(IJ-02) A Systematic Literature Review on Energy- Efficient Deep Learning Models on Edge Devices

Dr. Shams Al Ajrawi

Assistant professor at Alliant International University

Mounica Gopisetty

Department of Electrical Engineering San Diego State University

Dr. Aaron Wester

MSDA Program Director, CSML, Alliant International University

ABSTRACT

Energy efficiency is vital for edge devices due to the complex computations required by deep learning. The rapid adoption of intelligent systems has significantly increased energy demand, impacting resource constraints, battery life, and environmental sustainability. With the exponential growth of edge devices across various IoT sectors, optimizing hardware and models is essential to maintain accuracy without compromising energy efficiency. The demand for deep learning has surged, leading to a 300,000-fold increase in computational requirements over six years. The number of IoT devices is projected to reach 29.0 billion by 2027, up from 18.8 billion in 2024.

Index Terms— Deep Learning Models, Energy Efficiency, Edge Devices, IoT Devices, Model Compression,

INTRODUCTION

Energy efficiency is vital for edge devices due to the complex computations required by deep learning. The rapid adoption of intelligent systems has significantly increased energy demand, impacting resource constraints, battery life, and environmental sustainability. With the exponential growth of edge devices across various IoT sectors, optimizing hardware and models is essential to maintain accuracy without compromising energy efficiency. [1] The demand for deep learning has surged, leading to a 300,000-fold increase in computational requirements over six years. The number of IoT devices is projected to reach 29.0 billion by 2027, up from 18.8 billion in 2024. [2] This paper will explore the advancements in energy-efficient deep learning for edge devices and aspects of current applications and techniques of edge computing. [3]

This literature review aims to delve into the emerging advancements of energy-efficient deep learning and edge computing while also researching current techniques, applications, and challenges. Over the past few years, significant research has advanced regarding deep learning techniques and methods focused on fostering sustainability while reducing energy consumption. This paper will begin with an overview of the use of deep learning in edge computing and relevant applications like object and anomaly detection, which use real-time processing. [4] Then, in Section 3, we will explore energy-efficient deep-learning techniques, including model compression methods such as pruning, quantization, and knowledge distillation, and then move to lightweight neural network architectures and their design concerning efficiency and accuracy. Specifically, we will perform a comparative analysis of MobileNet, SqueezeNet, and EfficientNet and the tradeoffs between the accuracy and efficiency of early exit architectures and dynamic neural networks. For EfficientNet, we will evaluate the optimization of models through a neural architecture search. [3] In Section 4, we will examine multiple algorithmic optimization techniques like sparse

representation, low-rank factorization, federated learning, and attention mechanisms. [5] In Section 5, we will move to hardware optimization by exploring specialized edge-specific hardware and tailoring models for GPUs. A major aspect of the hardware is embedded batteries and techniques to improve energy consumption to extend operation seamlessly. [6] In Section 6, this paper will critically analyze the applications through multiple large sectors and the IoT market as well as the challenges between energy efficiency, accuracy, and latency. This review will continue to summarize emerging trends and future models that address the energy efficiency challenges in edge computing. [7]

BACKGROUND

Energy efficiency is crucial for the technological, environmental, and economic viability of AI implementation across various sectors. To address resource constraints caused by intensive computing, such as limited power supply from embedded batteries or energy harvesting systems in edge devices, as well as hardware limitations in computational power and memory, we can implement energy-efficient algorithms. [2] These energy-efficient models can extend operation without interruptions in real-time processing or recharging due to the increased energy consumption of deep learning algorithms. This will allow for the scalability of the IoT market and reduce the environmental impact by decreasing greenhouse gas emissions, lowering operational costs, and battery degradation. Energy-efficient models will maintain continuous operation for reliability purposes, which is essential in multiple sectors of edge devices. [8] The widescale deployment is relevant in sectors like health technology, autonomous vehicles, smart cities, space and defense, and renewable energy systems. Figure 1 below describe this flow of general principles of Green Edge AI. [3]

