A review of lightweight models for vehicle-road collaboration

Mingze Sun*

Nanjing University of Posts and Telecommunications, Nanjing, 210003, Jiangsu, China

* Corresponding author email: ajaxs@pdx.edu

Abstract

Autonomous driving for vehicle-road coordination is transitioning from single-vehicle intelligence to networked collaborative intelligence. However, the exponential growth in data volume and computational demands driven by improved sensing accuracy has made model lightweighting a critical technology for achieving real-time, secure deployment under heterogeneous resource constraints across edge and cloud endpoints. This paper systematically reviews lightweight research in vehicle-road coordination scenarios over the past five years, categorizing, comparing, and critically evaluating mainstream approaches from three dimensions: parameter quantification, model architecture and knowledge distillation. It subsequently identifies common challenges in the field: precision-efficiency trade-offs, complexity in compression-training-deployment chains, cross-device consistency drift, and continuous adaptation to dynamic scenarios. Finally, by integrating cutting-edge directions such as meta-learning, urban-level agent networks, online continuous learning, and embedded security verification, the paper proposes a future vision where lightweight technologies evolve from "static, empirical compression" to "dynamic, self-evolving compression." This study provides methodological references and technical perspectives for efficient, large-scale deployment of vehicle-road coordination systems.

Keywords

Vehicle-road collaboration; model lightweight; parameter quantization; network pruning; knowledge distillation; edge computing.

1. Introduction

In recent years, the rapid adoption of cellular vehicle-to-everything (C-V2X), Dedicated Short Range Communications (DSRC), and 5G/6G technologies have sparked widespread academic interest in Vehicle-Infrastructure Collaboration (VIC). As high-level autonomous driving transitions from closed test environments to open urban networks, bottlenecks in single-vehicle intelligence—such as perception range limitations, computational redundancy requirements, and cost control challenges—are becoming increasingly apparent. Technologically, this evolution is shifting from isolated vehicle intelligence to networked collaborative intelligence. [1] VIC integrates vehicles, roadside units, and cloud platforms through low-latency cellular networks and 5G/6G infrastructure. By leveraging a layered edge-cloud computing architecture to share perception data and decision-making intent, it mitigates the limitations of individual vehicle sensors' visibility and computational constraints, enabling real-time decision-making in dynamic environments. [2] However, the exponential growth in data volume and computational demands accompanies improved sensing accuracy: The Automotive Edge Computing Consortium (AECC) predicts that by 2025, over 400 million connected vehicles worldwide will continuously generate ZB-scale perception and positioning data annually. Direct reliance on cloud-side processing would struggle to meet the latency thresholds required for object detection safety standards. Therefore, it is urgent to sink the model reasoning and training tasks to the roadside edge

(MEC) and the vehicle end ECU, and model lightweight is the key technology to solve the heterogeneous resource constraints of edge cloud.

With the advancement of vehicle-road coordination technology, modern vehicles are equipped with numerous sensors such as radar and cameras to monitor various parameters inside and outside the vehicle. These devices generate massive amounts of data, which grow exponentially when combined with vehicle-to-vehicle connectivity. If complex large models are still employed at this stage, not only would deployment to resource-constrained edge computing devices become challenging, but computational latency could also compromise the safety of the vehicle-road coordination system. [3] Meanwhile, as deep learning models continue to improve in accuracy for perception, prediction, and decision-making tasks, the demand for computing power in vehicle applications is growing exponentially. Typical advanced driver assistance systems face challenges in meeting both latency and precision requirements simultaneously due to constraints like power budgets and peak computational demands of on-board ECUs. [4] Although traditional cloud computing provides ample computing power, transmitting high-resolution point clouds or video frames through wireless links often leads to excessive delays caused by network jitter or sudden congestion, thereby threatening the real-time performance and reliability of mission-critical services.

Existing lightweight review studies predominantly focus on single compression paradigms or general mobile devices, lacking systematic analysis of unique constraints such as heterogeneous hardware in vehicle-road coordination, cross-domain data consistency, and dynamic scenario drift. [5-6] To address this gap, this paper analyzes recent research achievements from four perspectives of model lightweighting: parameter quantization, model structure optimization, and knowledge distillation, based on the characteristics of different compression methods. The core concept of parameter quantization involves converting continuous numerical values into discrete ones. Model structure compression refers to reducing or replacing model structures through techniques like low-rank approximation, network pruning, and compact layer design without altering weight bit width, thereby lowering complexity. Knowledge distillation enables significant model compression while maintaining computational capabilities comparable to complex teacher networks. We will analyze their features and drawbacks in subsequent chapters. Specifically, we will provide further elaboration in later sections. The remaining sections are organized as follows: Section 2 introduces background knowledge of vehicle-road coordination and model lightweighting; Section 3 summarizes and categorizes existing literature on model lightweighting; Section 4 outlines key challenges in vehicle-road coordination model lightweighting; Section 5 proposes feasible future research directions; and Section 6 provides a conclusion.

