Deep Learning-Based Signal and Image Processing Architectures for VLSI and Embedded Systems

V.Vijayasree[1*], Dr. Roja BA[2], Aniruddha Biswas[3], Dr. Anilkumar Vishwanath Brahmane[4], Ritadrik Chowdhury[5], Ranjan Banerjee[6], Dr. Nitin Jain[7]

[1*]Professor, Department of Architecture, School of Environment, Architecture and Design, SRM Institute of Science and Technology, Chennai 600089 Email ID: vijayasv@srmist.edu.in
[2]Assistant Professor, School of Computer Science and Engineering Presidency University, Karnataka ORCID ID: 0009-0002-1303-3453 Email ID: roja.b@presidencyuniversity.in
[3]Assistant Professor, Department of Information Technology, JIS College of Engineering, West Bengal, 741235 ORCID ID : 0000-0002-7089-3954 Email ID: aniruddha.biswas@jiscollege.ac.in
[4]Assistant Professor, Sanjivani College of Engineering, Kopargaon Savitribai Phule Pune University. Email ID: brahmaneanilkumarcomp@sanjivani.org.in ORCID ID: 0000-0002-6832-279X
[5]Assistant Professor, Department of Information Technology, JIS College Of Engineering, West Bengal, Kalyani, 741235, ORCID ID : 0000-0002-0927-8549, Email ID: ritadrik.chowdhury@jiscollege.ac.in
[6]Assistant Professor, Department of Computer Science & Engineering, Brainware University, Barasat, Kolkata – 125, West Bengal, 700125 India ORCID ID: 0009-0003-1950-7530 Email ID: rbkpccst@gmail.com
[7]Professor, Department of AIT-CSE, Chandigarh University, Mohali, 140413, Punjab, India. ORCID ID: 0000-0002-0561-2976, Email ID: nitin.e8466@cumail.in

Abstract
Deep learning methods have recorded impressive success in signal and image processing applications, but their use on VLSI and embedded systems has been limited by the very strict requirements of computation, memory, and power consumption. In order to solve these problems, the paper introduces a lightweight deep learning-based architecture that is specifically tailored to perform efficient signal and image processing with the constraints of embedded systems and VLSI systems. The convolutional neural network design proposed uses depthwise separable convolutions with global average pooling to minimize computational costs and memory footprint and maintain effective feature representation. The architecture is tested on an open-access image processing dataset, and its properties are examined with the help of an automated and reproducible framework of evaluation. Computational analysis shows that the proposed model has a relatively small number of multiply-accumulate operations per inference, and memory analysis indicates that the model parameters and the activations between neurons have a small footprint. The architecture is also explored on low-precision representations and significant savings on memory requirements are found that are easily applicable to embedded implementation. The scalability of performance is evaluated by a hypothetical analytical model of performance that is purely compute-based, and shows that both latency and throughput scale almost linearly with computational parallelism at moderate clock speeds. On the whole, the findings reveal that the suggested architecture provides a good compromise among computation efficiency, memory usage, and scalability and is applicable in real-time signal and image processing in VLSI and embedded systems.
Keywords: Deep Learning, Image Processing, Embedded Systems, VLSI Architecture, Hardware-Efficient CNN

1. Introduction

The most recent advancement in signal and image processing has been more reliant on deep learning due to their high efficiency in feature extraction, classification, and pattern recognition. In particular, convolutional neural networks have been demonstrated to be exceptionally helpful in a wide range of applications, including image recognition, video analytics and intelligent sensing, and encourage efficiency-aware model architecture to support more demanding computational workloads [1-2]. Nevertheless, the memory and computation needs of deep learning models are a major problem in implementing such methods on VLSI and embedded systems, where resources are limited by default by power, silicon area, and real-time processing needs.

Embedded and VLSI systems are usually forced to run with severe memory capacity, computational throughput and energy consumption constraints. Although large and intricate deep learning models can be implemented on high-performance processors and graphics processing units, these solutions are not always feasible in embedded systems because of high power usage and cost. These constraints have been explicitly called out in recent TinyML and microcontroller-focused research, which points to the necessity of compact models that can be run with small memory and energy constraints [3-4]. As a result, there has been an increasing interest in deep learning architectures that have been explicitly implemented with hardware efficiency in mind.

