A Review of Model Lightweighting Techniques Based on Edge Computing

Jinrui Liu *

Nanjing University of Posts and Telecommunications, Nanjing, 210046, Jiangsu, China

* Corresponding Author Email: P24000214@njupt.edu.cn

Abstract. With the booming development of the Internet of Things (IoT) and the wide application of artificial intelligence at the edge, it has become an inevitable trend for deep learning models to be deployed on resource-constrained edge devices. Model lightweighting techniques have made significant progress in the past five years as a solution to the problem of limited computation, storage and power consumption resources in edge devices. In this paper, we systematically review the research on model lightweighting techniques for edge computing in the past five years (2020-2025). In this paper, the mainstream lightweighting methods are classified into two categories: parametric compression and structural compression based on the core idea of compression. Parameter compression focuses on reducing redundant parameters, and focuses on the latest breakthroughs in the directions of parameter pruning, parameter quantization, and parameter sharing, which significantly reduce the model size and computation volume while effectively balancing accuracy and hardware friendliness. Structural compression techniques, on the other hand, focus on designing more efficient network architectures, with an in-depth discussion of the core ideas and representative work on compact network design and knowledge distillation. Then, examples are given on the application scenarios and key challenges of lightweight models. Finally, future research directions for the above challenges are envisioned.

Keywords: Model Lightweighting Edge Computing Resource-Constrained Devices.

1. Introduction

Over the past decade, the Internet of Things (IoT) has gained significant traction, and Artificial Intelligence (AI) has become even more widely used. The previous approach of sending large-scale edge data to the cloud and processing it has increasingly shown its limitations in terms of latency, bandwidth, privacy, and other issues. It has become an inevitable trend to install deep learning models directly onto edge devices for local computation. However, edge devices (cell phones, sensors, automotive computing platforms) are greatly affected by resource constraints, computing power, memory capacity, storage space and power consumption are constraints, and large-scale deep learning models have huge parameters and computation volume, which cannot meet the above constraints, and thus, without significantly weakening the performance of the model on the premise of the model lightweight to meet the edge environment has become a trend.

Although there are some review articles on edge-oriented deployment or model compression [1, 2], some of the references are old [1], or they have not sorted out and categorized the related technologies of lightweighting in edge computing in the recent years [2], and this paper is written to address the research gaps, focusing on the model lightweighting for edge computing in the recent 5 years (2020-2025). This paper addresses this research gap and focuses on the research progress of model lightweighting techniques for edge computing in the last 5 years (2020-2025).

The paper is structured as follows: Chapter 2 summarizes and reviews the mainstream lightweighting methods from the perspective of compression, Chapter 3 elucidates the typical edge application scenarios of lightweight models, Chapter 4 summarizes the key challenges, and Chapter 5 discusses the future directions. See Figure 1 for details.