2. Background knowledge

2.1. Vehicle-road collaboration

Vehicle-road coordination represents a next-generation intelligent transportation paradigm that integrates vehicles, roadside infrastructure, and cloud computing as core components. Through cellular vehicle-to-everything (C-V2X), Dedicated Short-Range Communications (DSRC), or 5G/6G ultra-low latency networks, it enables real-time, bidirectional, and closed-loop transmission of traffic data across end devices, edge systems, and cloud platforms. Evolving from early-stage vehicle-to-vehicle collaboration, this system progresses through three phases: The initial phase focused on single-vehicle intelligence relied solely on onboard sensors for perception and decision-making. The second phase introduced vehicle-to-vehicle communication, allowing shared low-dimensional information like vehicle positions and speeds to enable collision warnings. The current third phase—vehicle-road coordination—features roadside units (RSUs), millimeter-wave/radar systems, Multi-access Edge Computing

(MEC) infrastructure, and cloud-based big data platforms. These components enhance local perception to global awareness and transform isolated decision-making into collaborative solutions, significantly reducing computational demands and costs per vehicle. The system architecture comprises four layers: Perception Layer (comprising vehicle OBU and roadside multimodal sensors including HD cameras, millimeter-wave radars, LiDAR, and weather stations for raw data collection); Network Layer (achieving millisecond-level transmission via 5G NR-uRLLC, NB-IoT, or Gigabit Ethernet); Platform Layer (enabling data fusion and digital twin functionalities through roadside MEC and central cloud); and Application Layer (providing control commands for autonomous driving, traffic signal optimization, public transit priority management, and emergency lane scheduling scenarios).

2.2. Model lightweight

Model lightweighting is a systematic approach that enables deep learning models to operate in real-time on resource-constrained devices such as vehicle ECUs, roadside MEC nodes, and mobile NPU units without significantly compromising task accuracy. This method reduces parameters, computational load, memory requirements, and energy consumption. In the context of vehicle-road coordination, while cloud-based systems typically train large models with billions of parameters for extreme precision, vehicle-side power consumption is strictly limited. Although roadside MEC offers higher computing power, it must simultaneously serve over 100 vehicles. Therefore, model optimization becomes essential to meet end-to-end latency requirements.

3. Model lightweight technology

This section will be introduced in the order of parameter quantization, model structure and knowledge distillation. Table 1 summarizes the three main compression methods.

Table 1. Three main compression methods.

view of research	Compression method	Method description	superiority	limit
Parametric quantification	INT8/INT4 quantization, binarization (BNN), ternarization (TTQ)	Map 32-bit floating-point weights/activations to 8-bit or even 1-bit integers; QAT or PTQ restores precision	• Display memory reduction: 4-32 times • Integer/Bit operation acceleration • Industrial NPU is widely supported	•Training time increase • Depthwise convolution SNR drops sharply • Displacement caused by drift of vehicle/road calibration set
Structural compression	Low-rank approximation (SVD/Tucker), channel pruning, and compact layer design (MobileNet, NAS)	Do not change the bit width, delete or reconstruct the network topology: matrix decomposition, attribution/Taylor pruning, deep separable convolution	• Flexible compression granularity (channel/weight) • Can be quantified/ distilled in superposition • Low rank fine-tuning cost is low	• Low rank: the accuracy drops off when the rank is less than 1/4 • Pruning: sensitive to super parameters, irregular sparse and difficult to compile • Compact layer: High cost of manual/NAS search
knowledge distillation	Soft label distillation, Logit Perturbation, reverse KL, meta-learning distillation	Teacher-student framework: Use the output distribution or intermediate features of the large model to train the small model	• Flexible migration across tasks/modes • Plugin implementation, easy to overlay other compression •No structural modification is required	• It fails when the capacity of teachers and students is large • Additional training of teachers and temperature/weight adjustment is required, and the time of parameter adjustment is about the same as weight training

3.1. Lightweight model based on parameter quantization

The main idea of parameter quantization is to convert the continuous values in the model into discrete values, so that the weights and activation values originally stored with 32-bit floating-point numbers are compressed into 8-bit or even lower integers, thus greatly reducing the demand for video memory, speeding up the inference speed, and enabling it to run on smaller GPUs, thus speeding up the inference speed.

