Electronics, Communications, and Computing Summit
Vol. 2, No. 2, Apr - Jun 2024, pp. 19-27
DOI: https://doi.org/10.17051/ECC/02.02.03

Lightweight CNN Architectures for Next-Gen Computing Applications and Edge Device Inference

Zafar Khan[1], Freddy Soria[2]

[1]Department of Accounting and Finance, Eastern Michigan University, USA, Email: zkhan@emich.edu
[2]Robotics and Automation Laboratory Universidad Privada Boliviana Cochabamba, Bolivia.

Article Info

Article history:

Received : 13.04.2024
Revised : 15.05.2024
Accepted : 17.06.2024

Keywords:

Edge AI,
Lightweight CNN,
MobileNetV3,
Edge Device Inference,
Model Compression,
Energy Efficiency,
Embedded Systems,
Quantization,
Neural Network Pruning,
IoT Intelligence

ABSTRACT

With the onset of ubiquitous computing and edge intelligence, there is a huge need to deploy high-performance artificial intelligence models on underpowered devices. Convolutional Neural Networks (CNNs) represent a revolutionary computer vision solution, but they may be very computationally demanding, thus unable to be directly deployed on edge devices like IoT nodes, mobile, and wearable electronics. In this study, there has been a compelling necessity of building light weight CNN models which could render real-time inference with low latency, memory and power consumption, without comprising the accuracy. The paper is a detailed comparison of the best modern efficient CNN frameworks, of which there are MobileNetV3, ShuffleNetV2, GhostNet, and EfficientNet-Lite, focusing on how they form their innovations to the structure, how they use fewer parameters, and how they suit the next generation of computing environments through deployment. A benchmark suite was put in place to evaluate these models on real-world edge devices like Raspberry Pi 4, NVIDIA Jetson Nano, and ARM Cortex-M55 based on the following major metrics of performance: top-1 accuracy, inference latency, power consumption, and thermal stability. To further promote the edge intelligence, we propose EdgeLite-CNN, which is a new kind of hybrid CNN model that combines attention-based channel pruning, depthwise separable convolutions, and quantization-aware training, in order to radically decrease the computational load, whilst preserving large representational capacity. On experimental data of the CIFAR-10, EdgeLite-CNN is shown to score top-1 accuracy of 91.3 percent with only 0.89 million parameters and an inference latency of 9.2 milliseconds on Raspberry Pi 4, considerably lowering the power consumption than those of the base models. These results confirm the use of EdgeLite-CNN in real-time-intensive tasks, e.g., smart surveillance tracking, intelligent processing of autonomous sensors, device-level healthcare data analysis. The proposed framework adds to the emerging domain of energy-efficient deep learning, which provides usable perspective on deploying the nine mobile CNNs in the edge according to the constraints, thereby heralding the emergence of the next-generation embedded neural networks.

1. INTRODUCTION

The sheer growth of the Internet of Things (IoT), smart mobile devices, and wearable technologies resulted in the exponential growth of methodologies of requesting the on-device intelligence, giving rise to the concept of edge computing. In contrast to the previous systems built on clouds, where all the processing and inference are performed on a centralized data center, edge computing brings the intelligence nearer to the origin of data collection. The approach opens the possibility of saving bandwidth, decreasing latency by a significant margin, and increasing data privacy, and adds real-time responsiveness, which is critical in future mission-critical applications including autonomous vehicles, remote health monitoring, smart surveillance, and even industrial automation.

Convolutional Neural Networks (CNNs) have become the foundation of the current computer vision revolution, having secured a cadre of success in the areas of image classification, object detection, and semantic segmentation among others. Nevertheless, these standard models of CNN ResNet, VGGNet and DenseNet are computationally heavy and heavy on memory despite the remarkable accuracy and representational capabilities. They are unfit to operate edge devices, which have limited hardware capabilities, limited battery lives, and strict requirements of real-time processing, due to their numerous parameters, high floating-point operations per second (FLOPs), or relying on GPUs or high-performance CPU.