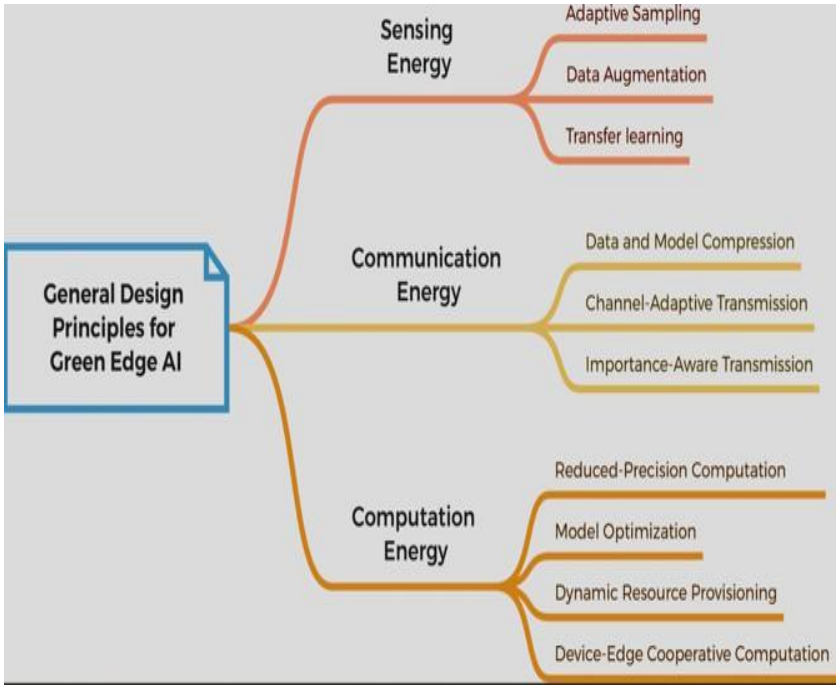

Fig. 1: General Principles of Green Edge AI

ENERGY-EFFICIENT DEEP LEARNING TECHNIQUES

Enhancing the energy efficiency of deep learning in edge computing relies heavily on algorithmic techniques. Deep learning demands substantial computational power and training, making the deployment of model compression methods and lightweight neural network architectures crucial for reducing energy consumption and storage requirements. A main facet of deep learning is object detection by localizing the objects, classifying objects, and identifying objects in image processing. [9] The implementation of deep learning has vastly improved image and speech processing for IoT devices by providing high-accuracy models and low latency. This AI technique is integral in a multitude of fields and researching efficiency is essential for future contributions in these fields and for discovering new advancements. The types of images using deep learning in edge devices include remote sensing, SAR, aerial imagery, and underwater imagery, which are small objects with device constraints and low detection accuracy. [10] Then we see digital images, X-rays, and MRI scans which have higher accuracy models and image classification and object detection. We

also look at istopathological and microscopic images that rely on the use of Generative adversarial networks (GANs) and preprocessing overhead. The relevant domains for classification using deep learning in edge computing include health technology and medicine, autonomous vehicles and transportation, space and defense, agriculture, and smart city development. [4]

Model Compression Methods

Model compression techniques create more compact networks by reducing the size and computational load of deep learning models without sacrificing accuracy. Common approaches to enhancing efficiency include pruning, quantization, knowledge distillation, low-rank factorization, neural architecture search (NAS), and adaptive inference. [10]