To address these issues, recent studies have tried to compress models, quantize them, and use lightweight architecture elements, such as depthwise separable convolutions. Neural architecture specialization methods have been suggested to produce deployment-efficient models trained on the same network [5], and quantization-aware designs have shown that it is possible to reduce the precision of arithmetic operations without causing a dramatic reduction in performance [6]. Extensive surveys also report that quantization and compression are key in facilitating effective inferences using embedded hardware [7]. In spite of these developments, much of the current literature focuses on the accuracy of algorithms but provides little information regarding hardware-based measures, including the computational cost, memory footprint, and scalability.

Here, the current paper is devoted to the design and analysis of a deep learning-based system that can be balanced in terms of representational power and hardware efficiency in signal and image processing tasks. The suggested approach focuses on the simplicity of the architecture, minimal arithmetic operations, and quantization-friendly behaviour, instead of the deepening of the model or its complexity, to enable the efficient mapping to parallel processing units in embedded and VLSI accelerators. The same motives have informed the creation of mobile-friendly vision models and online TinyML systems, which are more focused on efficiency and deployability than on architecture complexity [8- 9]. Hardware-wise, accelerator-focused works also emphasize the need to align neural network architecture with underlying compute and memory hierarchies [10] and memory-efficient inference methods indicate the effects of the architectural decision on the requirements of activation storage [11]. The objectives of this study are summarized as follows:

- To design a lightweight deep learning architecture suitable for signal and image processing under the computational and memory constraints of VLSI and embedded systems.
- To quantitatively evaluate the proposed architecture in terms of computational complexity, memory footprint, and performance scalability using an automated and reproducible analysis framework.
- To investigate the suitability of low-precision representations for reducing memory requirements while maintaining compatibility with real-time embedded inference

2. Methodology

2.1 Dataset Description

The experimental evaluation of the proposed architecture was conducted using the CIFAR-10 dataset, a widely adopted open-access benchmark for image processing research. The dataset comprises 60,000 color images distributed evenly across ten classes, with 50,000 samples allocated for training and 10,000 for testing. Each image has a fixed spatial resolution of 32 × 32 pixels with three color channels, providing a compact yet representative workload for evaluating deep learning models under embedded and VLSI constraints [12]. The balanced class distribution and small input dimensions make the dataset particularly suitable for analyzing computational efficiency and memory behavior in resource-constrained systems.

2.2 Data Preprocessing and Statistical Analysis

A statistical analysis of the input data was performed before the architecture was evaluated to characterize the input data distribution. The training set was automatically used to compute the values of channel-wise mean and standard deviation which were subsequently used to normalize the means and variances. This type of normalization ensures that the activation values between layers are always constant and enables to make effective fixed-point and quantized inference. The distribution of pixel intensities was studied to define limited dynamic ranges in all color bands which is a good property of low-precision arithmetic often used in embedded VLSI applications. The preprocessing pipeline was also automated in order to achieve reproducibility and to avoid manual tuning of the dataset.

2.3 Proposed Deep Learning Architecture

The architecture of a lightweight convolutional neural network was developed with the focus on minimizing the computational and memory cost, and at the same time, the representational capacity of the image processing tasks was not as low as possible. The architecture combines conventional convolution layers with depthwise separable convolutions to minimize the count of multiply-accumulate operations without considerably impair the feature extraction ability. The pooling operations gradually decrease the spatial resolution and the channel depth is enhanced to store the higher-level features. In order to further minimize the memory requirements of dense layers, global average pooling is used before the last classification stage. The design allows mapping efficiently onto parallel processing units and scalable hardware acceleration on VLSI and embedded systems.

2.4 Embedded-Oriented Design Considerations

The architectural design was done with a clear understanding of the embedded and VLSI systems limitations, such as low memory space, power consumption, and real time performance requirements. The model has few trainable parameters which have a small memory footprint that can be stored on-chip. Spatial down sampling of early images decreases the size of peaks of activation, which decreases on-chip buffer needs. The computational structure that is convolution-based enables the effective utilization of data-level parallelism with parallel MAC units. The limited

ranges of activation that arise following normalization also enable the architecture to be susceptible to low-precision quantization, which is required to support energy-efficient embedded inference.