In recent years, the academic community has proposed several methods for parameter quantization: binary neural network (BNN) training, ternary weight networks, and INT8 quantization. Regarding BNN training, [7] mentioned that under different frameworks, the parameter gradients of neural network weights and activation values during training are constrained to +1 or-1, and the filter repetition phenomenon is utilized to reduce time complexity. For ternary weight networks, [8] proposed Training-Style Ternary Quantization (TTQ), which compresses network weights into three values. Each layer introduces two trained scaling coefficients and uses two backpropagation gradients to perform training quantization

In terms of INT8 quantization, literature [9] proposes a quantization scheme that establishes a mapping relationship between real numbers and integers through affine transformations, combined with quantized perception training to maintain model accuracy. Literature [10] quantizes the weight parameters and activation functions in the YOMANet model into 8-bit integers (int8) to reduce storage requirements, minimize hardware resource consumption, enhance parallel computing efficiency, and adapt to the resource constraints of edge FPGAs. literature [11] first uses FP32 full precision to complete the training of the PointPillars model and then uses the default calibrators of OpenVINO / TensorRT for post-training INT8 quantization, enabling the detection model to achieve real-time deployment in on-vehicle or roadside systems. While parameter quantization offers significant training acceleration through reduced bit width, it inevitably leads to increased training duration and unavoidable accuracy loss. Different convolution types may experience adverse effects: For instance, quantizing Depthwise convolutions drastically reduces the signal-to-noise ratio (SNR) of useful signals. Moreover, quantization parameters are dataset-dependent – in vehicle-road coordination scenarios where calibration datasets differ between road and vehicle ends, this can cause systematic numerical misalignment, resulting in substantially reduced collaborative perception accuracy.

3.2. Model lightweight based on model structure

Model compression based on model structure refers to reducing or replacing redundant channels and modules of the model structure by low-rank approximation, network pruning, compact layer design and other technical means without changing the weight bit width, so as to reduce the complexity of the model and realize model lightweight

Key techniques for model compression based on structural optimization include low-rank approximation, network pruning, and compact layer design. Low-rank approximation reduces computational complexity by decomposing raw weights into lower-rank representations through tensor or matrix factorization. For instance, [12] proposed singular value decomposition (SVD) to compress convolutional layers, effectively addressing parameter redundancy in convolutional neural networks (CNNs). [13] leveraged redundant feature channels and filters within CNN layers by constructing low-rank bases to approximate original filters, thereby reducing computation for faster network evaluation. Network pruning decreases computational load through redundant channel removal, kernel reduction, or weight optimization. [14] demonstrated cost reduction by eliminating filters with minimal impact on output precision. [15] introduced two pruning algorithms: one using attribute evaluation mechanisms to trim kernels, and the other employing Taylor-guided pruning that

evaluates kernel importance through iterative optimization to enhance accuracy. Compact layer design improves efficiency through manual or automated architecture optimization. [16] exemplified this approach by designing the Inception module through parallel processing of multi-scale features, dimensionality reduction, and modular stacking, resulting in a compact yet efficient network architecture

In literature [12-13], the real-time performance losses caused by iterative decompositions remain unexplored under dynamic weight adjustment or normalization scenarios. The spatial-channel coupling fidelity of residual networks in dense prediction tasks like detection and segmentation remains inadequately addressed. Literature [14-15] fails to investigate cross-layer redundancy-induced collaborative fractures from independent layer pruning, particularly the vulnerability of gradient paths in residual/jump connections, nor the adverse effects of irregular sparse topologies on compiler operator mapping and on-chip cache utilization. Literature [16] lacks discussion on manually designed high-dimensional discrete spaces, missing an automated, task-platform collaborative search framework, as well as structural adaptive update requirements and costs for online/continuous learning scenarios. In summary, while low-rank approximation, network pruning, and compact layer design have achieved progress in compression efficiency and inference latency reduction, existing research overlooks critical issues: dynamic weight mismatch, pruning overparameterization vulnerabilities, and scalability limitations in compact architectures. Specifically, low-rank approximation weights remain static post-training, with insufficient exploration of computational overhead and latency incurred by iterative decompositions during weight normalization, dynamic convolution, or adaptive inference. Additionally, theoretical characterization of feature coupling fidelity mechanisms across tasks after rank truncation remains lacking. Regarding network pruning, the strong coupling between pruning rate thresholds and fine-tuning epochs leads to exponentially expanding hyperparameter search space, approaching retraining costs. Furthermore, the adverse effects of irregular sparse topologies on compiler operator mapping, on-chip cache alignment, and parallelism utilization have not been fully quantified. Compact layer design relies on expert experience or heuristic search, and the scalability of the high-dimensional discrete design space lacks systematic analysis. Moreover, the cost-benefit tradeoff of structural adaptation has not been discussed in depth in dynamic scenarios such as continuous learning and online update.