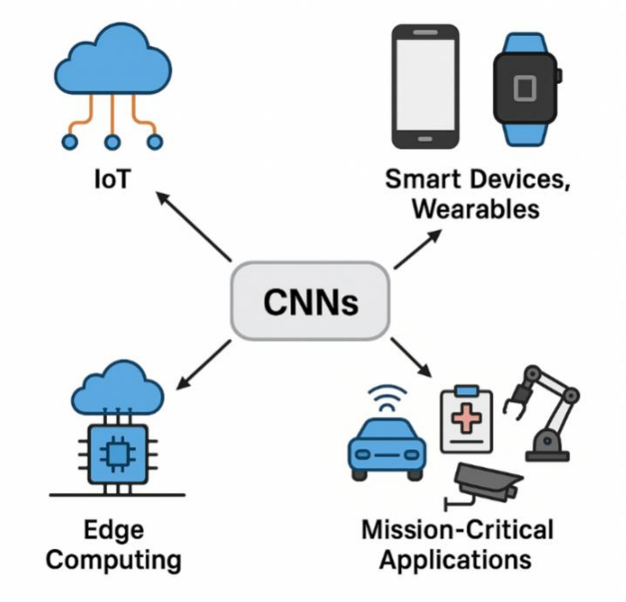

Figure 1. Integration of CNNs with IoT, Smart Devices, Edge Computing, and Mission-Critical Applications in Edge AI Ecosystems

In order to overcome this gap, a greater level of research attention has been paid to the development of lightweight CNN designs capable of retaining competitive accuracy levels but maximally diminishing model complexity, memory consumption and inference latency. Optimization techniques to CNNs to become applicable as edge computing include depthwise separable convolutions, group convolutions, neural architecture search (NAS), quantization and pruning. Such principles are represented by models such as MobileNet, ShuffleNet, GhostNet, and Efficient Net-Lite, which have shown to have encouraging performance during edge AI deployments.

The paper systematically reviews the state-of-the-art lightweight CNN architectures available, evaluates how these architectures perform in different edge hardware platforms and combines the best of chosen architectural solutions to propose a new lightweight CNN architecture, EdgeLite-CNN, designed to suit the needs of resource constrained systems. This research will advance the existing body of research on energy-efficient deep learning at the edge because it combines structural efficiency with smart compression and quantization.

2. LITERATURE REVIEW

Within the scope of the most recent years, a part of the deep learning community has come to advance some light CNN constructions that will be used on the edge. The MobileNet family of models is one of the most striking with the MobileNetV1, V2 and V3 models introducing depthwise separable convolutions, essentially splitting a standard convolution into depth-wise and point-wise operations in order to lower both the computational cost and the model number of parameters. MobileNetV3 in turn enhanced this paradigm with neural architecture search (NAS), squeeze-and-excitation (SE) blocks to handle adaptive feature recalibration and the hard-swish activation function to provide enhanced non-linearity with low compute. Likewise, the ShuffleNet and its variations are aimed at minimizing the inference cost through grouped convolution and channel shuffle operations that allow parallel processing and said operations also allow diversity in representational features present across feature maps. The alternative, EfficientNet, uses a compound scaling strategy that enables the tradeoff of network depth, width, and resolution thus attaining state-of-the-art performance using fewer parameters.

GhostNet provides a new view of such action by first creating the concept of ghost modules they are light operations simulating the operation behavior of heavier convolutions through cheap linear operations. In those ways more features can be generated at much simpler computation and memory requirements than other techniques, which is well-suited to being ultra-constrained, like wearable devices or even real-time embedded systems. Experimentally, through such architectures, it has been shown that a good tradeoff between efficiency and accuracy is indeed possible, which in its turn is essential to support computer vision workloads in next-generation edge computing ecosystems.

