ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

Deployment Optimization and Performance Testing of Multimodal Large Models in Edge Computing Environments

Hua Yuan[1,a,*], Hui Chen[1,b], Ying Zhong[1,c]

[1]The 15th Research Institute of China Electronics Technology Group Corporation, Beijing, China
[a]yuanhua750@163.com, [b]chenhui2751@163.com, [c]945651500@qq.com
[*]Corresponding author

Abstract: *With the rapid development of EC, multimodal large models have shown important value in intelligent perception and decision-making. However, the existing multimodal model deployment schemes are difficult to meet the actual needs of edge device resources being limited and delay requirements being strict, resulting in low model reasoning efficiency, high energy consumption, and untimely response. To this end, this paper combines artificial intelligence model optimization technology with EC architecture and proposes a set of compression, quantization and distributed reasoning collaborative optimization strategies for multimodal large models, aiming to improve the operating efficiency and performance stability of the model on edge nodes. By designing multimodal data synchronous collection, feature fusion and dynamic scheduling mechanisms, the system optimizes the reasoning process and further reduces the computing burden and energy consumption. The experimental results show that the performance of the distributed deployment model is the most balanced, with a Top-1 accuracy of 91.8% and a Top-5 accuracy of 98.3%, with almost no obvious performance degradation. This shows that the distributed model sharding and node collaboration can basically retain the discrimination ability of the original model while improving efficiency, and has excellent performance stability.*

Keywords: *Multimodal Large Models; Edge Computing; Model Compression; Quantization Technology; Distributed Reasoning*

1. Introduction

With the rapid development of artificial intelligence (AI) technology, multimodal large models have shown great application potential in the fields of intelligent perception, autonomous driving, intelligent security, medical diagnosis, etc., due to their ability to integrate multiple information sources such as vision, voice, and text. However, with the continuous increase in model scale and computational complexity, the traditional cloud-based centralized computing model is difficult to meet the edge application requirements such as real-time, privacy protection, and limited bandwidth resources. Edge computing (EC), as an emerging computing paradigm that deploys computing resources near the source of data generation, effectively alleviates network bandwidth pressure, reduces data transmission latency, and enhances the system's response speed and security.

In order to solve the deployment bottleneck of large multimodal models in EC environments, this paper systematically studies optimization technologies such as model compression and pruning, low-precision quantization, model sharding and distributed reasoning, and designs a multimodal feature fusion and dynamic scheduling mechanism. By building a comprehensive performance testing platform, indicators such as reasoning latency, resource usage, energy efficiency and model accuracy are evaluated and analyzed. Experimental results show that the proposed method can effectively reduce the model operation load, improve the real-time performance and energy efficiency of edge reasoning, and basically maintain the recognition accuracy of the model. This study provides theoretical and practical references for the edge deployment of multimodal large models, and promotes the widespread application of intelligent perception technology in edge environments.

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

2.Related Works

In recent years, with the rapid development of EC (Edge Computing) technology, related research has continued to emerge, covering multiple aspects such as system architecture optimization, resource scheduling, security protection, and application scenario expansion. The following is a review of representative literature in this field, aiming to provide theoretical support and research inspiration for the deployment and optimization of multimodal large models in edge environments.