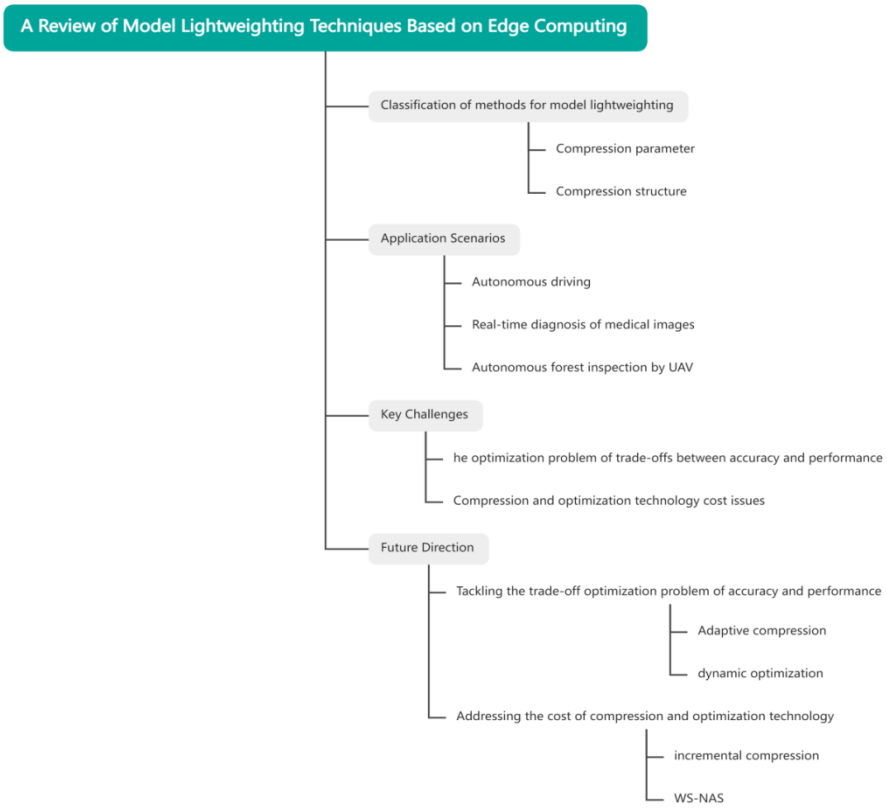

Fig. 1 Structure of paper

2. Classification of Model Lightweighting Methods

Here, model lightweighting methods mainly include hardware adaptation optimization and compression and acceleration methods for the model itself, and this paper classifies the compression and acceleration methods for the model itself. Based on the research paper, this paper classifies the compression methods of deep learning models into three categories according to the compression idea: compression parameters and compression structure and hybrid methods.

2.1. Compression parameters

Compression parameters that are to zero, mapping, sharing and other ways to reduce redundant parameters, in order to achieve the purpose of preventing overfitting, reducing the overhead of storage and computation. Compressed parameters include: parameter pruning, parameter quantization, parameter sharing, and low-rank decomposition. Since the past research on low-rank decomposition has been quite mature, coupled with the wide application of 1×1 convolution, such small convolution kernels are not favorable for using low-rank decomposition methods, and thus there is less related literature in the past five years, so we will not go into details here.

Parameter pruning is the process of formulating evaluation criteria for network parameters based on a pre-trained large-scale model and trimming redundant parameters based on these criteria.Rheey et al. showed that the Shapley Value-based layer pruning method, SV-SAE, achieves high accuracy with a small number of connections retained, and minimal performance loss at high sparsity rates.

In addition to assessing the importance of connections between neurons, direct evaluation of neurons can also serve the purpose.The MDPP technique was used for neuron screening by Mariet et al. in [3], who utilized the technique to redistribute data from neurons so that they could be incorporated into the remaining neurons without further model adjustments.2020 Liu, XF et al[4] Designed group-level sparse induction paradigm to implement network self-pruning (LDNN) Proposed improved two-step training TLDNN: coarse partitioning of base classes followed by fine

optimization for confusing types (e.g., QAM16/QAM64) 2024 Sahbi, H [5] Proposed designing CTF parameterization paradigm: fusion of Hadamard product via four levels of pruning masks at channel, row, column and element levels to achieve a hierarchical constraint mechanism that automatically invalidates fine-grained pruning when coarse-grained pruning is activated, enabling the possible simultaneous use between structured and unstructured pruning. In the same year, to address the suboptimal compression problem caused by traditional pruning methods relying on the assumptions that "filters with small weighting paradigms are unimportant" and "highly similar filters are redundant", Liao.w et al. [6] proposed a two-stage filter pruning method (CSP) that incorporates cosine spatial correlation. They proposed a two-stage filter pruning method incorporating cosine spatial correlation (CSCTFP) to address the paradox of deploying complex models for resource-constrained embedded devices.

The above work shows that pruning is effective in reducing the amount of computation (number of parameters), is hardware friendly, relatively fast in computation, but subsequently more or less requires recovery of accuracy, and is suitable for edge devices and real-time reasoning.

Parameter quantization is the representation of 32-bit floating-point network parameters (e.g., weights, activation values, gradients, errors, etc.) by lower bitwidths, often using uniform bitwidths (1-bit, 2-bit, etc.), but also by a certain strategy to freely combine different bitwidths.