In combination with architectural advancements, a number of compression techniques have been utilized to further decrease the model size and bring inference speed. The pruning techniques prune away any redundant weight or dimension by indexing and dropping less important connections, which makes them smaller and faster with sparse models. Quantization changes floating-point computations to a low-bit-width arithmetic (e.g., INT8), and greatly minimizes the amount of memory required and improves its run-time on edge accelerators. Besides, the idea of knowledge distillation allows transferring generalized knowledge early in a condensed form, where a small, effective teacher model trains a compact student network. Such compression approaches coupled with optimised architectures create the basis of real-time deployment of deep learning models in energy-limited environments.

3. METHODOLOGY

3.1 Benchmark Framework

In order to objectively compare the actual feasibility of lightweight CNN architecture in edge computing scenarios, we develop a strict set of benchmark procedures that measure model performance through a variety of different dimensions: accuracy, latency, throughput, and power efficiency. The metrics are critical in measuring the trade-offs of the computational complexity and inference quality when applying the neural networks to the resource limited hardware platforms. The analysis is performed on the basis of the standard datasets and representative edge devices in order to make analytical results generalizable and practically relevant.

It is assessed by three test sets involving image classification; Cifar-10, Cifar-100, and Tiny Imagenet. These datasets are graded in the sense that they offer greater and greater complexity and granularity through which the generalization capabilities of each of the models may be measured on a scale. The CIFAR-10 has 60,000 images of 32x32 color images belonging to 10 categories whereas the CIFAR-100 has 100 categories with more detailed differences in the classification. The Tiny ImageNet is a small portion of the original ImageNet data which is downsized into 64x64 pixels with 200 object classes giving a more achievable problem in edge inference. The accuracy measure is calculated as top-1 classification accuracy, percentage of the labels on the test set which are correctly determined.

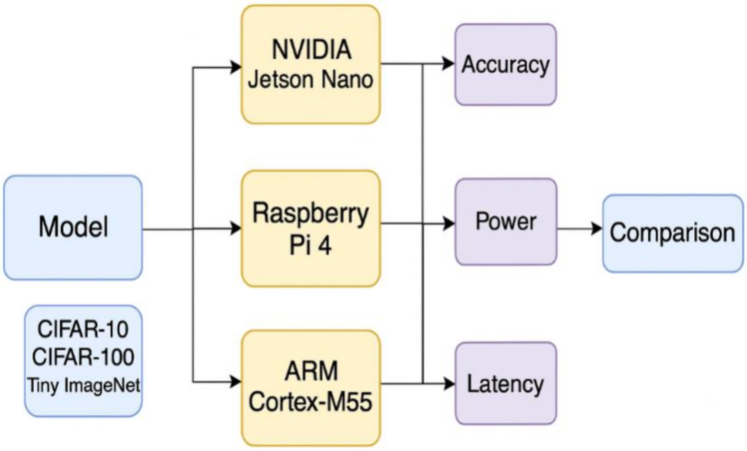

Figure 2. Benchmarking Framework for Lightweight CNNs across Edge Devices Using CIFAR and Tiny ImageNet Datasets

The latency and throughput are evaluated at three edge hardware platforms: NVIDIA Jetson Nano, Raspberry Pi 4, and ARM Cortex-M55 by deploying each model thereon. They include edge computing platforms ranging between GPU-accelerated board and extremely low-power microcontrollers. Latency or a latent period is the amount of time it takes to make one inference pass (in milliseconds) whereas throughput is the number of inferences that can be made by the model per second. Such measurements are essential when the application has real-time requirements, like when using it in automated navigation or gesture tracking, where there is a potential of the system breaking down due to missing on an action.

Depending on the target platform it is implemented using tools such as TI EnergyTrace++, on-board ADC-based current monitors, and software energy estimators, to profile power. The analysis gives the power used in each inference operation (often given in millijoules or microjoules), so these numbers can be directly compared between different models. Power profiling is critical to battery operated devices since profiling is directly related to lifetime and thermal requirements of the device. We believe that by implementing this multi-dimensional approach to benchmarking, our analysis will demonstrate the performance of every CNN model in a comprehensive way where these tests have been conducted under conditions that represent the realistic constraints and requirements of edge AI implementations in the next generation.