1. Pruning

Pruning a deep neural network (DNN) involves removing the redundant connection of neurons in a neural network to improve network performance while maintaining the same standard of network accuracy. There are multiple types of pruning, such as structured pruning, channel pruning, and unstructured pruning. Structured pruning eliminates entire filters from multiple layers to reduce model size. Structured pruning can remove entire units of connections, including channels, filters, or neurons, which allows for hardware efficiency. [11] Chanel pruning is a specific type of structured pruning where entire corresponding channels are removed and new advancements in convolutional neural networks propose acceleration methods so the removal of filters will not have a higher influence. Unstructured pruning eliminated individual weights based on magnitude, allowing a higher compression with sparse matrix operations. [9] There is a 90% weight reduction for pruned ResNet models for image classification with negligible loss of accuracy as seen in Figure 2. [7] Larger, highly compressed models can require retraining to recover accuracy. This technique reduces the number of operations and significantly lessens energy consumption during inference. Regarding efficiency, pruning is better for larger models with redundant

parameters, and its latency is improved on edge devices that support spare computation. [11]

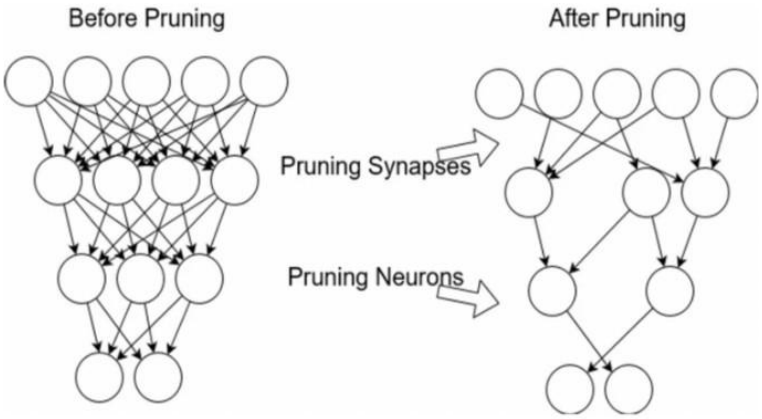

Fig. 1: Example of Pruning Process

2. Quantization

Quantization is a model compression method to reduce the size by storing weights in different formats. This lowers the precision of the weights from 32-bit floating-point (FP32) to lower-bit formats, such as 16-bit (FP16) or 8-bit integers (INT8). [10] There are two types of quantization: post-training quantization (PQT) and quantization-aware training (QAT). Post- training quantization will quantize the model after training, while quantization-aware training has higher accuracy retention by integrating quantization during the training process. This model compression method will reduce bandwidth requirements, and computational costs, and increase efficiency because integer operations consume less power than floating-point operations. In both post- training Quantization and Quantization-Aware Training there is up to a 4-time reduction in model size. For efficient quantization in edge devices, we can provide a framework through TensorFlow Lite or PyTorch Mobile for real-time image classification of mobile devices.

[12] The limitations of quantization include a risk of accuracy loss for models that have a high sensitivity to numerical precision, and this method requires hardware that supports

integer computations such as Tensor Processing Units and Edge TPUs. Tensor Processing Units are developed for large-scale matrix computations and Edge TPUs are utilized for edge devices with a limited power consumption. This method is more effective for edge devices with integer computation abilities and latency is reduced on the compatible hardware. [10]

3. Knowledge Distillation

Knowledge distillation is an efficient model compression technique for training neural networks. This method trains a smaller "student" model to replicate and learn the predictions of a larger, thus more accurate, "teacher" model. This process is fully automated compared to quantization and pruning. This model has a lot of energy advantages since the smaller models will be computationally inexpensive compared to the larger, complex networks reducing energy consumption during inference. The "student" model will be faster and smaller while continuing to maintain the accuracy of its "teacher model" without requiring the use of additional specialized hardware. [13] There are many current advancements in knowledge distillation including collaborative learning where multiple "students" can simultaneously learn through the use of "teacher-assistant" models. This method can incur additional computational costs during training based on the depth of the "teacher-student" setup. The most important factor for this compression method is the quality of the "teacher" model since this will heavily affect the performance of the student model. [14] An application of knowledge distillation is deploying compact Natural Language Processing (NLP) like DistilBERT for in edge devices by placing the NLP in a live environment for real application of mobile text processing. The most efficient use of knowledge distillation is when the "teacher" model is readily available with depth and latency is minimal because the "student" model is compressed. [13]

