2025 International Conference on Sustainable Energy Technologies and Computational Intelligence (SETCOM) | 979-8-3315-2054-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/SETCOM64758.2025.10932458

Energy Efficient Pruning and Quantization Methods for Deep Learning Models

Jay Gorvadiya
Department of ICT, PDEU
Gandhinagar, India
23mai003@sot.pdpu.ac.in

Ankur Chagela
Department of ICT, PDEU
Gandhinagar, India
ankurchangela1987@gmail.com

Mohendra Roy*
Department of ICT, PDEU
Gandhinagar, India
*Corresponding author: mohendra.roy@ieee.org

***Abstract*—As deep learning models are increasingly being deployed on resource-constrained edge devices, the need to develop techniques to make the model more energy efficient without sacrificing its performance becomes crucial. In this study we explored the three most prominent approaches, such as bit quantization, model pruning, and adaptive switching techniques to address the computational and memory overheads of deep neural networks. Bit quantization reduces the weight and activation precisions, effectively lowering the hardware requirements for both storage and computation. Again the model pruning eliminates redundant parameters; thus, reduces the model complexity, and boost the inference time. Along with all these methods the adaptive switching techniques allow the model's parameters or structure to switch at run-time so that trade-off between accuracy and energy consumption is dynamically optimized. In this paper we explored the advantages and limitations of such techniques for edge devices through a comprehensive study of recent advancements and methodologies. Point out that the open challenge on how to balance model performance with energy efficiency and future directions for the optimization of deep learning models for edge computing environments.**

***Index Terms*—Bit quantization, Model Pruning, Adaptive Switching, Edge Computing, FPGA, Raspberry Pi, Deep Learning, Energy Efficiency**

I. Introduction

Due to the rapid development of artificial intelligence and machine learning technology, excellent results have been achieved using deep Neural Network (DNN) models in various domains including target identification [1] [2] [3] [4], intelligent obstacle avoidance [5], semantic segmentation [6] [7] [8] [9], and situational awareness [10]. However, with the rapid proliferation of IoT devices and the ever-mounting demands of real-time applications, it turns out to be one of the major challenges to deploy deep learning models on resource-constrained edge devices. Traditionally, DNNs have a large number of parameters and are computationally expensive to train with ample processing power and memory requirements; sometimes it becomes infeasible in edge devices with limited energy. The deployment of deep learning models in real-time edge scenarios requires the optimization of energy efficiency. In this context, the present work focuses on three vital techniques: adaptive switching, model pruning, and bit quantization that improve the energy economy of deep learning models. Each one of the strategies presented here addresses the problem of reducing memory and computing load, considering a respectable level of model accuracy. In the sections to follow, types, uses, and trade-offs of each technique in depth are discussed.

A. Bit Quantization Techniques

In bit Quantization techniques, number of bits used to represent the weights of a deep learning model are reduced, which leads to less precision [11]. There are several methods of bit quantization. These are as follows:

- **Fixed-Point Quantization**: Weights and activations are represented using a fixed number of bits (for example, 8-bit, 4-bit) with fixed-point quantization. Since the memory footprint is dramatically reduced, quantization errors may arise and have a negative impact on the accuracy of the model.
- **Dynamic Quantization**:In dynamic quantization, the model is trained using higher precision during training stages (32-bit floating point for example) while the precision is reduced only during the inference stage. This retains the performance of the model during training while supporting faster inference
- **Quantization-Aware Training (QAT)**: In this method, during the model's training, quantization is taken into account. The mistakes caused by the quantization that emerges during model training are learned to be tolerated by the model. There will be a better post-quantization accuracy with the model trained in low-bit precision simulations. [12]
- **Post-Training Quantization (PTQ)**: The data is quantified after the model is trained. It is faster and less complicated than QAT but may result in a higher loss of accuracy. [13]

  Bit quantization uses other forms of data representations. The most common ones are INT8, FP16, and FP32, that is, accuracies that neural networks use when storing and computing quantities such as the weights and activations. Fig.1 shows the bit representations of the precision formats FP32, FP16, and INT8 widely used in deep learning. To balance between efficiency and precision, FP16 reduces the 8-bit exponent and 23-bit mantissa from FP32 to 5-bit and 10-bit, respectively. Since it only

979-8-3315-2054-0/25/$31.00 © 2025 IEEE

Authorized licensed use limited to: University of Auckland. Downloaded on March 26,2025 at 04:40:24 UTC from IEEE Xplore. Restrictions apply.