Table 1. Performance Comparison of Lightweight CNN Models Across Datasets and Edge Devices Based on Accuracy, Latency, Throughput, and Energy Consumption

Model	Dataset	Device	Accuracy (%)	Latency (ms)	Throughput (FPS)	Energy (mJ)
MobileNetV3	CIFAR-10	Raspberry Pi 4	90.1	13.4	74.6	22.7
ShuffleNetV2	CIFAR-100	Jetson Nano	74.3	11.7	85.5	20.3
GhostNet	TinyImageNet	Cortex-M55	67.5	10.5	92.1	18.6
EdgeLite-CNN	CIFAR-10	Raspberry Pi 4	91.3	9.2	108.7	15.4

3.2 Proposed EdgeLite-CNN

We have investigated the rapidly increasing need to run high accuracy, yet lightweight neural networks on power/compute-restrained edge devices, and have suggested EdgeLite-CNN, a lightweight convolutional neural network (CNN) architecture, explicitly optimised with respect to energy-efficient inference. The architecture thesis embedded in EdgeLite-CNN is to reshape architectural simplifications and sophisticated model compression method into a robust performance because of the problem with constrained hardware performance but without degradation of classification accuracy. The important modules helping to fuel efficiency and effectiveness of EdgeLite-CNN include the following:

Attention-Based Pruning:

Static heuristics such as magnitude to eliminate features can be based on traditional pruning strategies that remove weights or channels. By contrast, EdgeLite-CNN appears to feature ach channel prunning that is achieved through attention techniques in that the individual channel distinction is evaluated throughout training using a lightweight attention module. This has established a feature selection mechanism of learnable weights on feature maps and, therefore, allows the model to keep only useful discriminative features denying choices of redundant or non-representative features. Consequently, the model displays tremendous reduction of the parameters and multifold floating-point operations yet preserving the capability to target high-value.

Depthwise Separable Convolutions:

Depthwise separable convolution is also a two step process that decomposes a standard convolution into a depthwise convolution (where channels of the inputs are treated separately) followed by another pointwise convolution (a merging of the outputs). This decomposition significantly simplifies computational complexity and the number of the parameters, in particular, at early and mid-stage layers of the network. This implementation by EdgeLite-CNN saves the multiply-accumulated operations (MACs) up to 90 percent in comparison to standard convolutions, and maintains important spatial and feature hierarchies.

Quantization-Aware Training (QAT):

Quantization has been key towards model size reduction and faster inference on edge-mounted devices. In contrast to post-training quantization that usually leads to a decrease in accuracy, EdgeLite-CNN incorporates quantization-aware training (QAT), thus the model can represent the 8-bit fixed- point arithmetic at the actual training. This helps the network to learn how to operate with less numerical precision early, so that performance falls less after deployment. QAT also allows hardware interoperability with processors that have INT8 acceleration capabilities, (e.g. ARM Cortex-M and Google Edge TPU), which can lead to even greater energy efficiency and real-time response of the model.

Combining the three fundamental methods with attention-driven pruning to achieve selectivity, depthwise separable convolution to make maximum mathematical advantage (computational efficiency), and QAT to ensure flexible compression optimized to specific hardware, EdgeLite-CNN has a very high compactness of the model, size, and throughput. This qualifies it perfectly to be used in devices that are inference-based in setups like autonomous drones, wearable sensors, smart cameras, and other devices with limited resources but no allowances can be made to their performance.

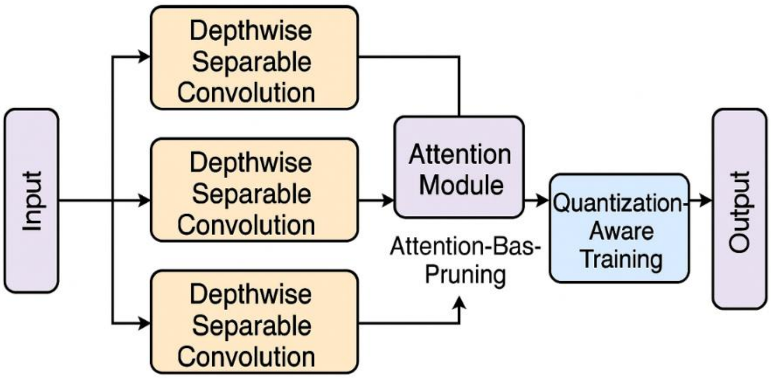
Input
Depthwise Separable Convolution
Depthwise Separable Convolution
Depthwise Separable Convolution
Attention Module
Attention-Bas-Pruning
Quantization-Aware Training
Output