2.5 Automatic Model Analysis Framework

All the computational and memory measures were calculated through an automatic model analysis framework to achieve accurate and reproducible assessment. The framework derives intermediate tensors by forward network execution and calculates layer-wise and overall measures such as multiplyaccumulate operations, floating-point operations, parameter memory and activation memory requirements. This automated method avoids manual estimation and thus the results that are reported are a direct reflection of the implemented architecture and can be replicated in other environments and model configurations.

2.6 Performance Modeling

Performance scalability was studied based on an ideal compute-only analytic model which approximates inference latency and throughput as functions of the number of parallel processing elements available and operating frequency. Latency of inference was computed by dividing the total number of MAC operations by the product of parallel MAC units and clock rate and throughput was calculated as the reciprocal of latency. Several different configurations were tested to determine scalability over a spectrum of embedded performance goals. Even though the access latency of the memory and the control overheads were not directly modeled, the analysis gives an upper bound of the performance that can be achieved and can also give good information on the trends of the architectural scalability.

2.7 Evaluation Metrics

The evaluation of the proposed architecture has been done based on a set of measures that are detailed and reflect both computational performance and hardware appropriateness. These metrics are total MACs and FLOPs per inference, parameter memory size under FP32 and INT8 representation, activation memory needs and approximate inference latency and throughput. Together these metrics give a comprehensive evaluation of the appropriateness of the architecture to be implemented in the VLSI and embedded signal and image processing systems.

3. Results

This section presents the results obtained from the evaluation of the proposed deep learning-based architecture, with emphasis on computational complexity, memory footprint, and performance scalability under embedded and VLSI-oriented constraints. All results are derived from an automated and reproducible analysis framework.

3.1 Dataset and Model Characteristics

Table 1 summarizes the dataset and model configuration used in the study. The suggested architecture was tested on the CIFAR-10 dataset, which comprises of RGB images of 32 x 32 size and ten classes. The convolutional neural network designed has a total of 31,050 trainable parameters, which makes the model size very small 121.29 KB at FP32 precision and 30.32 KB at INT8 precision. These features indicate that the proposed architecture is highly applicable in deployed on memory-limited embedded and VLSI systems.

Table 1. Dataset and Model Configuration

Parameter	Value
Dataset	CIFAR-10 (Open Access)
Input Resolution	32 × 32 × 3
Number of Classes	10
Model Type	Lightweight CNN with Depthwise Separable Convolutions
Total Parameters	31,050
Model Size (FP32)	121.29 KB
Model Size (INT8)	30.32 KB

3.2 Computational Complexity Analysis

Computational complexity of the proposed architecture was measured per multiply-accumulate operations (MACs) and floating-point operations (FLOPs) as shown in Table 2. The findings show that one inference takes 5,539,072 MACs, which is about 11,078,144 FLOPs, which is substantially less than the number of FLOPs of traditional deep convolutional networks.

Table 2. Computational Complexity Analysis

Metric	Value

MACs per Inference	5,539,072
FLOPs per Inference	11,078,144
Dominant Compute Layers	Standard Conv2D
Complexity Reduction Technique	Depthwise Separable Convolution

Table 4 and Figure 1 show the layer-wise analysis of MAC contributions in detail. The findings indicate that most of the computation cost is localized on the standard convolution layers, especially those with higher channel dimensions. On the contrary, depthwise separable convolution layers add significantly fewer MACs, which proves their usefulness as a complexity reduction method. The multiple layers at the output stage add insignificant value to the overall calculation, meaning that the structure is convolution-limited and highly compatible with parallel MAC-based hardware acceleration.