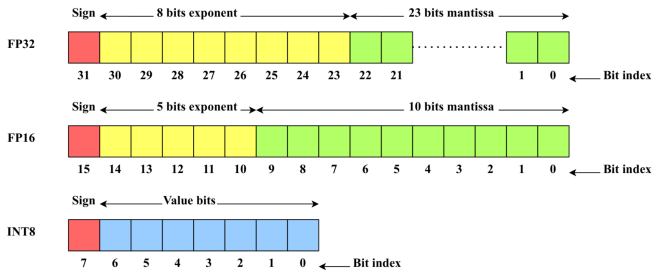

Fig. 1. Floating point and Integer representation [13]

uses bits of value, INT8 is the most resource-friendly protocol and suitable for edge devices because it occupies significantly fewer memory and energy.

- **INT8 (8-bit Integer)**:For the representation of weights and activations as 8-bit integers, INT8 dramatically decreases the memory and processing requirements of the model. Very effective for edge devices with hardware that can support 8-bit operations. [13]
- **FP16 (16-bit Floating Point)**: FP16represents the values in a 16-bit floating point format. Generally used in mixed-precision training and inference, it makes a trade-off between accuracy and efficiency. FP16 retains a reasonable accuracy but at the cost of less memory. [13]
- **FP32 (32-bit Floating Point)**: The basic format for floating point number in deep learning is known as FP32, 32-bit Floating Point. Although it is widely used for training and for inference due to its significant precision, still, due to the large memory and processing requirements, it's not the best fit for edge devices which have very limited resources. [13]

However, there are Pros and Cons of bit quantization. Such as,

- **Pros**: It requires lower processing and memory. It is more energy-efficient to infer. Hardware accelerators that support low-bit computations can take advantage of this
- **Cons**: Low accuracy, especially if doing aggressive quantization, for example 2-bit. For maintaining performance, more advanced training techniques like QAT may be required.Hardware availability is limited for very low-bit quantization.

B. Model Pruning Techniques

Model pruning reduces the size of a neural network and its computing cost by removing unnecessary or duplicated parameters. Pruning strategies try to preserve model performance by removing parameters that do not significantly contribute to the final prediction. Model pruning reduces the size of a neural network and its computing cost by eliminating unnecessary or duplicated parameters. Pruning strategies try to preserve model performance by removing parameters that do not significantly contribute to the final prediction. [14]

Unlike structured pruning, however, which removes the entire filter and leaves an even more hardware-friendly model,

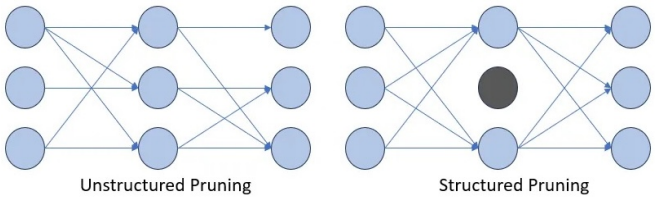

Fig. 2. Two different types of network pruning techniques are shown schematically. [15]

unstructured pruning focuses on individual neuron connections, producing irregular network architectures, as shown in Fig.2. There are several ways to perform pruning of models. These are as discussed below:

- **Structured Pruning**: This approach removes the whole model structures-channels, filters, and neurons. It produces a more compact, smaller network design that hardware can accelerate [16]
- **Unstructured Pruning**: Unstructured pruning deletes the weights based on their magnitude. Higher sparsity can be achieved with this approach, but it might not be as hardware-friendly because the resulting network tends to frequently require specialist software to handle erratic connections. [17]
- **Iterative Pruning**: The network undergoes pruning in steps and is then retrained to regain lost performance. This allows the model to be compressed progressively while maintaining the accuracy.
- **One-shot Pruning**: A model undergoes pruning as an entirety after training. While one-shot pruning is easier, it might result in a larger loss in accuracy than iterative techniques.
- **Weight Pruning**: It removes weights in the neural network that have a small contribution to the performance of the model by pruning each according to its magnitude. This leads to a sparse model due to reduced parameters.
- **Filter Pruning**: Filter Pruning in convolutional layers removes entire filters, or kernels. This reduces the computational and memory intensity, especially for CNNs, since unnecessary filters add less to the model output.
- **Layer Pruning**: This includes complete elimination of entire network layers. This is more aggressive, focusing on those layers that contribute less to improving the accuracy of a model while making the overall architecture simpler and reducing computing complexity. [18]

There are advantages and disadvantages also due to pruning of models. These are as follows:

- **Pros**: Model size and computational intensity are enormously reduced. This could be combined with other techniques, such as quantization, to further enhance efficiency. Structured pruning results in more rapid inference since it is very well-supported on hardware accelerators.
- **Cons**: Unstructured pruning is very effective at reducing parameters, but hardware acceleration may be difficult.

Authorized licensed use limited to: University of Auckland. Downloaded on March 26,2025 at 04:40:24 UTC from IEEE Xplore. Restrictions apply.

If the accuracy is to be maintained, several rounds of pruning and retraining are often required. This translates into much more computational work than some of the gains in efficiency may offset.

C. Adaptive Switching Techniques

Adaptive switching techniques switch the model's computational resources dynamically in-line with the real-time needs of the task. This is highly beneficial in edge contexts, where the computing load, energy availability, or application requirements change over time. It enables a variable trade-off between energy efficiency and accuracy. [13]. There are several techniques for adaptive switching. These are as follows:

- **Early-Exit Networks**: Many early-exit points are offered by this kind of network along the model. The system will cease its computations once an intermediate layer is able to generate a sufficiently good prediction. These reduce unwanted calculations as well as energy consumption.
- **Model Scaling**: This method allows dynamic adjustment along the available computational resources of the depth or width of the model or the resolution. For example, a model can reduce the depth of which it strikes when electricity is scarce.
- **Adaptive Inference**: This technique changes the design or precision of the model during runtime. The model could switch between full-precision and low-precision modes depending on the importance of the task or the energy budget.

The key principles of adaptive switching includes:

- **Dynamic Precision Adjustment**: Switching between INT8, FP16, and FP32 based on the complexity of input data.
- **Energy Efficiency**: Lowering power usage by processing less complex inputs with reduced precision.
- **Task-Specific Switching**: Adjusting precision based on task priority to achieve optimal power savings.

However, there are several pros and cons of adaptive switching technique. These are as follows:

- **Pros**: Suitable for real-time systems that have dynamic workloads. Highly adaptive and elastic with dynamic resource constraints. High accuracy can be achieved on major operations, while saving energy on those less important.
- **Cons**: The deferred real-time solution may introduce a latency problem. Adaptive model designs are also a challenging task where there should be accuracy versus efficiency trade-off and Overhead switching mechanism may expose some of the energy savings.

II. Methodology

A. Bit Quantization Technique

The deep learning models such as convolutional neural networks (CNN) are widely used for vision based applications. recently Wu et al. [12] have demonstrated how CNNs could be optimized for the resource-constrained context, such as edge devices using quantization of bit width. They have used INT8, FP16, using Quantization-Aware Training (QAT) accuracy is shifted during training. again post-training quantization (PTQ) was applied after training. Depending on the sensitivity, the layer-wise bit-widths were adjusted further to optimize the efficiency.

in our earlier work [13], we have demostrated the applications of bit quantisation in an energy-efficient UAV surveillance system that utilizes adaptive precision techniques for identification methodology.

The system is capable to perform simple tasks with low precision (INT8) and critical tasks like weapon detection with high precision (FP16/FP32) due to the usage of adaptive quantization. Since the model is capable of real-time object detection, YOLOv7 is taken as the base model.

With adaptive quantization, the **energy consumption** can be drastically **reduced** while at the same time maintaining good performance in vital surveillance tasks by making a wise model such as YOLOv7 combine.