4. Neural Architecture Search (NAS)

Neural Architecture Search is a machine learning technique that automatically uses

algorithms with the ideal combination of layers, filters, and connections to identify the optimal model architecture based on set parameters. This process is fully automated to create specifically designed architectures for energy efficiency operation on edge devices. The NAS models are extremely accurate and are optimized for memory usage and computational output compared to human- designed models. Examples of NAS we will explore are MobileNet, SqueezeNet, and EfficientNet, which are all higher performing than their generic counterparts designed with the same defined parameters. [15] Neural Architecture Search initially has expensive computational requirements during the search phase and requires significant resources for initial training. The efficiency is extremely high when designing models for IoT devices and there is low-latency interference. [5]

5. Adaptive Inference

The final model compression technique we will overview is adaptive inference, where the computational complexity of a model is adjusted based on the input data. This model is effective because it works based on the idea that inputs will not require a consistent amount of power to reach their prediction. This works well for variable workloads because the simpler inputs can bypass deeper model layers and then reduce latency for the simple data. An application of adaptive inference is in the environmental and agriculture sectors, where real-time video analytics and edge AI systems are often utilized. [16] This model has a risk of compromising accuracy for the complex inputs that require full model processing, given the complexities of designing the dynamic architectures. The efficiency is high for real- time applications in edge devices with varied input complexities, and the latency is highly optimized. Table 1, seen below, depicts a comparative analysis of the model compression techniques.

Method	Compression Ratio	Accuracy	Inference Latency
Pruning	High	Moderate to High	Moderate
Quantization	High	Moderate to High	Very Low
Knowledge Distillation	Moderate	Moderate to High	Very Low
NAS	High	Very High	Low
Adaptive Inference	Variable	High	Very Low

Table 1. Comparative analysis of model compression methods.

The advancements to enhance energy efficiency and reduce computational complexity through model compression methods for deep learning models on edge devices vary based on the model parameters. Pruning and quantization are effective for rapid compression and high-speed inference for energy-constrained edge devices. Knowledge distillation is highly accurate for small models in text and speech. Neural Architecture Search has a high initial demand for computational resources but will provide an optimized network based on the given constrained parameters. [5] Adaptive inference is highly efficient for variable input complexity with an extremely low latency. To further the deployment of these compression techniques, new research demonstrates the effectiveness of maintaining accuracy while reducing model size by creating a network in which knowledge distillation and pruned models are used in conjunction. [8]

Neural Network Architectures

There are multiple lightweight neural network architectures designed for edge devices to drive efficiency and performance by reducing the number of parameters and computational power required. These neural architectures effectively utilize resources to minimize consumption for accurate image and object detection and recognition. The three deep-learning models we will examine are MobileNet, SqueezeNet, and EfficientNet, which are

all specifically designed for edge devices. The lightweight models balance performance and accuracy with the challenges and memory and power constraints of edge devices and their hardware. [17] A key model is MobileNet which uses depth-wise separable convolutions to reduce the number of floating- point operations (FLOPs). This lowers the number of parameters to adjust to achieve a five-time reduction in cost compared to the standard CNNs where the convolution is applied to all M channels. The lower amount of computation reduced overfitting, making this highly energy-efficient for mobile and IoT devices. [14] MobileNet has a lower accuracy at approximately 72% top-1 accuracy which is acceptable for edge applications. For real-time applications like object detection and facial recognition, MobileNet demonstrates low latency, balancing out the accuracy compared to larger models. SqueezeNet archives an extremely energy-efficient for memory- constrained devices as it requires 50x fewer parameters than the standard CNN, AlexNet, with the same level of accuracy. SqueezeNet uses fire modules, with a combination of 1x1 and 3x3 filters, to reduce model size. This network has the same level of accuracy as AlexNet but is less accurate than MobileNet or EfficientNet, which it makes up for with its lightweight design, allowing a very low latency idea for edge applications with quick response times. Another popular lightweight neural network architecture is EfficientNet. This model uses a compound scaling factor to scale the models' depth, width, and resolution to optimize computational resources and energy usage. This makes it much more efficient than general CNNs for edge deployment because the constant ratioed architecture reduces unnecessary computation. The smallest variant, EfficientNetB0, has a higher accuracy than MobileNet with approximately 77% top-1 accuracy with a similar efficiency. EfficientNet also lowers the number of floating-point operations compared to MobileNet. [17] The latency for this model is higher than MobileNet due to the complex architecture, but the optimized energy efficiency and accuracy balance this drawback. Table 2, seen below, depicts a

