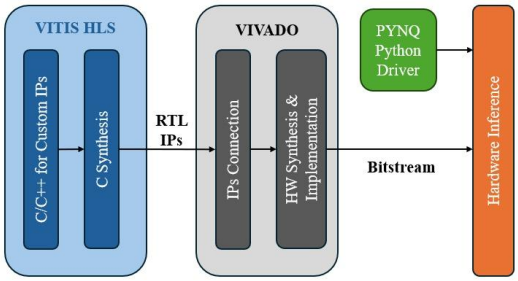


Fig. 7. The flowchart of the platform

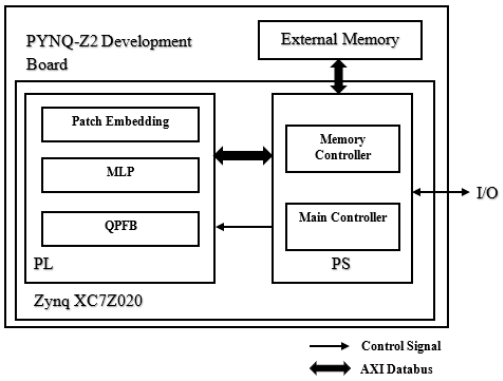


Fig. 8. The architectural configuration of the developed accelerator

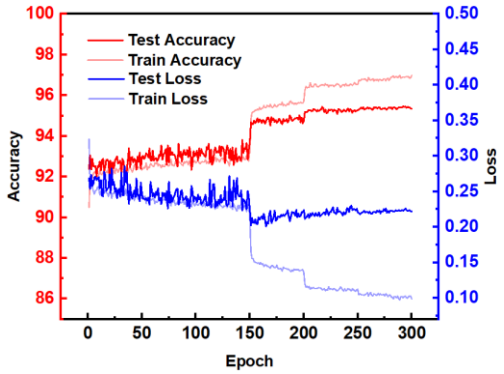
MLP layer. Given that cells in different channels are processed concurrently, the channel MLP layer is structured as two fully connected layers operating in a pipeline fashion. In the final stage, the resulting RTL file is exported as an IP core for the PoolFormer Block after completing the synthesis, mirroring the process for the patch embedding layer and MLP classifier.

B. HLS Framework

The Xilinx PYNQ platform is a good candidate for edge computing and neural network implementation since it offers both FPGA hardware and Python programming feature. Fig. 7 shows the flowchart to deploy NN accelerator on this platform. The flexibility and adaptability of PYNQ make it ideal for deployment of high performance customized neural network hardware accelerator. Additionally, it streamlines the deployment of custom hardware accelerators for neural networks, largely reducing the deployment time, while offering high performance and high energy efficiency. All these traits make PYNQ a valuable platform as edge AI devices.

In this work, we utilized the PYNQ-Z2 platform, a System-on-Chip (SoC) solution featuring both a processing system (PS) and programmable logic (PL) components. This platform enables the deployment of a high performance customized PoolFormer (PF) model. The PS is leveraged for its control logics while the PL is harnessed for neural network hardware acceleration. The architectural configuration of the developed accelerator is illustrated in Fig. 8.

Within this setup, PS executes the python driver and is responsible for all the control logics including memory allocation for both weights and activations in the dynamic random-access memory (DRAM). On the other hand, PL consists of three hardware accelerator layer modules for three different neural network layers. Notably, each layer module



Resource	Utilization	Available	Utilization%
LUT	35640	53200	66.99
FF	42528	106400	39.97
BRAM	27.5	140	19.64
DSP	156	220	70.91

Fig. 9. The resultss of the fine-tuned retraining process and resource utilization on PYNQ-Z2 board.

running on the PL predominantly comprises of multiply and accumulations (MACs) array customized to compute each layer most efficiently. Due to the constrained on-chip memory capacity and the substantial number of parameters in the current model, the parameters associated with the target model and the resulting output feature tensor for each layer are stored in external memory. Consequently, the AXI4 master data movers play a critical role in establishing a connection between the on-chip buffers and the external memory. Furthermore, AXI4 burst mode data transfers are employed to facilitate higher data throughput rates.