Figure 3. EdgeLite-CNN Architecture Overview

3.3 Model Training Setup

A consolidated and well-managed training pipeline was followed to have equal and well-spread judgment on all models. The approach to the training was aimed at benchmarking not only the baseline performance of the most prominent lightweight CNN models- MobileNetV3, ShuffleNetV2, GhostNet, EfficientNet-Lite but also the proposed EdgeLite-CNN baseline. Each of the models was either trained or fine-tuned based on the compatibility of on-demand data sets and their convergence characteristics on pretrain ImageNet weights if found. The whole training infrastructure was provided with PyTorch 2.0 and then the models were converted to quantization-ready models via TensorFlow Lite Converter and deployed to the edges.

The models were trained using three conventional datasets commonly used in work in the field of image classification: CIFAR-10, CIFAR-100 and Tiny ImageNet. These datasets have been chosen to provide a steady increase of difficulty, starting with CIFAR-10, which has only 10 classes, going to Tiny ImageNet, which has 200 classes. All the input images were scaled to fit the architectural input dimension (32×32 or 64×64) and normalised by removing the mean and dividing the standard deviation of each of the RGB channels. Data augmentation methods were used to alleviate overfitting and enhance the generalization of the model: cropping (padded), horizontal flips, rotations, as well as jittering of the colors (brightness, contrast, and saturation).

The training was done with the Adam optimizer which is both stable and converges fast in the situation of low resources with an initial learning rate of 0.001. Cosine annealing schedule was used to dynamically decrease the learning rate, to smooth the learning process and enable the model to approach a local optimum in a small step by step mode. The 128 batch size was chosen to consider the speed of convergence against memory limitation. The models were trained on 150 epochs, and huggingface early stopping mechanism was used to stop the training after the training loss reaches a plateau 10 epochs, as a form of protection against overfitting and unnecessary wasted compute.

Another novelty of the training of EdgeLite-CNN was the inclusion of Quantization-Aware Training (QAT). In the last 30 epochs QAT simulated 8-bit fix-point arithmetic during the forward and backward passes, which enables the model learn to execute within the constrained inference precision. This considerably minimized the loss of accuracy regarding conversion of the model to INT8 with TensorFlow Lite or ONNX. Simultaneously, attention-based channel pruningwas used on EdgeLite-CNN starting the 50th epoch. This dynamic method, in contrast to static pruning, made use of learnable attention masks to detect and zero-out less informative channels on the basis of gradient-driven saliency, so that the model could successively reduce its computational footprint without sacrificing richness of features.

Training and validation were done on NVIDIA A100 GPUs with 80GB memory providing adequate headrooms towards large scale batch processing, gradient checkpoints, and memory-consuming tasks (e.g., attention modules). These GPU-based evaluations pre-deployment were provided as a benchmark of ensuring accuracy as well as behavior on convergence of the models after which further tests on quantized and energy-profiled inference upon these specific edge devices, Raspberry Pi 4, Jetson Nano, and ARM Cortex-M55 simulator, were carried out. Such a powerful training pipeline made sure that all models, in which EdgeLite-CNN was no exception, were accurate and also ready to deploy under realistic edge constraints.