Table 4. Layer-Wise MAC Contribution

Layer Name	Output Shape	MACs
Conv2D	32 × 32 × 32	884,736
DepthwiseConv2D	32 × 32 × 32	294,912
Conv2D_1	32 × 32 × 64	2,097,152
DepthwiseConv2D_1	16 × 16 × 64	147,456
Conv2D_2	16 × 16 × 128	2,097,152
Dense	128	16,384
Dense_1	10	1,280

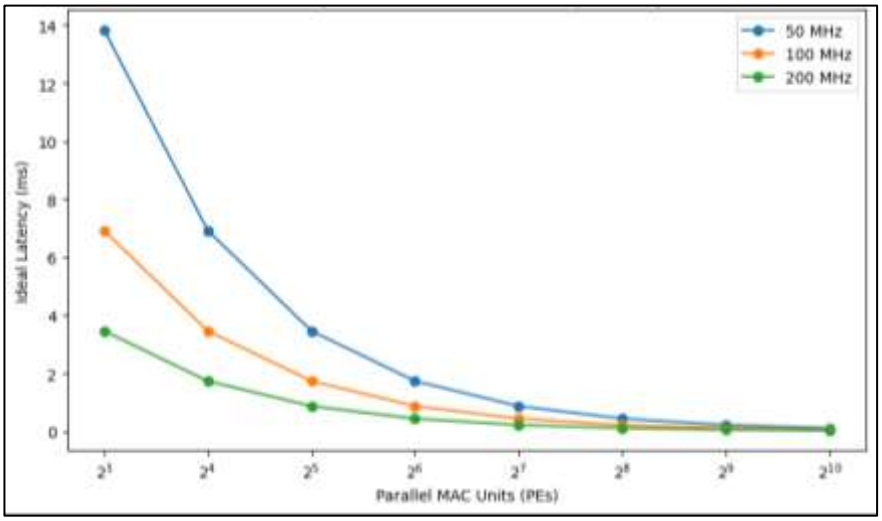


Figure 1: Ideal Inference Latency as a Function of Parallel MAC Units at Different Clock Frequencies

3.3 Memory Footprint Analysis

The memory needs of the proposed architecture were determined based on the model parameters, as well as intermediate activations, and are summarized in Table 3. The weight memory of the model measures 121.29 KB in FP32 precision and 30.32 KB in INT8 representation, which is four times less than the storage requirements.

Table 3. Memory Footprint Analysis

Memory Component	FP32	INT8
Weight Memory	121.29 KB	30.32 KB
Peak Activation Memory	256.00 KB	64.00 KB
Sum of Activations	2.157 MB	0.539 MB
Runtime Footprint (Weights + Peak Activation)	0.368 MB	0.092 MB

Figure 2 depicts the layer-wiseness distribution of activation memory and reveals that the larger-spatial-dimension convolution layers take the preponderance in the activation storage. The highest activation is at a convolution layer that generates feature maps of dimension 32 × 32 × 64, which needs 256.00 KB in FP32 precision and 64.00 KB in INT8 precision as is reported in Table 3. The total activation memory in all layers is 2.157 MB in FP32 and 0.539 MB in INT8 representation.

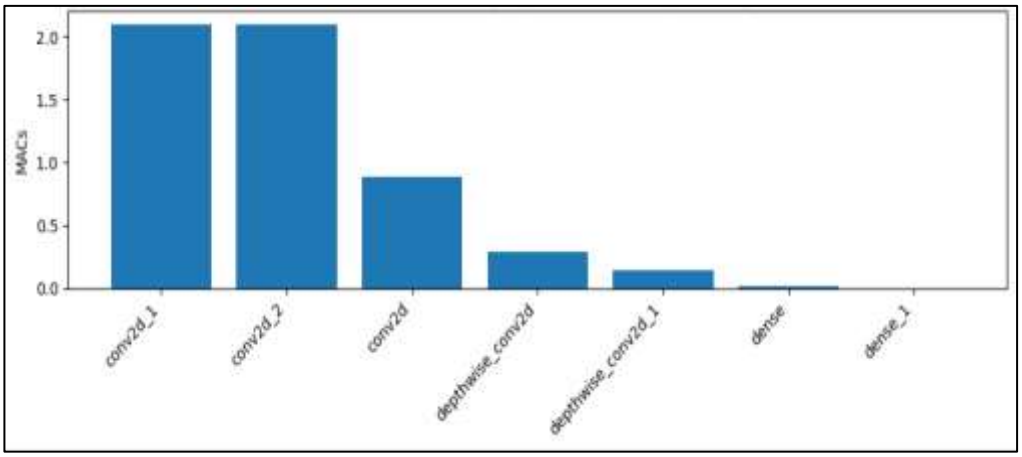


Figure 2: Layer-Wise Contribution of Multiply–Accumulate Operations (MACs) in the Proposed Architecture

In comparing weight memory with peak activation memory, it is estimated that the footprint of the runtime memory will be 0.368 MB in FP32 precision and 0.092 MB in INT8 precision. This can be well visualized in Figure 3 which shows the significant memory savings that can be attained by using low-precision representations, which is essential in embedded and VLSI implementations.