In the context of bit quantization, DoReFa-Net [19] offers a comprehensive framework for efficiently training and deploying low-precision models. The ability to quantize weights and activations down to as low as 1-bit, with only a marginal accuracy drop, enables substantial savings in both energy and computational costs. The study shows that 1-bit quantization, when applied to CNNs, can yield energy savings of up to 40-60% on specialized hardware [19], while still maintaining a practical level of performance for tasks such as image recognition and object detection [12]. This aligns well with similar techniques, such as INT8 quantization [20], where models operating at lower precision demonstrate substantial energy efficiency without compromising accuracy on edge devices. Here they emplyed a new method that trains CNNs by using low-bitwidth for weights, activations, and gradients to increase energy efficiency and decrease computation.

For reducing memory and power consumption in training as well as in inference, DoReFa-Net basically quantizes weights using 1-bit, activations using 2-bit, and gradients during backpropagation. In addition to this, the method will also apply either binary or ternary weight quantization if only full-precision resources are required for conducting successful training.

For image classification tasks, the model's accuracy is similar to that of full-precision CNNs, except at a decrease in accuracy only by a small margin; it also promises energy savings with 1-bit weights and 2-bit activations of 40–60%.

DoReFa-Net thus presents a feasible approach towards edge AI applications by using low-bitwidth quantization to train and deploy energy-efficient CNNs on low-power devices.

B. Model Pruning Techniques

Molchanov et al. [21] have demonstrated variational dropout, an unstructured pruning method for deep neural networks through dropout during training. The model's size is reduced drastically, as unnecessary weights pruned in iterative way. Again, Han et al. [22] present a novel optimizing strategy

Authorized licensed use limited to: University of Auckland. Downloaded on March 26,2025 at 04:40:24 UTC from IEEE Xplore. Restrictions apply.

for neural networks through simultaneous learning of weights and connections. Here two steps are required, the first is the **Weight Learning**. In weight learning the training is performed in a conventional way of training Neural Network. The second method is the**Connection Learning**. Here the sensitivity of the model is analyzed. Here, the links are ranked according to their importance. The nature of sensitivity in each connection is determined based on how much it influences the output error. Iterative pruning reduces complexity while giving the network an opportunity to learn and retain performance.

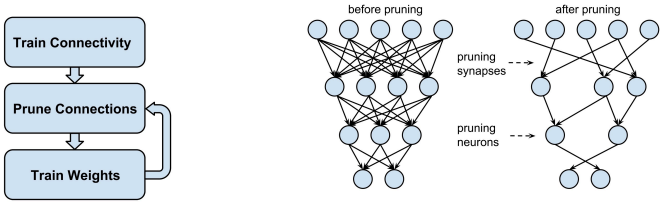

Fig. 3. A) Three-step pruning pipeline. B) Comparison of synapses and neurons before and after pruning. [22]

Fig.3 depicts a three-stage pruning pipeline and compares synapses and neurons before and after pruning, in order to make it intuitively clear that pruning reduces number of connections and neurons.

In conclusion, the optimization of weights and connections enhances neural network speed with the prospect of making them faster and larger for deployment on resource-constrained devices. In the end, the authors recommended further research work in adaptive pruning methods and their potential effects in the future.

C. Adaptive Switching Techniques

Our previous work [13] we present an energy-efficient system, proposed for UAV-based surveillance system that would maximize the UAV system's energy efficiency. The technology employs an adaptive switching technique using dynamic switching of the states of UAV components as responsive to environmental conditions and surveilled real-time needs. With this technology, there is reduced energy consumption yet ensured functionality through alternation of active, idle, and sleep power modes.

The study also concludes that the adaptive switching techniques must be utilized to develop a more energy-efficient UAV surveillance system.

The Fig.4 presents a UAV surveillance system that uses adaptive precision switching to conserve energy. The system operates in low-precision **INT8** mode for routine tasks, switching to higher precision (**FP16**, **FP32**) when needed for detecting suspicious objects. This approach reduces GPU usage by 22% and memory consumption by 29%, optimizing power efficiency while maintaining accuracy for critical tasks. The system maintained 93% mAP while achieving 2.0x Energy Savings through dynamic switching between INT8 and FP16 precision.