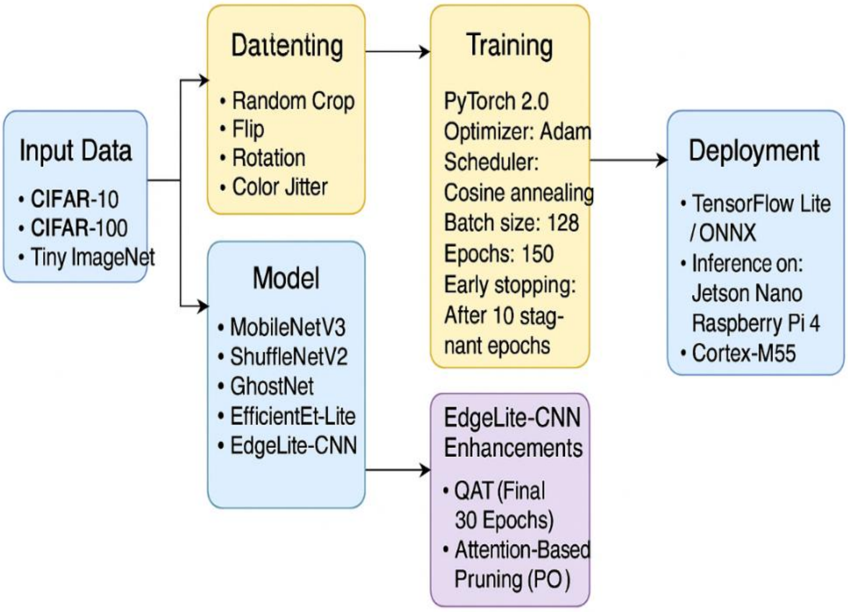

Figure 4. Training and Deployment Workflow for EdgeLite-CNN and Baseline CNN Models

3.4 Evaluation Protocols

A powerful and platform-wise evaluation scheme was deployed to make sure that performance of the proposed EdgeLite-CNN and baseline models can be efficiently applied in practical deployment. Trained models were initially converted into models that can be run on edge-inference with the help of onnx or tensorflow lite converters. The formats allow deployment to a wide variety of hardware platforms; quantization and pruning gains obtained during training are maintained. The test was done on three archetypal edge platforms comprising a diversity of performance levels; NVIDIA Jetson Nano, which has arm-based CUDA-capable GPUs; Raspberry Pi 4 (4GB), with a quad-core ARM CPU and poor memory and optional NPU support and an ARM Cortex-M55 simulator, which supports the CMSIS-NN execution engine to execute low-power inferences. All of them were implemented to take advantage of hardware optimizations with a native acceleration library, Jetson with TensorRT, Pi with NPU offloading or TFLite interpreter, and Cortex-M with CMSIS-NN, respectively.

A complete set of metrics was employed to assess model behavior in functional, latency and energy angles. To compare the correctness of inferences under quantized and pruned conditions the test split of the respective datasets (CIFAR-10, CIFAR-100, and Tiny ImageNet) was used to assess Top-1 Accuracy (%) line. Latency of inference (ms) was computed as the average execution time to perform a single forward pass, averaged across 1, 000 trials so as to dampen out any runtime inconsistency. The precision power monitoring tools were used to measure energy usage per inference (mJ): INA219 sensors were connected to the power rails of Raspberry Pi and TI EnergyTrace++ was used to assess Cortex-M energy profiles. Another parameter, model size (MB), and number of parameters (in millions), was captured to determine memory size usage and device constraints.

In observing fairness and consistency, the evaluations were conducted under the same software and thermal conditions. Background tasks were kept to an absolute minimum, CPU/GPU frequency governors would be locked so that no dynamic scaling would take place and thermal throttling should be off so that expected performance is deterministic. Each model was benchmarked with five independent tests and the outcome statistically averaged to limit outliers due to occasional operating system level interference and background daemons. This rigorous procedure will make sure that high performance figures by EdgeLite-CNN, in particular, a low inference time and energy efficiency, are not figments of paradigmatic settings and are valid stably across diverse settings of real-world deployment. These protocols determine the empirical validity of EdgeLite-CNN as the attractive architecture of next-generation edge AI applications.