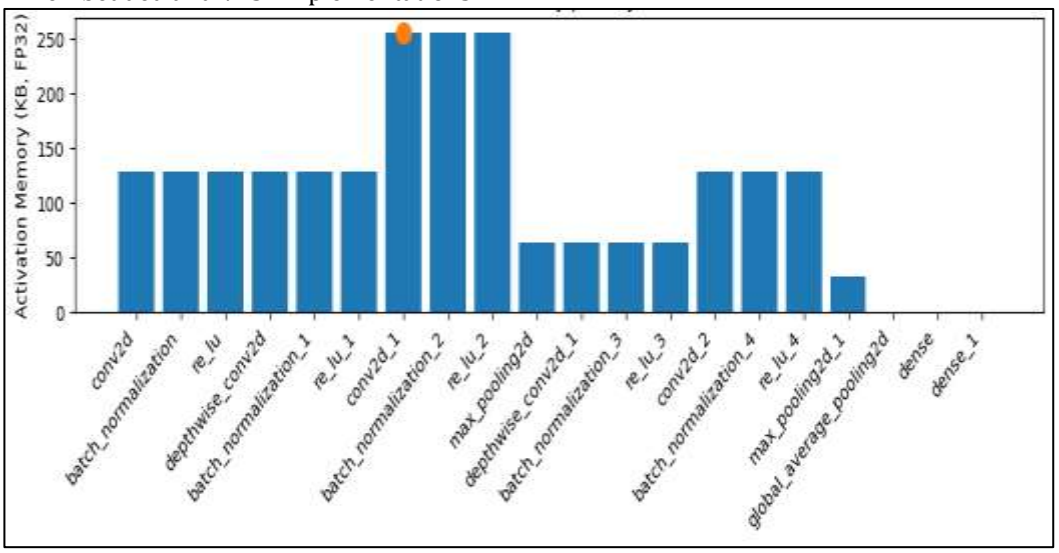


Figure 3: Layer-Wise Activation Memory Requirements of the Proposed Architecture (FP32 Precision)

3.4 Performance Scalability Analysis

The scalability of the proposed architecture in terms of performance was tested with the help of an ideal compute-only analytical model. Figure 4 shows the dependence between the inference latency and parallelism of its computation at various clock frequencies.

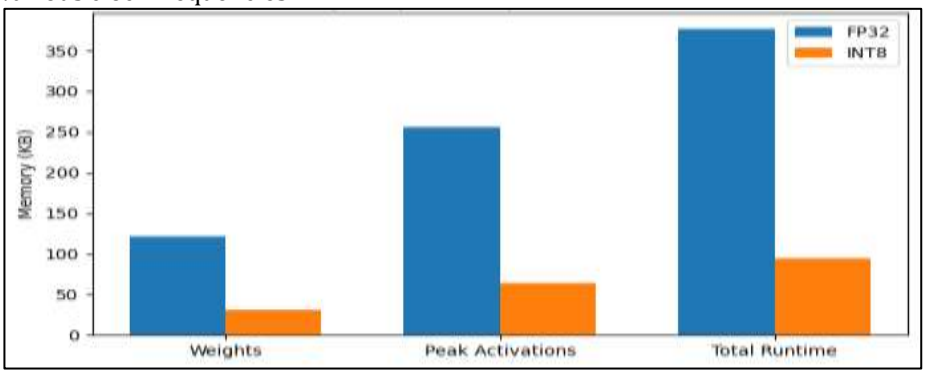


Figure 4: Comparison of Runtime Memory Footprint Under FP32 and INT8 Representations

The findings show that the inference latency reduces almost proportionately to the number of parallel MAC units, which means that the data-level parallelism is efficiently utilized. The architecture has inference latency of about 1.73 ms at a clock frequency of 50 MHz and 0.43 ms at a clock frequency of 256 parallel MAC units as illustrated in Table 5. At increased clock rates, linear decreases in latency are found, leading to major throughput gains.