Table 2. Comparative analysis of neural network architectures.

Model	Energy Efficiency	Accuracy	Latency
MobileNet	High	Moderate	Very Low
EfficientNet	Moderate- High	High	Moderate
SqueezeNet	Very High	Low- Moderate	Very Low

Adaptive Models

Combining dynamic neural networks like early-exit architectures with model compression techniques like pruning or quantization is a promising approach for high accuracy and computational efficiency in edge computing. Through this, we can amplify efficiency gains and minimize trade-offs for real-world applications. This method adapts its computations based on the complexity of inputs and given constraints. [17]

1. Early-Exit Architecture

Early-exit architecture allows a network to "exit" sooner when the intermediate layers produce confident predictions and will allow the neural network to terminate computation sooner. This will increase efficiency rather than waiting for all layers to compute when rapid response Is required. Within the model, there are intermediate exit points in the inference process after specific layers. At one of these specific layers, if the output meets a certain threshold, the computation is completed, and more complex inputs will continue to process through deeper layers. [18] This is essential for real-time, low-power applications in IoT and mobile devices and will achieve faster inference times. Early exit architecture has less accuracy if the early exit occurs too quickly when the confidence threshold is incorrectly completed or if too many inputs exit early. This design is complex and will require additional processing of inputs to evaluate confidence at exit points creating a larger model.

Current applications of early exit architecture are BranchyNet and Multi-Scale DenseNet. BranchyNet is popular for image classification tasks with a high level of energy reduction and Multi-Scale DenseNet is used for processing simpler inputs without compromising complex model accuracy as it is designed for real-time inference. [16]

2. Dynamic Neural Networks

Dynamic neural networks adjust their computation based on input as well through three key approaches: dynamic layer skipping, dynamic width networks, and dynamic depth networks. Dynamic layer skipping skips unnecessary layers for simpler inputs by deciding which layers to activate based on the inputs. Dynamic width networks scale the number of active neurons by only activating a subset of neurons during inference to allow model compression based on resource constraints. Lastly, dynamic depth networks adjust depth without predefined exit points, which will quickly change model complexity. [19] Adaptive computation time in recurrent neural networks optimizes inference speed for simple inputs and maintains performance for complex ones, adjusting to hardware limitations. This dynamic model uses sophisticated mechanisms for decision-making but has high implementation costs and can lead to inaccuracies due to lack of predefined exit points. [6]

ALGORITHMIC OPTIMIZATION TECHNIQUES

Optimizing algorithms is vital for creating energy-efficient deep learning models on edge devices. Recent advancements combine methods to improve efficiency for complex tasks. This section covers three techniques for edge devices: sparse representations, low-rank factorization, and federated learning. [20]