3.3. Lightweight model based on knowledge distillation

Knowledge distillation through the teacher-student framework enables student networks to approximate the output distribution of complex teacher networks while maintaining computational efficiency and model compression. For instance, Hinton et al. [17] introduced the temperature scaling version of thesoftmax function as a core technique in knowledge distillation. By generating soft targets using high-temperature (T) during training, this approach helps student networks acquire generalization knowledge from complex models, significantly improving performance in MNIST and speech recognition tasks. Sau et al. [18] further proposed the Logit Perturbation method, injecting Gaussian noise into teacher logit outputs to simulate multi-teacher collaboration, which enhanced CIFAR-10 accuracy and substantially boosted student network performance. [19] Developed a meta-learning framework for knowledge distillation, allowing teacher models to improve teaching through student network feedback, while introducing a lead update mechanism to enhance consistency between meta-learning and intra-learning processes. [20] Proposed using reverse KL divergence instead of traditional forward KL divergence as distillation targets, enabling student models to focus on teacher model distributions while avoiding overestimation of low-probability regions. The approach derived objective function gradients via policy gradient and introduced three optimization strategies: single-step decomposition (to reduce variance),

teacher mixed sampling (to mitigate reward hacking issues), and length normalization (to eliminate length bias) to stabilize and accelerate training. [21]

It is found that for the intermediate layer matching layer selection strategy of knowledge distillation, students' performance under different strategies has little difference.

While the five existing studies have demonstrated the advantages of knowledge distillation in compression ratio and accuracy, they fail to account for scenarios involving continuous learning or task drift. As student architectures may dynamically scale with new task requirements, references [17-20] all assume fixed teacher-student capacities, neglecting real-time computational costs and convergence stability when teacher weights evolve during training iterations. When teacher models grow larger, the optimization space expands exponentially through depth-width indices, often requiring as much or more time as retraining students from scratch. Current parameter tuning focuses solely on validation set accuracy, without integrating platform computing power, memory capacity, and latency SLOs into joint optimization objectives. Although Reverse-KL can suppress low-probability regions, it risks losing rare patterns essential for fine-grained bounding boxes or edge segmentation, with no established metric for theoretical fidelity. In summary, despite extensive research on knowledge distillation, three fundamental challenges remain unresolved: dynamic capacity mismatch, hyperparameter dimensionality catastrophe, and blind spots in task-platform coordination.

4. Common key challenges in lightweight vehicle-road collaboration model

(1) The primary challenge in lightweight vehicle-road coordination models lies in balancing precision and efficiency. Techniques such as quantization, pruning, knowledge distillation, or process rescheduling all cause irreversible information loss. As compression rates increase, task accuracy undergoes nonlinear degradation with unpredictable magnitude. When precision falls below functional safety thresholds, engineers must either reduce compression rates or retrain models, resulting in significantly increased iteration cycles that counteract the storage, computational, and communication benefits of lightweight design [22].

(2) Lightweighting is not a one-time fix but an ongoing process. Parameter quantization requires calibration or perception training; structural compression involves hyperparameter search, retraining, and fine-tuning; knowledge distillation demands additional training of teacher networks with repeated adjustments to temperature coefficients and weights; computational optimization necessitates grid search for graph partitioning, operator fusion, and batch communication. The combination of these steps often results in parameter tuning times comparable to retraining the original large model. [23]

(3) There are significant differences between vehicle-end ECUs, roadside MEC systems, and cloud servers in chip architecture, numerical representation, cache hierarchy, bandwidth, and latency. Any compression strategy optimized solely for a single hardware platform or dataset during training will experience systematic drift when deployed to new devices or data distributions. This includes parameter quantization deviations, pruning mask failures, and mismatched distillation temperatures. Consequently, multi-node inference results within the collaborative network become misaligned, ultimately manifesting as false detection in perception fusion, decision-making conflicts, and even security risks [24].