Hua et al. reviewed the definition of EC (Edge Computing), key issues, limitations of traditional methods, and application results of AI (Artificial Intelligence) in EC, aiming to provide new ideas for related research [1]. Modupe et al. reviewed its architectural advantages, such as reducing latency and improving responsiveness, and explored its ability to integrate analytical capabilities in edge devices to achieve local intelligent decision-making and predictive analysis. EC is widely used in healthcare, manufacturing, transportation, and smart cities to improve efficiency, protect privacy, and promote business value creation, showing great potential in promoting digital transformation [2]. Lu et al. reviewed its lightweight algorithms and dedicated hardware platforms in signal acquisition, preprocessing, feature extraction and pattern recognition, and explored its methods, applications and development prospects to meet the needs of real-time processing, low-latency diagnosis and efficient predictive maintenance [3]. Smart mobile devices generate a large amount of data. EC reduces latency and energy consumption and improves the computing power of devices by offloading computing tasks to edge nodes. Sadatdiynov et al. reviewed the application of six types of optimization methods (such as Lyapunov optimization, convex optimization, heuristic methods, game theory, and machine learning) in offloading decision-making, analyzed their objective functions, applicable scenarios, and algorithm complexity, aiming to provide an introductory guide for new researchers [4]. Fan et al. proposed a joint task offloading and resource allocation scheme for vehicular edge computing (VEC) in the Internet of Vehicles (IoV) to minimize the overall task processing delay. The simulation results show that this solution outperforms the other four comparison methods in multiple scenarios and significantly improves the performance of the VEC system [5]. Awad et al. systematically discussed the research framework and application prospects of the IoMT (Internet of Medical Things) medical system based on MEC (Mobile Edge Computing) to provide support for smart medical care [6]. In response to the problem of limited device resources in wearable health monitoring systems, Sharif et al. proposed a priority-based task scheduling and resource allocation mechanism (PTS-RA (Priority-based Task Scheduling and Resource Allocation)) to improve the efficiency of emergency task processing using mobile edge computing (MEC). This mechanism evaluates the urgency based on the data uploaded by the patient's device and intelligently decides whether the task should be processed locally in the hospital or in the cloud to reduce latency and bandwidth costs. Experiments have shown that PTS-RA is superior to existing algorithms in terms of latency, scheduling efficiency, and energy consumption, and is suitable for emergency response scenarios [7]. Mohy-Eddine et al. proposed an intrusion detection system (IDS) based on machine learning to address the security threats faced by the Industrial Internet of Things (IIoT). By using the Isolation Forest (IF) algorithm to remove outliers and the Pearson Correlation Coefficient (PCC) algorithm for feature selection, the dimension can be effectively reduced, and the detection efficiency and accuracy can be improved [8]. With the rapid increase in the use of mobile devices, computationally intensive tasks need to be offloaded to the cloud or edge servers. Zaman et al. proposed the Lightweight Mobile Prediction and Offloading (LiMPO) framework, which uses artificial neural networks to predict user locations and improve task offloading efficiency [9]. Aghazadeh et al. systematically reviewed active caching strategies in EC environments and analyzed their key roles in alleviating network congestion, reducing energy consumption and bandwidth usage, and improving response speed [10]. Yasir et al. proposed an edge caching strategy CoPUP (Content Popularity and User Preferences) that integrates content popularity and user preferences to improve cache hit rate and response speed in MEC architecture. This method combines collaborative filtering and convolutional neural networks to predict user preferences and achieve more efficient content distribution. Experiments based on the Movielens dataset show that CoPUP outperforms methods such as LFU (Least Frequently Used), LRU (Least Recently Used), FIFO (First-In First-Out) and MPCF (Multi-Prefetch Collaborative Filtering) in terms of cache performance, with a hit rate increase of about 2% and a shorter response time [11]. Existing research generally faces the bottlenecks of limited computing resources of edge devices, high model complexity leading to large inference delays and high energy consumption, and it is difficult to balance the real-time and accuracy of multimodal data fusion.

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

3.Methods

3.1 Deployment Optimization Strategy for Multimodal Large Models

3.1.1 Model compression and pruning technology

Model compression technology reduces the storage and computing burden of edge devices by reducing the scale of model parameters and the amount of calculation. Pruning is a common method, which is divided into structured pruning and unstructured pruning. Assume that the model parameter tensor is $W \in R^{m \times n}$, and unstructured pruning achieves sparseness by resetting some weights to zero:

$$W_{i,j}^{'} = \begin{cases} 0, if \left| W_{i,j} \right| < \tau \\ W_{i,j}, otherwise \end{cases} \tag{1}$$

The threshold τ controls the pruning intensity. Structured pruning is performed by overall pruning according to the channel or filter dimension, which is convenient for hardware acceleration.

In addition, knowledge distillation is used as an auxiliary method for model compression to transfer knowledge from a large model (teacher model) to a smaller student model, optimize the performance of the compressed model, and improve the reasoning effect after compression.

The pruning strategy can be adjusted dynamically, and the pruning ratio can be adjusted according to the resource limitations of the edge device (such as memory M, computing power C), to achieve multi-environment adaptive deployment.