Sparse Representations

Sparse representation models reduce computation and memory usage by storing only nonzero values based on the idea that most neural network weights will not compromise performance at zero. This allows for optimized energy consumption with a reduced number of computations through multiple approaches, including algorithms like weight pruning,

using ReLU to increase sparsity in activations where only a specific subset of neurons will be activated, and finding sparse sub-networks that are trained which will theoretically perform as well as a newly trained model. This is observed with Sparse BERT maintaining accuracy with a 40% reduction in omputational cost as dense BERT. [21] This is effective for both convolutional and recurrent neural network models and will have low storage needs because of the sparsity in weights. High levels of sparsity without retraining will potentially comprise accuracy and will require specialized hardware like GPUs optimized for sparse matrices. [22]

Low-Rank Factorization

Low-rank factorization approximates the weights of matrices A with m x n dimensions in neural networks and reduces the number of parameters by identifying redundancies using low-rank matrix decomposition. This method effectively compresses models by up to 50% and works best with fully connected layers to provide a high level of accuracy retention. [23] A common application of low-rank factorization is with speech recognition and image classification models in bioinformatics. This method requires advanced optimization techniques and is less effective for convolutional layers. The efficiency of this method is not as high as pruning and quantization and the latency improves with models that have heavy dense-layer dependencies.

Federated and Transfer Learning

Federated learning eliminates the constant energy-intensive communication with the cloud by allowing training models across multiple edge devices. This reduces the need to transfer data to a central server when training models while only transferring gradient updates that are made during this process to the cloud to minimize the loss function. This is important to scale efficiently across millions of edge devices, maintain privacy by retaining data on devices, mitigate risks associated with data transfer, and lower overall overhead for updates. [24] Federated learning can complicate training due to non-identical distribution across

devices and will increase computational demand on edge devices during local training. Similarly, transfer learning uses pre-trained models across edge devices for specific tasks. This will decrease the use of computational resources during training for higher efficiency.

HARDWARE ACCELERATORS

Edge-Specific Hardware

Edge-specific hardware reduces energy requirements through its tailored design, focusing on inference to minimize unnecessary computations and power usage from training. This hardware supports quantization and performs computations using lower precision, such as INT8, with pre-optimized libraries to minimize energy consumption when deploying models across edge devices. This integrated framework is used in the NVIDIA Jetson Nano, a small AI accelerator designed for edge computing with a GPU based on NVIDIA's Maxwell architecture. The framework supports TensorFlow, PyTorch, and ONNX, which are used for myriad IoT applications. [25], [26] Another popular hardware accelerator designed for inference tasks is Google Coral, which is built around the Edge TPU. This has the highest impact on medical imaging and environmental monitoring as it delivers up to 4 trillion operations per second at 2 watts. Google Coral runs models quantized to 8-bit integers and is ideal for battery-powered IoT devices. Lastly, edge TPUs are specialized to perform inferencing on low-power devices with high accuracy for matrix multiplication operations essential for deep-learning computations. We can further this through algorithm-hardware co-design between the deep learning algorithms and hardware to create a custom, highly optimized design. This will allow hardware architectures to work in conjunction with specific algorithms eliminating redundant operation, memory limitations, and increasing scalability. [27]

APPLICATION CHALLENGES AND FUTURE ADVANCEMENTS

Deep learning for edge devices has the potential to revolutionize various sectors. One of the most significant areas of improvement is health technology and medical applications. AI

edge devices can process X-rays, CT scans, and MRIs on-site to detect diseases in real-time using Convolutional Neural Networks. [8] Recurrent Neural Networks can be used in patient monitoring systems for time-series data. Wearables with deep learning capabilities can analyze vital signs to detect abnormalities for faster response time and early detection. Co-design hardware like NVIDIA Jetson Nano will accelerate real-time medical imaging processing at healthcare facilities by increasing efficiency and reducing power. Another sector that is highly impacted by edge devices is autonomous vehicles for real-time decision-making systems like object detection, lane tracking, and collision avoidance. [2] EfficientNet can optimize accuracy and latency in visual tasks like lane detection or increase traffic flow by dynamically adjusting signals based on current traffic data. Agriculture can deploy MobileNets to analyze drone and satellite imagery to monitor crop health as well as irrigation based on analysis of weather and soil data from Google Coral EdgeTPU with IoT sensors. [11] There is an unlimited amount of potential for deep learning for edge devices in every major sector. This will work towards managing the numerous challenges due to the constraints of devices, complexity of deep learning models, and the need for real-time processing.