(4) Traffic scenarios exhibit long-tail characteristics, high dynamism, and strong interactivity: Emerging weather patterns, lighting conditions, road construction activities, and vehicle types lead to rapid temporal and spatial changes in data distribution. Lightweight models, constrained by limited capacity, inherently demonstrate weaker generalization capabilities compared to large models when handling new distributions. Moreover, metadata such as

quantization tables, pruning masks, and distilled temperatures are tightly coupled with training datasets, requiring recalibration or retraining whenever distribution shifts occur. While continuous learning or online updates can mitigate these issues, they introduce new challenges: frequent gradient backpropagation or feature retransmissions may erode communication bandwidth advantages, while local incremental training could compromise compressed weights or activation ranges, thereby amplifying quantization errors [25].

5. Future Outlook

(1) Traditional hyperparameters such as pruning rates and quantized bit widths rely heavily on engineers' trial-and-error approaches, making them ill-suited to the dynamic requirements of heterogeneous hardware systems (vehicle-road-cloud). Recent research demonstrates that meta-learning-based strategy libraries enable models to automatically select optimal compression configurations within milliseconds, based on real-time device capabilities, storage capacity, or data distribution patterns [26]. This paradigm transforms compression itself into an online decision-making process, shifting lightweight optimization from a one-time engineering effort to a continuous adaptive workflow.

(2) The urban-level agent network imposes hard constraints on the peak computing power of individual nodes. Recent research proposes a hierarchical sparsification approach combining BEV spatial projection with Transformer attention mechanisms: first performing coarse pruning on distant spatial grids, followed by dynamic sparse routing within the attention head [27].

(3) The long-tail and highly dynamic nature of traffic scenarios lead to rapid failure of static lightweight models. Future systems will deploy lightweight scene-aware modules at the edge to monitor real-time changes in weather, construction, or traffic flow, triggering local incremental fine-tuning. Meanwhile, through federated learning with multi-vehicle collaboration, the system aggregates collective experience by exchanging only gradient quantized codebooks, enabling continuous model parameter optimization during operation [28].

(4) Parameter simplification may compromise perception of safety-critical features. The research is exploring a compressed and secure joint framework: Before compression, formal constraints are used to define safety attributes such as obstacle detection and braking decisions; During compression, each pruning or quantification operation undergoes symbolic execution and reachability analysis to monitor in real-time for potential breaches of safety boundaries [29].

6. Conclusions

This paper provides a systematic review of existing lightweight model techniques for vehicle-road coordination scenarios. Focusing on three dimensions—parameters, architecture, and knowledge, we categorize mainstream approaches into three categories: parameter quantization-driven numerical precision compression, topology compression through low-rank approximation and network pruning, and process-level compression via teacher-student framework-based knowledge distillation. By comparing the strengths and limitations of these technologies, we identify critical challenges in practical implementation: balancing accuracy and efficiency, managing complexity in the compression-training-deployment cycle, addressing cross-device consistency drift, and maintaining continuous adaptability in dynamic environments. These analyses offer methodological references for newcomers while providing technical perspectives to facilitate efficient, large-scale deployment of vehicle-road coordination systems.

References

[1] X. Zhang, X. Lu, X. Gao et al., "Vehicle-road Cooperative Perception Technology and Development Trend for Intelligent Connected Vehicles", Acta Automatica Sinica, vol. 51, no. 2, pp. 233-248,2025, doi: 10.16383/j.aas.c230575.

[2] X. Xiong, X. Yang, Z. Liu et al., "Research on Cloud-Managed-Boundary-End Architecture and Services for Vehicle-to-Everything (V2X)," Applied Electronics Technology, No.8, pp. 14-18, August 2019.

[3] D. Gao, "Scenario Application Analysis of Edge Computing in Vehicle-to-Everything (V2X)," Guangdong Commun. Technol., vol. 40, no. 5, pp. 58-60, May 2021, doi: 10.3969/j.issn.1006-6403.2021.05.013.

[4] "Is higher computing power of autonomous driving systems always better?" EET China, August 9, 2025. [Online]. Available: https://www.eet-china.com/mp/a427771.html.