Binarization is a form of limiting network parameters to 1 or -1, i.e., compression of parameters to 1-bit, which reduces the storage and memory requirements of the model, while replacing the original multiplication operations with addition or displacement operations. Xing, XR et al[7] proposed the Binarized Pre-training Basis Transformer (BiPFT) with a view to solving the problem of large-scale pre-training models in resource constrained deployments, such as mobile devices. Arithmetic and Memory Challenges of Device Deployment Becerra-Rozas, M et al [8] proposed reinforcement learning-driven binarization scheme selection strategy, which argues for integrating reinforcement learning and meta-heuristic techniques in a BSS framework: binarization scheme selection is modeled as an action selection mechanism, and the binarization process is dynamically optimized by reinforcement learning strategies.

Parameter sharing refers to reducing the number of parameters by using methods such as structured matrices or clustering to project network parameters. Unlike parameter pruning which directly cuts unimportant parameters, parameter sharing designs a form of mapping

Circulant Matrix, as a kind of structured matrix, is used as a common mapping form for parameter sharing.Yunhe Wang et al.[9] use Circulant Matrix to reduce the feature map embedding complexity to O(d log d) (d is the feature map dimension) Li.FY et al.[10] propose a hierarchical parameter sharing strategy to improve the ability of semantic composability modeling. Clustering of hidden unit weights based on semantic hierarchy Above methods

As far as the existing work is concerned, the model lightweighting of compressed parameters is more developed on pruning and has more room for development for parameter quantization, which has become more popular in recent years, the prospect of parameter sharing is promising. In terms of method mixing, the mixing of several methods is becoming more and more popular. It has a bright prospect for speeding up the processing speed and reducing the expenses. In recent years, the work focuses on the comprehensive use of several methods, and puts forward different ideas from the past, such as literature 6 [6], and the edge of the device is more closely linked.

2.2. Compression structure

Compression structure includes: compact network and knowledge distillation.

Compact networks are networks that are redesigned to be more compact by constructing specially structured filters, network layers, or even networks, and retrained to achieve performance suitable for deployment in edge networks with limited resources. There are traffic saving approaches such as Chaudhary. Shubham [11] et al. proposed to improve the quality of experience (QoE) and reduce the latency of real-time video streaming by encoding video in chunks and utilizing parallel transmissions over multi-device cellular networks in order to solve the lagging and high latency problem of live

online classes in remote areas due to insufficient bandwidth in cellular networks. There are also arithmetic-saving methods such as Anzhe Chen et al [12] proposed to propose a compact model with a unified multi-operation scheme to obtain the best operating conditions by deriving a stochastically switched differentiable distribution model for co-optimization with neural networks.

Compact network comparing parameter compression does not need to use pre-trained large model, and does not need to be fine-tuned to improve the performance, which relatively saves time and cost, and has the characteristics of less storage arithmetic overhead and superior network performance, but due to the special structure, it is difficult to be mixed with other methods. It is not suitable as a pre-training model.

Knowledge distillation, i.e., the compression model used to train a strong classifier with pseudo-data labeling and replicate the output of the original classifier, requires two models, the teacher model and the student model. The teacher model is generally a large model with relatively good performance, and the knowledge in the teacher model is migrated to the student model through Soft Targets, so that the student model achieves the performance of the teacher model with a more compact and efficient structure. In industrial applications Zhitao Wen et al [13] proposed triple semantic-aware knowledge distillation network (TSKD), which contains three modules, namely dual relation distillation (DRD), decoupled expert distillation (DED) and cross-response distillation (CRD), to enhance the semantic perception at the level of relation, feature and response, respectively. In the computer vision target detection task Hong Liang et al. proposed to present the ADKD framework (with Adaptive Generative Distillation AGD and Multiplexed Head Distillation RHD modules), which enhances student representations through AGD, and resolves the target discrepancy by multiplexing the teacher detection head with the mask feature maps using RHD in order to solve the problem of inconsistency between the distilled target and the real target in the existing knowledge distillation methods, as well as the problem of the students' network to the insufficient learning of teacher features. Knowledge distillation can make deep networks shallow and reduce computational cost, but it has limitations. Its generalization in other tasks is not good; and as of now, its compression ratio and the performance of the distilled model still have much room for improvement.