CONCLUSION

In this literature review, Energy-Efficient Deep Learning Techniques, Algorithmic Optimization Techniques, and Hardware Accelerators for Energy Efficiency were analyzed for implementation in major sectors. MobileNet has high energy efficiency with low latency for real-time applications with lower accuracy compared to EfficientNet. SqueezeNet has high energy efficiency and low latency with lower accuracy compared to both MobileNet and EfficientNet. EfficientNet has higher accuracy than SqueezeNet and MobileNet with a higher latency due to complexity. Application of early-exit architectures are image classification and speech recognition for the space and defense sectors or mobile use. Applications for dynamic neural networks are object detection and time-series analysis for health technology and autonomous vehicles. These approaches represent the cutting edge of sustainable edge computing, making them essential for applications ranging from autonomous vehicles to environmental monitoring and healthcare.

REFERENCES

Gupta, S. Das, and P. Roy, "Optimizing Edge AI with Energy-Efficient Neural Architectures," arXiv preprint arXiv:2312.00333v2, 2023. [Online]. Available: https://arxiv.org/html/2312.00333v2

Y. Zhang, H. Chen, and T. Wang, "Hardware- Accelerated Energy Efficiency in Edge AI Systems," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 6, pp. 1025–1035, Jun. 2024. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10198424

S. Ahmed and M. Saleh, "Energy-Efficient Edge Devices: Applications and Challenges," in *Renewable Energy and Environmental Sustainability*, vol. 29, no.4,pp.158–168,2024.[Online].Available: https://www.sciencedirect.com/science/article/pii/S1110016824001091#bib15

A. S. Javed, A. P. Raja, and M. A. Shah, "Energy- efficient IoT with deep learning: Optimizing resource allocation in smart grids," *ResearchGate*, 2024. [Online]. Available: https://www.researchgate.net/publication/377863218_Energy-Efficient_IoT_with_Deep_Learning_Optimizing_Resource_Allocation_in_SmartGrids

S. J. Wu, M. S. Kang, and C. S. Xie, "Optimizing energy use in deep learning models for smart grid applications," *Energy Procedia*, vol. 37, no. 12, pp. 1214-1226, 2024.[Online].Available: https://www.sciencedirect.com/science/article/pii/S0925231224008671

M. S. Zhang, J. L. Xu, and H. M. Lee, "Smart energy management for AI applications in IoT systems," *Energy Procedia*, vol. 37, no. 2, pp. 834-847, 2024. [Online]. Available: https://pdf.sciencedirectassets.com/271597/1-s2.0-

S0925231224X00296/1-s2.0-S0925231224008671/main.pdf

M. B. Yang, "Green AI: Reducing the carbon footprint of machine learning," *E-SDST*, 2024. [Online]. Available: https://esdst.eu/green-ai-reducing-the-carbon-footprint-of-machine-learning/#:~:text=Optimize%20Model%20Efficiency&text=Optimizing%20model%20efficiencies%20through%20the,overall%20carbon%20footprints%20of%20ML

L. H. Wu, "Energy-efficient deep learning for real-time applications in edge devices," *arXiv*, 2024. [Online].Available: https://arxiv.org/html/2404.07236v2

Wang et al., "Energy-efficient deep learning algorithms for edge computing: A survey," *IEEE Access*, vol. 11,pp. 12435-12450, 2023. [Online]. Available: https://arxiv.org/pdf/2110.00961

J. S. Lee et al., "Dynamic pruning for energy-efficient deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 5, pp. 2377-2389, May 2023. [Online]. Available: https://arxiv.org/pdf/1702.05309