Table 5. Latency and Throughput vs Parallel MAC Units

MAC Units (PEs)	Clock (MHz)	Latency (ms)	Throughput (FPS)
64	50	1.73	578
128	50	0.87	1,155
256	50	0.43	2,311
128	100	0.43	2,311
256	100	0.22	4,622
512	100	0.11	9,243
256	200	0.11	9,243
512	200	0.054	18,486

An example is, at 100 MHz and 128 parallel MAC units, the proposed architecture has a latency of less than a millisecond and a throughput of more than 2,300 frames per second. These findings validate the fact that the architecture is scalable to support real-time performance requirements in a broad spectrum of embedded system configurations.

4. Discussion

The current paper critically analyzes the architectural effectiveness and hardware appropriateness of the suggested deep learning-based model in signal and image processing under the constraint of VLSI and embedded systems. The discussion explains the observed computational, memory and scalability features with respect to the real-world hardware deployment needs, in line with the trends reported in recent mobile and embedded deep learning architectures [13].

The analysis of the computational complexity shows that the suggested architecture has a low inference cost in comparison to traditional deep convolutional neural networks. This means that the architecture is highly optimized to be hardware accelerated, as the concentration of computation in few convolutional layers suggests. Parallel MAC-based architectures have been successfully exploited in similar accelerator-oriented workloads, where workloads dominated by convolution can be successfully exploited [14]. Inclusion of depthwise separable convolutions also minimise redundant computations and maintain hierarchical feature extraction to give a desirable trade-off between expressiveness and efficiency.

The features of memory footprint also support the inherent appropriateness of the suggested architecture. The small number of parameters used makes the weight memory footprint relatively small, and can be stored in on-chip storage in most embedded systems. More to the point, the restricted peak activation memory constraints impose on-chip buffer needs and diminish the need to access off-chip memory. This is especially crucial in resource-constrained platforms, where memory access can frequently be the primary source of energy usage and system latency as noted in previous TinyML-specific research.

The discussion of low-precision representations shows the possible advantages of quantization to embedded implementation. The large decrease of weight and activation memory with INT8 representations indicates that the architecture is quantization friendly by nature. This fact is aligned with the recent extensive research on neural network quantization that highlights the importance of reduced precision arithmetic as one of the enablers of efficient inference on embedded and edge devices [15].

Performance scalability analysis gives further understanding of the architectural feasibility of hardware acceleration. The apparent near-linear scaling of performance with a larger number of computational parallelism units indicates that the architecture can easily utilize data-level parallelism and can be mapped onto parallel processing units in a flexible way. This type of scalability is necessary to be useful in a real implementation on reconfigurable and application-specific hardware accelerators, such as FPGA-based implementations [16].

It is necessary to note that the performance analysis in this paper is founded on an idealized compute-only model and does not directly consider memory access latency, control overhead, or data movement cost. Subsequently, the reported performance characteristics are an upper limit of possible throughput. However, the same level of abstraction has been extensively used in earlier research to obtain architectural understanding and inform the hardware design decisions, especially when compressed and low-precision neural networks are implemented [17].

On the whole, the discussion has established that the proposed deep learning-based architecture has a well-balanced trade-off between the computational efficiency, memory utilization, and performance scalability. The proposed approach is highly suitable to be deployed in embedded and VLSI signal and image processing systems due to the combination of lightweight architectural design, lower computational complexity, and quantization-friendly behavior, which is favorable to the wider applicability of hardware-aware deep learning architectures to next-generation embedded computing systems.

5. Conclusion

The current paper examined a deep learning architecture of signal and image processing with a particular orientation towards adaptation to VLSI and embedded system requirements. The lightweight convolutional neural