IV. RESULTS

Model quantization is a key step in converting the original software model to a hardware-deployable format. The 16-layer PoolFormer, initially trained in FP32, is fine-tuned and quantized symmetrically to INT8, with features represented in FP16. This optimized model is then deployed on the PYNQ-Z2 platform (ZYNQ XC7Z020 SoC). After spiking feature extraction, input data is classified by the custom PYNQ framework. System-level power and end-to-end latency measurements are not reported here, as this study primarily focuses on feasibility and core module efficiency.

The initial training phase for the PoolFormer network is conducted using FP32, achieving an accuracy rate of 95.64%. Subsequently, the model is transitioned to the proposed quantized system. Fig. 9 displays the outcomes of the fine-tuned retraining process and resource utilization. After post-implementation, our proposed edge speech recognition system attains an impressive accuracy rate of 95.41% on the 35-class Google Commands dataset [10], all while maintaining resource utilization levels that are compatible with the PYNQ-Z2 board.

In Table I, we provide a comprehensive summary of results, facilitating a detailed comparison with recent speech recognition solutions. To ensure an equitable evaluation of network performance, we focus specifically on the Google Commands classification task, aligning our analysis with findings from various contemporary works in the field. While

Table. I. Performance Comparison

	This Work	[18]	[19]	[20]	[21]
FPGA/Rpi	Zynq XC7Z020	SOMA accelerator	Intel Cyclone V	Rpi 3B+	No Hardware
Resolution (W/A)	8b/16b	8b/32b	1b/NR	32b/32b	32b/32b
class	35	12	10	10	12
process	SFE	MFCC	MFCC	MFCC	MFCC
Network	PoolFormer	TCN	CNN	RNN	BCResNet-3
Input size	32×32	40×60	1024×16	152×181	40×100
No. weight	39k	23k	433k	830k	54.2k
accuracy	95.41	93.31	90.3	96.62	97.6

metrics such as latency and throughput would offer further insight, our current comparison centers on accuracy and resource utilization due to measurement limitations and scope constraints. Notably, our research showcases a remarkable achievement in this context. Despite the relatively modest parameter count of our model, consisting of only 39k parameters, our proposed edge speech recognition system achieves outstanding results for 35-class recognition. This underscores the efficiency and effectiveness of our approach, positioning it as a highly competitive and viable solution within the realm of speech recognition technologies. Such accomplishments hold significant potential to advance the field of edge computing, particularly in scenarios where resource constraints and real-time processing are pivotal considerations.

V. Conclusion

This study presented a Python-driven HLS framework to integrate a spiking feature extraction module with a PoolFormer model for speech recognition on FPGA hardware, specifically targeting the 35-class Google Commands dataset [10]. The analog spiking feature extraction circuit consumes only 1.02 μW, and the system demonstrates a negligible accuracy drop of just 0.2% compared to its digital counterpart, achieving 95.41% accuracy after quantization. The PoolFormer, with only 39k parameters, maintains high performance while optimizing resource utilization on the PYNQ-Z2 board. This work demonstrates the viability of FPGAs for efficient edge speech recognition and highlights the advantages of HLS in accelerating the conversion from software algorithm to hardware for rapid validation. While the current system demonstrates excellent efficiency and accuracy for the 35-class Google Commands dataset, scaling to more complex tasks or substantially larger datasets may introduce challenges related to hardware resource limitations and real-time performance. Addressing these challenges will be an important focus of our future work.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[2] T. Cao et al., “A non-idealities aware software–hardware co-design framework for edge-AI deep neural network implemented on memristive crossbar,” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 12, no. 4, pp. 934-943, Dec. 2022.

[3] T. Cao, C. Liu, Y. Gao, and W. L. Goh, “Parasitic-aware modeling and neural network training scheme for energy-efficient processing-in-memory with resistive crossbar array,” *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 12, no. 2, pp. 436-444, June 2022, doi: 10.1109/JETCAS.2022.3172170.

[4] Meloni P, Deriu G, Conti F, et al. “A high-efficiency runtime reconfigurable IP for CNN acceleration on a mid-range all programmable SoC,” *2016 IEEE International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, 2016.