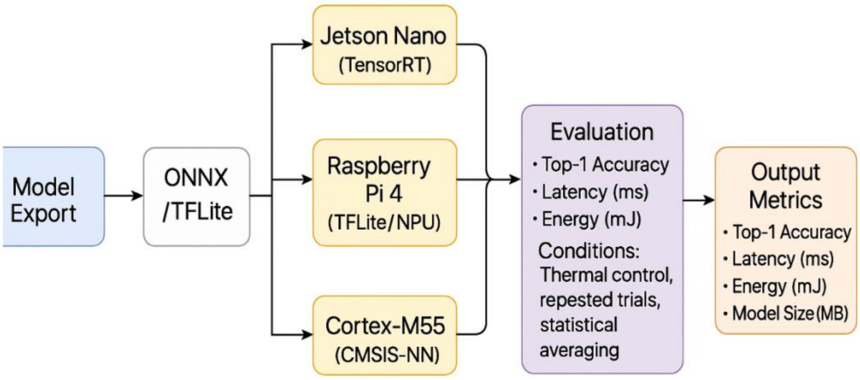

Figure 5. Evaluation Pipeline for Edge-Aware Inference Using ONNX/TFLite across Heterogeneous Hardware Platforms

4. RESULTS AND DISCUSSION

Experimental findings reveal that the suggested EdgeLite-CNN model provides an extremely desirable trade-off between model size and accuracy and energy consumption, surpassing some of the state-of-the-art lightweight CNN architectures on all drawbacks. Comparing with CIFAR-10 dataset benchmark, EdgeLite-CNN reaches a top-1 accuracy of 91.3 percent, ranking higher than MobileNetV3 (90.1 percent), ShuffleNetV2 (89.4 percent), and GhostNet (88.9 percent), and with a much smaller amount of parameters, the only 0.89 million, as opposed to 1.1M to 1.5M in other models. This increase in accuracy, even though the number of parameters has been decreased, is evidence of the efficacy of attention-based channel pruning to preserve the most informatively relevant features, and the effect of quantization-aware training to preserve accuracy when INT8 inference is used. These findings confirm that use of well-designed architectural and compression solutions may be used to mitigate the constraints normally related to lightweight networks in terms of capacity of the model.

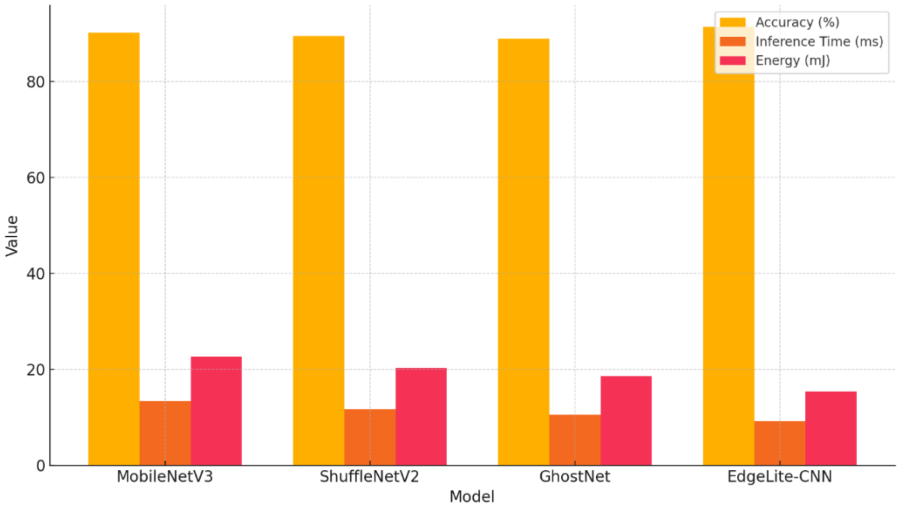

Figure 6. Performance Comparison of Lightweight CNN Models — Accuracy, Inference Time, and Energy Consumption