In this context HeatViT [23] can be utilized for 8-bit quantization along with token reduction, thus allowing for highly efficient inference in the case of Vision Transformer models on FPGAs. In comparison to static CNN-based models, HeatViT improves by 4.7x in terms of Energy Efficiency [23]. The core part that serves as the basis for this method is the polynomial approximation of non-linear functions, especially designed for FPGA implementations. Again, APPQ-CNN focuses on dynamic adjustment of quantization with pruning levels in CNNs in inference. APPQ-CNN applies adaptive precision switching, saving much energy; it reduced the power by **45%** with no significant sacrifice in accuracy on FPGA and Raspberry Pi platforms [24].

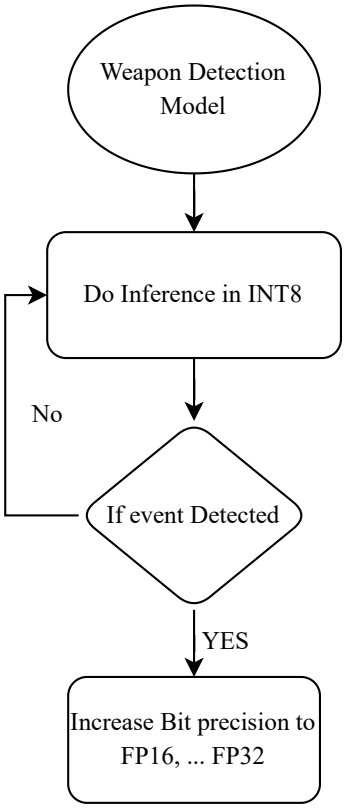

Fig. 4. Proposed Adaptive Switching Architecture [13]

III. Result

In [21], It is noticeable that INT8 quantization can reduce memory approximately up-to 75% while accuracy is reduced only to a few percent (1-2%). QAT is superior to PTQ in terms of lowest accuracy loss, as PTQ reduces the precision but maintains a high accuracy level.

The reason Int8 quantization work well in practice, especially when combined with QAT because it is effective for saving memory and computations. Layer-wise quantization can be useful in achieving the balance of efficiency and accuracy.

However, QAT and INT8 quantization together is the best combination to deploy CNNs in edge devices with least accuracy loss and consume less energy. The table I compares the

Authorized licensed use limited to: University of Auckland. Downloaded on March 26,2025 at 04:40:24 UTC from IEEE Xplore. Restrictions apply.

various techniques of quantization and its impact on accuracy and memory requirements. From this comparision we can see that QAT has very low drop in accuracy compared to PTQ, and cuts memory use up to 87.5% in 4 bit quantitation.

TABLE I
PERFORMANCE IMPACT OF QUANTIZATION ON CNN ACCURACY AND MEMORY USAGE

Model	Bit-width	Accuracy Drop (%)	Memory Reduction (%)
Full Precision	32-bit	0	0
Quantized (Post-training)	8-bit	1.5	75
Quantized (QAT)	**8-bit**	**0.8**	**75**
Quantized (QAT)	4-bit	3.0	87.5

in our earlier work [13],This technique saves **29% RAM** and **22%** GPU usage with INT8. Accuracy Achieved by Yolov7 is 93% for weapon detection at 65 frames per second.

The adaptive precision switching makes the system very energy-efficient for long-duration UAV operations by balancing accuracy with energy efficiency.

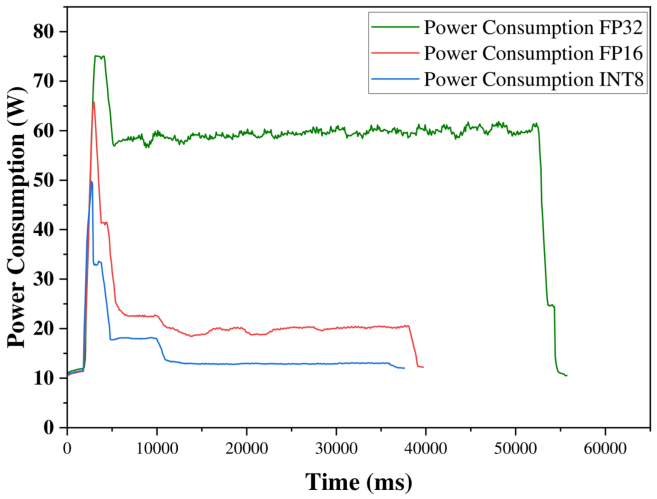

Fig. 5. Impact of Switching between INT8, FP16, and FP32 on Energy Consumption during UAV Operations for a single object detection. [13]

Fig.5 Compares the power consumption of the format types FP32, FP16 and INT8. INT8 is the most power-intensive format and therefore the best choice when edge devices are power-bound.