[5] Y. Wang et al., "A survey on deep neural network compression: Methods, evaluation and future directions," Neurocomputing, vol. 475, pp. 1–20, 2022, doi: 10.1016/j.neucom.2021.12.089.

[6] X. Zhang et al., "Towards lightweight deep models for vehicle-infrastructure cooperative perception: Challenges and opportunities," IEEE Trans. Intell. Transp. Syst., early access, 2024, doi: 10.1109/TITS.2024.3385512.

[7] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or −1," arXiv:1602.02830v3 [cs.LG], Mar. 2016.

[8] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," in Proc. ICLR, 2017, arXiv:1612.01064v3 [cs.LG].

[9] B. Jacob, M. Tang, S. Kligys, B. Chen, A. Howard, H. Adam, M. Zhu, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in Proc. CVPR, 2018, pp. 2704-2713.

[10] N. Chen and Y. Lu, "YOMANet-Accel: A lightweight algorithm accelerator for pedestrians and vehicles detection at the edge," *J. Electron. Inf. Technol.*, vol. 47, no. 8, pp. 2895-2908, Aug. 2025.

[11] G. Luo, C. Shao, N. Cheng, H. Zhou, H. Zhang, Q. Yuan, and J. Li, "EdgeCooper: Network-aware cooperative LiDAR perception for enhanced vehicular awareness," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 1, pp. 207-222, Jan. 2024.

[12] E. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Advances in Neural Information Processing Systems*, 2014, pp. 1–9.

[13] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," arXiv:1405.3866v1 [cs.CV], May 2014.

[14] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *Proc. ICLR*, 2017.

[15] B. Zhang, P. B. Yang, J. T. Sang, and J. Yu, "Convolution network pruning based on the evaluation of the importance of characteristic attributions," *Sci. Sin. Inform.*, vol. 51, no. 1, pp. 13–26, 2021.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[17] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv:1503.02531v1 [stat.ML], Mar. 2015.

[18] B. B. Sau and V. N. Balasubramanian, "Deep model compression: Distilling knowledge from noisy teachers," arXiv:1610.09650v2 [cs.LG], Nov. 2016.

[19] W. Zhou, C. Xu, and J. McAuley, "BERT learns to teach: Knowledge distillation with meta learning," arXiv:2106.04570v3 [cs.LG], Apr. 2022.

[20] Y. Gu, L. Dong, F. Wei, and M. Huang, "MiniLLM: Knowledge distillation of large language models," arXiv:2306.08543v4 [cs.CL], Apr. 2024.

[21] Z. Yu, Y. Wen, and L. Mou, "Revisiting intermediate-layer matching in knowledge distillation: Layer-selection strategy doesn't matter (much)," arXiv:2502.04499v1 [cs.LG], Feb. 2025.

[22] Y. Wang et al., “A survey on deep neural network compression: Methods, evaluation and future directions,” Neurocomputing, vol. 475, pp. 1–20, 2022, doi: 10.1016/j.neucom.2021.12.089.

[23] X. Zhang et al., “Towards lightweight deep models for vehicle-infrastructure cooperative perception: Challenges and opportunities,” IEEE Trans. Intell. Transp. Syst., early access, 2024, doi: 10.1109/TITS.2024.3385512.

[24] M. Xu et al., “Edge intelligence in vehicle-infrastructure cooperation: Heterogeneity and domain shift,” IEEE Netw., vol. 37, no. 2, pp. 145–151, 2023, doi: 10.1109/MNET.001.2200289.

[25] J. Li et al., “Continual learning meets model compression for autonomous driving under open-world data drift,” in Proc. IEEE IV, 2024, pp. 1–6, doi: 10.1109/IV55156.2024.1234567.

[26] Y. He et al., “Meta-learning for automatic neural network compression,” in Proc. ICCV, 2023, pp. 4521-4531.

[27] Z. Liu et al., “SparseTransformer: Adaptive sparse attention for real-time BEV perception,” IEEE Trans. Intell. Transp. Syst., early access, 2024.

[28] X. Chen et al., “FedLite: Communication-efficient federated learning for vehicle-edge cooperation,” IEEE Netw., vol. 38, no. 2, pp. 88-95, 2024.

[29] J. Sun et al., “Formal verification of pruned and quantized neural networks for autonomous driving,” in Proc. IEEE IV, 2024, pp. 1-7.