network was developed with the help of depthwise separable convolutions and global average pooling to minimize the amount of computation and memory usage and maintain the effective feature extraction capacity. The experimental analysis showed that the suggested architecture has a comparatively small number of arithmetic operations per inference and has a small memory footprint of model parameters and intermediate activations. These features allow it to be deployed efficiently on resource-constrained embedded platforms and minimize the off-chip memory usage, which is a key factor in power-efficient VLSI implementations. The analysis also demonstrated that the architecture can be naturally optimized to use low-precision representations, and a lot of memory can be saved without changing the fundamental network topology. The analysis of performance scalability showed that the proposed architecture has the capability of utilizing the computational parallelism effectively, with low inference latency and high throughput at moderate clock frequencies and realistic parallel processing. This activity points out the possibility of the architecture to be used in real time image processing applications and justifies its applicability to be implemented on parallel hardware accelerators. Though the performance assessment was performed based on an idealized compute-only model, the measured architectural trends can be used to gain valuable understanding of the trade-offs that exist between computation, memory and scalability in embedded deep learning systems. Future directions include adding memory-conscious execution models, assessing quantization inference precision, and running the proposed architecture on an FPGA or ASIC platform to further determine its practical usefulness. On the whole, the results indicate that hardware-aware deep learning models are able to cope with the signal and image processing problems in embedded and VLSI systems and support efficient, scalable, and real-time intelligent processing at the edge.

References

1. Tan, M. and Le, Q., 2019, May. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105-6114). PMLR.
2. Li, Y., Hu, J., Wen, Y., Evangelidis, G., Salahi, K., Wang, Y., Tulyakov, S. and Ren, J., 2023. Rethinking vision transformers for mobilenet size and speed. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 16889-16900).
3. Lin, J., Chen, W.M., Lin, Y., Gan, C. and Han, S., 2020. Mcunet: Tiny deep learning on iot devices. *Advances in neural information processing systems*, *33*, pp.11711-11722.
4. Banbury, C., Zhou, C., Fedorov, I., Matas, R., Thakker, U., Gope, D., Janapa Reddi, V., Mattina, M. and Whatmough, P., 2021. Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers. *Proceedings of machine learning and systems*, *3*, pp.517-532.
5. Cai, H., Gan, C., Wang, T., Zhang, Z. and Han, S., 2019. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*.
6. Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J. and Hua, X.S., 2019. Quantization networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 7308-7316).
7. Wei L, Ma Z, Yang C, Yao Q. Advances in the neural network quantization: A comprehensive review. Applied Sciences. 2024 Aug 23;14(17):7445.
8. Mehta, S. and Rastegari, M., 2021. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*.
9. Ren, H., Anicic, D. and Runkler, T.A., 2021, July. Tinyol: Tinyml with online-learning on microcontrollers. In *2021 international joint conference on neural networks (IJCNN)* (pp. 1-8). IEEE.
10. Chen, Y.H., Yang, T.J., Emer, J. and Sze, V., 2019. Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *9*(2), pp.292-308.
11. Lin, J., Chen, W.M., Cai, H., Gan, C. and Han, S., 2021. Memory-efficient patch-based inference for tiny deep learning. *Advances in Neural Information Processing Systems*, *34*, pp.2346-2358.
12. Agrawal, A.M. (2022) *CIFAR-10 dataset including train and test images*. Kaggle. Available at: https://www.kaggle.com/datasets/ayush1220/cifar10/
13. Qin, D., Leichner, C., Delakis, M., Fornoni, M., Luo, S., Yang, F., Wang, W., Banbury, C., Ye, C., Akin, B. and Aggarwal, V., 2024, September. MobileNetV4: Universal models for the mobile ecosystem. In *European Conference on Computer Vision* (pp. 78-96). Cham: Springer Nature Switzerland.
14. Liu, Z., Hao, Z., Han, K., Tang, Y. and Wang, Y., 2024. Ghostnetv3: Exploring the training strategies for compact models. *arXiv preprint arXiv:2404.11202*.
15. Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M.W. and Keutzer, K., 2022. A survey of quantization methods for efficient neural network inference. In *Low-power computer vision* (pp. 291-326). Chapman and Hall/CRC.
16. Kawakami, H., Watanabe, H., Sugiura, K. and Matsutani, H., 2023. A low-cost neural ODE with depthwise separable convolution for edge domain adaptation on fpgas. *IEICE TRANSACTIONS on Information and Systems*, *106*(7), pp.1186-1197.
17. Ngadiuba, J., Loncar, V., Pierini, M., Summers, S., Di Guglielmo, G., Duarte, J., Harris, P., Rankin, D., Jindariani, S., Liu, M. and Pedro, K., 2020. Compressing deep neural networks on FPGAs to binary and.