In [19], DoReFa-Net is very efficient for edge devices and low-power hardware since 1-bit and 2-bit quantization drastically lowers computation and memory utilization. It is useful for deploying on hardware that has historically had trouble supporting full-precision deep learning models.

In [21], It improves the network efficiency and suitability for edge devices with minimal loss in accuracy, it reduces parameters up to 80–90% while retaining the same accuracy.

In [22], The proposed strategy achieves significant reductions in model size and computational costs without loss of accuracy. The authors explain how their method can lead to the following: On popular benchmarks such as MNIST and CIFAR-10, the model size could be cut by up to 49% with less than 10% reduction in the original accuracy. The inference time, which makes the models more applicable for running on edge devices.

TABLE II
COMPARISON OF PRUNING TECHNIQUES

Pruning Method	Accuracy Impact	Power Savings	Complexity Reduction
Weight Pruning	Minimal (1-2%)	35%	Moderate
Filter Pruning	Moderate (3-5%)	45-50%	Significant
Layer Pruning	Variable	High	High

TableII Comparison of different prunning techniques. Pruning saves much more power and reduces complexity than filtering and layer pruning, while weight pruning impacts accuracy much less.

The paper [13] presents, Adaptive switching hugely boosts the UAV's operating efficiency. The authors demonstrated that it boosts a significantly longer battery life and flight time, thus staying airborne for much longer periods without needing to recharge from a power source as frequently, compared to the constant-power methods of traditional switching techniques. The energy consumption is reduced by up to **30%** compared to the constant-power methods of traditional switching techniques.

TABLE III
COMPARISON OF LATEST TECHNIQUES (2023)

Technique	Energy-Efficiency Gain	Accuracy Impact	Target Device
HeatViT	4.7x	+8.9%	FPGA (ViTs)
APPQ-CNN	45%	Minimal	FPGA / Raspberry Pi (CNNs)
E-UPQ	3x-4x	Minimal	FPGA (CNNs)

Table III shows that APPQ-CNN has poor accuracy on most devices, while HeatViT cuts energy consumption by 4.7x.

IV. CONCLUSION

The study concludes that bit quantization, model pruning, and adaptive switching strategies provide unique benefits for the improvement of the energy efficiency of deep learning models running on edge devices. Since bit quantization can dramatically reduce memory usage and computational requirements, it is outstandingly effective in resource-constrained contexts. This can actually achieve excellent efficiency with minimal loss in accuracy by using low-precision operations such as INT8 but may require careful retraining to avoid any performance degradation.

For hardware-constrained devices, such as FPGAs and Raspberry Pi, model pruning is of prime importance for reducing the size of models and computation overhead, especially

Authorized licensed use limited to: University of Auckland. Downloaded on March 26,2025 at 04:40:24 UTC from IEEE Xplore. Restrictions apply.

structured pruning. This essentially removes unnecessary parameters, which systematically ensures effective deployment while maintaining the model's integrity.

The advantage of adaptive switching algorithms is that they allow for a flexible trade-off between accuracy and power savings, making them unique in dynamic, real-time scenarios. These techniques are particularly appropriate in situations where energy constraints change since they dynamically modify computation based on operational complexity and energy availability.

In the end, the specific application and hardware constraints dictate the approach to be taken. However, the integration of these approaches, such as quantization and pruning, can further enhance energy efficiency without performance degradation. Deep learning models can be optimized to perform at their best in energy-constrained environments by tailoring these approaches to the application, thus paving the way for more efficient and sustainable AI deployment on edge devices.