Kim et al., "Low-rank factorization for model compression in energy-efficient AI systems," *IEEE Journal of Emerging and Selected Topics in Circuits and Systems*, vol. 13, no. 1, pp. 15-28, Jan. 2024. [Online].Available: https://arxiv.org/pdf/2106.08962

H. R. Patel and S. K. Mohanty, "Federated learning: A decentralized approach for reducing energy consumption in deep learning systems," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 3, pp. 1615-1624, Mar. 2024. [Online]. Available: https://arxiv.org/pdf/1710.09282

H. S. Fang et al., "Energy-efficient deep learning models: Challenges and opportunities," *arXiv preprint arXiv:2310.18329v2*, 2023. [Online]. Available: https://arxiv.org/html/2310.18329v2#bib.bib24

J. Smith et al., "Optimization techniques for energy- efficient neural networks," *Neurocomputing*,vol.438,pp.95-110,2023.[Online].Available: https://www.sciencedirect.com/science/article/pii/S0925231223001388#b1360

J. K. Patel, "Deep learning for IoT: Energy-efficient frameworks," in *Deep Learning Applications for IoT Devices*, Elsevier, 2023, pp. 245-270. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128238172000152

X. Zhang and Y. Wang, "Efficient computing for real- time AI: Challenges in energy and processing," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 60, no. 4, pp. 276-280, Apr. 2014. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6638949

S. K. Singh et al., "Energy-efficient AI on edge devices: Emerging trends and solutions," *arXiv preprint arXiv:2207.00112*, 2022. [Online]. Available: https://arxiv.org/pdf/2207.00112

P. Kumar et al., "AI-driven approaches for low-power systems in edge computing," in *Proceedings of the IEEE International Conference on Artificial Intelligence Computing (AIC)*, vol. 2023, pp. 1-7, Oct. 2023. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10608160

Y. Chen and X. Liu, "Low-energy image recognition: Algorithms and system design," *IEEE Transactions on Circuits and Systems for Video Technology*,

vol. 31, no. 7, pp. 275-288, July 2021. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9394240

J. Bornholt, "Energy-efficient hardware for real-time data processing," in *Proceedings of the 12th Annual Hot Chips Symposium (HOTCHIPS)*, Stanford, CA, 2012. [Online]. Available: https://jamesbornholt.com/papers/energy-hotchips12.pdf

optimization in industrial IoT systems: Emerging methodologies, "*Engineering Applications of Artificial Intelligence*, vol. 119, Oct. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197623002191

S. K. Gupta et al., "Scalable Deep Learning on Resource- Constrained Edge Devices," *arXivpreprintarXiv:2306.02652*,2023.[Online]. Available: https://arxiv.org/abs/2306.02652

M. Abadi et al., "TensorFlow: A System for Large-Scale Machine Learning," *arXiv preprintarXiv:1603.08983*,2016.[Online]. Available: https://arxiv.org/abs/1603.08983

J. S. Botha, "Optimization Techniques for Distributed Energy Systems," Stellenbosch University, 2021. [Online]. Available: https://scholar.sun.ac.za/server/api/core/bitstreams/7a84c521-16b6-4844-8f2e-bfaf97296df4/content

J. W. Smith, *Algorithmic Optimization Techniques: A Comprehensive Guide*, 1st ed., Springer,2018.[Online].Available: https://books.google.com/books?id=KBxtDwAAQBAJ

Numenta Research Team, "Sparsity Without Sacrifice: Accurate BERT with 10x Fewer Parameters," *Numenta Blog*, Dec. 2021. [Online]. Available: https://www.numenta.com/blog/2021/12/13/sparsity-without-sacrifice-accurate-bert-with-10x-fewer-parameters/

T. Lei et al., "Foundations of Sparse Neural Networks in Large-Scale AI Models," *arXiv preprint arXiv:2310.04564*, 2023. [Online]. Available: https://arxiv.org/abs/2310.04564