[5] Y. Wang et al., “AutoMap: automatic mapping of neural networks to deep learning accelerators for edge devices,” *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 42, no. 9, pp. 2994-3006, Sept. 2023

[6] M. Ahn et al., “AIX: a high performance and energy efficient inference accelerator on FPGA for a DNN-based commercial speech recognition,” *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Florence, Italy, 2019.

[7] T. Cao, W.S. Ng, W. L. Goh and Y. Gao, “DWT-PoolFormer: Discrete wavelet transform-based quantized parallel PoolFormer network implemented in FPGA for wearable ECG monitoring,” *2024 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Xi’an, China, 2024.

[8] S. Han, et al., “ESE: Efficient Speech Recognition Engine with Sparse LSTM on FPGA,” *2017 ACM/SIGDA international symposium on Field-programmable gate arrays*, 2017.

[9] A. Vaswani, et al., “Attention is all you need,” *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[10] P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” arXiv:1804.03209, 2018, [online] Available: https://arxiv.org/abs/1804.03209.

[11] M. Croce, B. Friend, F. Nesta, L. Crespi, P. Malcovati and A. Baschirotto, “A 760-nW, 180-nm CMOS fully analog voice activity detection system for domestic environment,” *IEEE J. Solid-State Circuits*, vol. 56, no. 3, pp. 778-787, Mar. 2021.

[12] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 28, no. 4, pp. 357-366, Aug. 1980, doi: 10.1109/TASSP.1980.1163420.

[13] R. G. Carvajal et al., "The flipped voltage follower: a useful cell for low-voltage low-power circuit design *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 7, pp. 1276-1291, July 2005, doi: 10.1109/TCSI.2005.851387.

[14] Z. Zhang, T. Zhang, C. Shen, W. L. Goh and Y. Gao, "A Nanowatt Temperature-Independent Tunable Active Capacitance Multiplier with DC Compensation in 0.13−μm CMOS," 2023 IEEE International Symposium on Circuits and Systems (ISCAS), Monterey, CA, USA, 2023, pp. 1-5, doi: 10.1109/ISCAS46773.2023.10181779.

[15] R. Mittal *et al*., "A 6.4-GS/s 1-GHz BW Continuous-Time Pipelined ADC With Time-Interleaved Sub-ADC-DAC Achieving 61.7-dB SNDR in 16-nm FinFET," in *IEEE Journal of Solid-State Circuits*, vol. 59, no. 4, pp. 1158-1170, April 2024, doi: 10.1109/JSSC.2023.3338686.

[16] T. Cao, W. Yu, Y. Gao, C. Liu, S. Yan and W. L. Goh, “RRAM-PoolFormer: a resistive memristor-based PoolFormer modeling and training framework for edge-AI applications,” *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, Monterey, CA, USA, 2023

[17] T. Cao *et al.*, “Edge PoolFormer: Modeling and training of PoolFormer network on RRAM crossbar for edge-AI applications,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 33, no. 2, pp. 384-394, Feb. 2025, doi: 10.1109/TVLSI.2024.3472270.

[18] J. S. P. Giraldo, V. Jain, and M. Verhelst, “Efficient execution of temporal convolutional networks for embedded keyword spotting,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 12, pp. 2220–2228, Dec. 2021, doi: 10.1109/TVLSI.2021.3120189.

[19] J. Yoon, D. Lee, N. Kim, S. -J. Lee, G. -H. Kwak and T. -H. Kim, “A real-time keyword spotting system based on an end-to-end binary convolutional neural network in FPGA,” *2023 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS)*, Tokyo, Japan, 2023.

[20] S. Yang, Z. Gong, K. Ye, Y. Wei, Z. Huang and Z. Huang, “EdgeRNN: a compact speech recognition network with spatio-temporal features for edge computing,” *IEEE Access*, vol. 8, pp. 81468-81478, 2020.

[21] B. Kim, S. Chang, J. Lee, and D. Sung, “Broadcasted Residual Learning for Efficient Keyword Spotting,” in *Proc. INTERSPEECH*, 2021.