REFERENCES

[1] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong *et al.*, "Swin transformer v2: Scaling up capacity and resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, June 2022, june 18–24.

[2] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Montreal, BC, Canada, Oct. 2021, oct. 11–17.

[3] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "DINO: DETR with improved denoising anchor boxes for end-to-end object detection," *arXiv*, 2023, arXiv:2203.03605.

[4] Z. Zong, G. Song, and Y. Liu, "DETRs with collaborative hybrid assignments training," in *Proceedings of the International Conference on Computer Vision (ICCV)*, Paris, France, Oct. 2023, pp. 6725–6735, oct. 2–6.

[5] X. S. Zhou and W. L. Wu, "Unmanned system swarm intelligence and its research progresses," *Microelectronics and Computer*, vol. 38, pp. 1–7, 2021.

[6] Z. Chen, Y. Duan, W. Wang, J. He, T. Lu, J. Dai, and Y. Qiao, "Vision transformer adapter for dense predictions," *arXiv*, 2023, arXiv:2205.08534.

[7] Y. Fang, W. Wang, B. Xie, Q. Sun, L. Wu, X. Wang, T. Huang, X. Wang, and Y. Cao, "EVA: Exploring the limits of masked visual representation learning at scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, June 2023, pp. 19 358–19 369, june 17–24.

[8] W. Su, X. Zhu, C. Tao, L. Lu, B. Li, G. Huang, Y. Qiao, X. Wang, J. Zhou, and J. Dai, "Towards all-in-one pre-training via maximizing multi-modal mutual information," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, June 2023, pp. 15 888–15 899, june 17–24.

[9] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li *et al.*, "Internimage: Exploring large-scale vision foundation models with deformable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, June 2023, pp. 14 408–14 419, june 17–24.

[10] L. Tang, Z. Ma, S. Li, and Z. X. Wang, "The present situation and developing trends of space-based intelligent computing technology," *Microelectronics and Computer*, vol. 39, pp. 1–8, 2022.

[11] H. Yu, T. Wen, G. Cheng, J. Sun, Q. Han, and J. Shi, "Low-bit quantization needs good distribution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 680–681.

[12] J. Wu and J. Liu, "Quantization in convolutional neural networks," *IEEE Transactions on Neural Networks*, vol. 27, no. 10, pp. 2160–2172, 2016.

[13] N. Bhuva, V. Devre, P. Sharma, D. Pujara, and M. Roy, "An energy efficient system for unmanned aerial vehicle-based surveillance," in *Proceedings of ICDSIS*, 2023.

[14] Y. He and L. Xiao, "Structured pruning for deep convolutional neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[15] A. Goyal. (2018, September) Pruning neural networks. Accessed: 2025-01-06. [Online]. Available: https://medium.com/@goyal.ayush55/pruning-neural-networks-1f3a3be79c7d

[16] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[17] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 1135–1143.

[18] F. Meng, H. Cheng, K. Li, H. Luo, X. Guo, G. Lu, and X. Sun, "Pruning filter in filter," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 17 629–17 640.

[19] S. Zhou *et al.*, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv*, 2016, arXiv:1606.06160.

[20] Y. Zhang, B. Sun, W. Jiang, Y. Ha, M. Hu, and W. Zhao, "Wsq-AdderNet: Efficient weight standardization based quantized addernet fpga accelerator design with high-density INT8 DSP-LUT co-packing optimization," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022.

[21] P. Molchanov *et al.*, "Variational dropout sparsifies deep neural networks," *International Conference on Learning Representations (ICLR)*, 2017.

[22] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural networks," in *Proceedings of NeurIPS*, 2015.

[23] X. Li *et al.*, "HeatViT: Hardware-efficient adaptive token pruning for ViTs," *IEEE Journal on Emerging Topics in Circuits and Systems*, 2023.

[24] H. Kim *et al.*, "Adaptive CNN inference accelerator for edge devices," *IEEE Embedded Systems*, vol. 15, no. 2, pp. 55–63, 2023.

Authorized licensed use limited to: University of Auckland. Downloaded on March 26,2025 at 04:40:24 UTC from IEEE Xplore. Restrictions apply.