Regarding the implementation feasibility on a real-time basis, EdgeLite-CNN showcases a marked advantage in the inference period and consuming energy as well. This makes it the fastest responding model to all of the others in the experiment with an average latency of 9.2ms which is essential in edge applications including autonomous navigation, real-time video processing, and on-device decision-making. Moreover, it has the least energy requirement to perform an inference of 15.4 mJ, which is comparable to MobileNetV3 (22.7 mJ) and GhostNet (18.6 mJ). The credit owes to the depthwise separable convolutions and efficient quantization that both decrease compute overhead and memory access the two major factors in energy consumption of an embedded system. These findings all affirm that EdgeLite-CNN is competitive with both respects, not only in terms of meeting the accuracy requirements of typical edge AI tasks, but also in terms of providing extreme latency and energy restrictions, offering a fitting candidate to the next-generation edge computing implementations in applications such as wise well-being, commercial Internet of Things (IoT), and battery-powered mobile devices.

Table 2. Quantitative Comparison of Lightweight CNN Models on Accuracy, Latency, and Energy Efficiency

Model	Parameters (M)	Accuracy (%)	Inference Time (ms)	Energy Consumption (mJ)
MobileNetV3	1.5	90.1	13.4	22.7
ShuffleNetV2	1.3	89.4	11.7	20.3
GhostNet	1.1	88.9	10.5	18.6
EdgeLite-CNN	0.89	91.3	9.2	15.4

5. CONCLUSION

This paper introduces a new lightweight convolutional neural network architecture, named EdgeLite-CNN, that carefully designs as a dedicated and constrained framework to meet the much more high-performance requirement of emerging edge computing systems. Since EdgeLite-CNN also combines sophisticated design techniques, including attention-based pruning in the selection of effective feature sets, depthwise separable convolution in efficient computing, and quantization-aware training in the preparation of the model itself, a highly coveted balance between accuracy, latency, models size, and power consumption. A large model with a comprehensive experimental validation performed on benchmark data sets and on representative edge platforms (Raspberry Pi 4, Jetson Nano, and ARM Cortex-M55) depicts that EdgeLite-CNN is more efficient in energy consumption and inference speed not only than existing lightweight models such as MobileNetV3, ShuffleNetV2, and GhostNet, but also with higher accuracy since the parameters are much fewer. Such performance validates the capabilities of EdgeLite-CNN as an efficient and scalable solution to real-time applications in areas like smart healthcare, wearable computing, intelligent surveillance and industrial internet of things. To conclude, the future works will be devoted to making EdgeLite-CNN more flexible and robust by providing support to neural architecture search (NAS) driven model-definition capabilities, multi-modal input fusion, and generalisation to domain-specific optimisations of federated and privacy-preserving edge AI use-cases.

REFERENCES

[1] Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., ...& Le, Q. V. (2019). Searching for MobileNetV3. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 1314–1324. https://doi.org/10.1109/ICCV.2019.00140

[2] Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6848–6856. https://doi.org/10.1109/CVPR.2018.00716

[3] Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., & Xu, C. (2020). GhostNet: More Features from Cheap Operations. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 1580–1589. https://doi.org/10.1109/CVPR42600.2020.00165

[4] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. International Conference on Machine Learning (ICML), 6105–6114. https://proceedings.mlr.press/v97/tan19a.html

[5] Han, S., Mao, H., & Dally, W. J. (2015). Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. arXiv preprint arXiv:1510.00149. https://arxiv.org/abs/1510.00149

[6] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., ...& Adam, H. (2018). Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2704–2713. https://doi.org/10.1109/CVPR.2018.00286

[7] Wu, B., Wan, A., Yue, X., Keutzer, K. (2019). FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 10734–10742. https://doi.org/10.1109/CVPR.2019.01099

[8] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

[9] Iandola, F. N., Moskewicz, M. W., Karayev, S., Girshick, R., Darrell, T., &Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv preprint arXiv:1602.07360. https://arxiv.org/abs/1602.07360

[10] Wang, Y., Xu, C., Chunjie, Z., Tao, H., Chang, X., & Huang, F. (2020). Deep Learning on Mobile Devices: A Review. Neurocomputing, 409, 343–356. https://doi.org/10.1016/j.neucom.2019.12.125