3. Application Scenarios

3.1 Autonomous driving

Autonomous driving is to allow a computer to operate a motor vehicle automatically and safely without any human initiative. In its operation to ensure the safety of human and vehicle, the deep learning model requires quick response, accurate judgment, and can keep driving even in the case of disconnection and other situations. The lightweight model can provide high-speed processing under the limited arithmetic power of the vehicle.

3.2 Real-time diagnosis of medical images

Real-time diagnosis of medical images is to use AI technology to quickly and automatically analyze medical images to assist doctors in identifying lesions and making preliminary judgments. In its application scenarios, especially in emergency rescue, remote area consultation or grassroots hospitals, the diagnostic model needs to be able to respond instantly, accurately identify key pathological features, and work independently in unstable networks or even disconnected environments. Lightweight models can run efficiently with limited computing power on mobile or edge devices to provide reliable and immediate diagnostic support.

3.3 Autonomous forest inspection by UAV

Autonomous forest inspection by UAV requires UAVs to fly independently in forested areas to identify fire hazards (e.g., smoke spots, hot zones), pest and disease areas, or signs of illegal logging in real time. In the face of the complex and changing forest environment and communication delays caused by long-distance flights, it is necessary to quickly parse the camera and sensor data on the

airborne side, and have the ability to autonomously avoid obstacles and continue to carry out the inspection task. The lightweight model can process the image information at high speed with limited computing power and respond quickly.

4. Key Challenges

4.1 The optimization problem of trade-offs between accuracy and performance

Optimization problem of accuracy and performance trade-offs. Edge devices typically have tight resource constraints and limited computational power, memory capacity, storage space, and power consumption budgets. The deployment of deep learning models on them inevitably requires lightweighting of the models. In the process of lightweight compression, too much pursuit of lightweight reasoning speed may lead to the model in the complex data, the performance of the decline and cannot meet the accurate demand, and vice versa, excessive requirements on the accuracy level of the model may not be able to effectively run in the current edge equipment under the substantive requirements and functionality, so in the limited resources of the edge equipment and the application scenario of the rigid demand for accuracy and performance, an optimal balance needs to be sought between. Therefore, an optimal balance needs to be sought between the limited resources of edge devices and the rigid demands for accuracy and performance of application scenarios.

4.2 Compression and optimization technology cost issues

If the compression methods sorted out in this paper have many, each with its own advantages and disadvantages, can be mixed use use Yes How to debug after use is its primary problem How to trade-off Manual micro-debugging relies on experts with high costs, and automated methods with high time costs，The computational cost is high in the Neural Architecture Search (NAS) approach.

5. Future Direction

5.1 Tackling the trade-off optimization problem of accuracy and performance

To cope with the trade-off between accuracy and performance optimization problem now there are two kinds of solution ideas Adaptive compression technology and dynamic optimization technology. Adaptive compression is a technique that dynamically selects the optimal compression algorithm based on data characteristics. Its main purpose is to automatically adjust the compression strategy in different data scenarios to achieve the best compression effect and performance. Generally, the data to be solved is analyzed first, and then the most suitable compression method is selected from these data, and then dynamically adjusted according to the situation, so as to maintain the optimal compression method at all times. Dynamic optimization technology is through the perception of the environmental context and resource status, in the inference stage of the real-time adjustment of the model computing strategy to achieve the optimization effect.

5.2 Addressing the cost of compression and optimization technology

To cope with the cost of compression and optimization techniques for long time cost, incremental compression can be used, and for high-cost automated compression techniques there is Weight-Sharing Neural Architecture Search (WS-NAS) whose core idea is to avoid training each candidate architecture individually by sharing the weights in the Supernet, in order to avoid the need to train each candidate architecture individually. Each candidate architecture is trained separately to reduce the search cost, time and financial expenses.

6. Conclusion

In this paper, we systematically review the development of model lightweighting in edge computing in the past five years, categorize the existing work, and propose to classify it by parameter compression and structure compression. It then provides examples of application scenarios and key challenges, and concludes with an outlook on solutions to the above key challenges.

References

[1] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A Survey of Model Compression and Acceleration for Deep Neural Networks," arXiv:1710.09282, Jun. 14, 2020. doi: 10.48550/arXiv.1710.09282.

[2] H.-I. Liu et al., "Lightweight Deep Learning for Resource-Constrained Environments: A Survey," ACM Comput. Surv., vol. 56, no. 10, pp. 1–42, Oct. 2024, doi: 10.1145/3657282.

[3] Z. Mariet and S. Sra, "Diversity Networks: Neural Network Compression Using Determinantal Point Processes," arXiv:1511.05077, Apr. 18, 2017. doi: 10.48550/arXiv.1511.05077.

[4] H. N. Karimah, C. Lee, and Y. Seo, "Batchnorm-Free Binarized Deep Spiking Neural Network for a Lightweight Machine Learning Model," Electronics, vol. 14, no. 8, p. 1602, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/14/8/1602

[5] H. Sahbi, "Coarse-to-Fine Pruning of Graph Convolutional Networks for Skeleton-based Recognition," in 2024 International Conference on Content-Based Multimedia Indexing (CBMI), New York: IEEE, 2024, pp. 79–85. doi: 10.1109/CBMI62980.2024.10859230.

[6] L. Wei, L. Guanghui, D. Chenglong, and Z. Feifei, "Two-stage filter pruning incorporating cosine-spatial correlation," J. Image Graph., vol. 29, no. 12, pp. 3628–3643, 2024.

[7] X. Xing et al., "Bipft: Binary pre-trained foundation transformer with low-rank estimation of binarization residual polynomials," in Proceedings of the AAAI Conference on Artificial Intelligence, 2024, pp. 16094–16102. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/29542

[8] M. Becerra-Rozas, B. Crawford, R. Soto, E.-G. Talbi, and J. M. Gómez-Pulido, "Challenging the Limits of Binarization: A New Scheme Selection Policy Using Reinforcement Learning Techniques for Binary Combinatorial Problem Solving," Biomimetics, vol. 9, no. 2, p. 89, 2024. [Online]. Available: https://www.mdpi.com/2313-7673/9/2/89

[9] Y. Wang, C. Xu, C. Xu, and D. Tao, "Beyond filters: Compact feature map for portable deep model," in International Conference on Machine Learning, PMLR, 2017, pp. 3703–3711. [Online]. Available: http://proceedings.mlr.press/v70/wang17m.html

[10] F. Li, M. Chi, D. Wu, and J. Niu, "Hierarchical Parameter Sharing in Recursive Neural Networks with Long Short-Term Memory," in Neural Information Processing, vol. 10635, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Cham: Springer, 2017, pp. 582–592. doi: 10.1007/978-3-319-70096-0_60.

[11] S. Chaudhary et al., "COMPACT: Content-aware Multipath Live Video Streaming for Online Classes using Video Tiles," in Proceedings of the 16th ACM Multimedia Systems Conference (MMSYS 2025), 2025, pp. 201–213. doi: 10.1145/3712676.3714451.

[12] A. Chen, J. Hu, H. Ma, Y. Jiang, and B. Yu, "Differentiable Distribution Model of Stochastic Volatile Memristor-Based Neuron," IEEE Trans. Electron Devices, vol. 72, no. 4, pp. 1709–1714, Apr. 2025, doi: 10.1109/TED.2025.3538660.

[13] Z. Wen, J. Liu, H. Zhao, and Q. Wang, "A triple semantic-aware knowledge distillation network for industrial defect detection," Comput. Ind., vol. 166, Apr. 2025, doi: 10.1016/j.compind.2025.104252.