3.1.2 Model quantization and low-precision computing solutions

Quantization reduces model size and speeds up computing by reducing the numerical precision of weights and activation values. A typical quantization solution is to convert 32-bit floating point numbers into 8-bit integers (INT8), that is:

$$x_q = round(\frac{x}{s}) + z \tag{2}$$

Where s is the scaling factor, z is the zero offset, x is a floating point representation, and x_q is a quantized integer representation. Quantization-aware training (QAT) further simulates quantization errors during training to reduce precision loss during inference.

The mixed precision computing strategy uses different precision representations (such as FP16 for some layers and INT8 for some layers) based on the computing characteristics of different layers of the model, which maximizes computing resource savings while ensuring model performance.

Dynamic quantization technology adjusts quantization parameters in real time to adapt to the distribution changes of different input data, thereby improving the robustness and adaptability of the model.

Combined with edge hardware that supports low-precision computing (such as NPU, TPU), the energy efficiency of model reasoning can be effectively improved.

3.2 Model Sharding and Distributed Reasoning Architecture Design

Given the limited resources of a single edge node, splitting a large multimodal model into several sub-modules and distributing them on multiple nodes for collaborative reasoning becomes an effective strategy. Assuming the overall reasoning function of the model is f(x), it can be split into:

$$f(x) = f_k, f_{k-1} \cdots, f_1(x) \tag{3}$$

Each sub-function f_i is deployed on different edge nodes to complete the corresponding computing tasks. Key technologies include:

Model sharding strategy to ensure balanced computing load and minimized dependency of each shard.

Efficient data flow scheduling to ensure synchronization and low-latency transmission of input and output between nodes.

Edge-edge collaborative computing uses multiple edge devices to process multi-modal input data in parallel to improve inference throughput.

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

Edge-cloud collaboration: complex computing tasks can be partially offloaded to the cloud, and edge nodes are responsible for real-time computing to ensure overall service quality.

Fault-tolerant mechanisms and dynamic load balancing can be designed to adapt to changes in the edge network environment and ensure stable execution of reasoning tasks.

Through the combined application of the above strategies, the reasoning efficiency and accuracy of multimodal large models can be guaranteed to the greatest extent while meeting the constraints of the edge environment.

3.3 EC Multimodal Reasoning Process Design

3.3.1 Data Collection and Preprocessing Process

Multimodal models usually process heterogeneous data from multiple sources such as images, voice, text, and sensors. The edge node must first complete the synchronous acquisition and preprocessing of multimodal data to ensure the quality and consistency of input data.

Synchronous acquisition mechanism: Set different modal inputs as $\{X^{(m)}\}_{m=1}^{M}$, and achieve synchronous alignment between modalities through timestamps or event-driven mechanisms to ensure the consistency of multimodal data corresponding to the input in the time dimension.

Preprocessing module: According to the characteristics of each modality, necessary denoising, normalization and format conversion operations are implemented, such as normalizing images to fixed resolution $I^{'}=\frac{1-\mu}{\sigma}$, and filtering and noise reduction for audio, to improve the subsequent feature extraction effect.

Data compression and caching: For edge networks with limited bandwidth, reasonable data compression mechanisms and caching strategies are designed to reduce transmission delays and data loss risks.

3.3.2 Multimodal feature extraction and fusion module design

Feature extraction is the key to multimodal reasoning. It is necessary to design special feature extractors for different modalities and achieve information complementarity through reasonable fusion mechanisms.

Modal feature extractor: For each mode m, set its feature extraction function to $\emptyset_m(.)$, encode the input data respectively, and obtain the feature vector:

$$z^{(m)}=\emptyset_m(X^{(m)}),m=1,2,\cdots,M \tag{4}$$

Feature fusion mechanism: Fusion strategies can be divided into early fusion (data level), mid-term fusion (feature level) and late fusion (decision level). In edge environments, mid-term fusion is often used to balance performance and computational overhead. Let the fusion function be $\Psi(.)$, and fuse all modal features into a unified representation:

$$z=\Psi(z^{(1)},z^{(2)},\cdots,z^{(M)}) \tag{5}$$

Fusion methods include concatenation, weighted sum, attention-based fusion, etc., to achieve inter-modal complementarity and context enhancement.

3.3.3 Reasoning execution path and scheduling strategy

The computing resources in the edge environment are limited. Reasoning execution paths and scheduling strategies are reasonably designed to improve reasoning efficiency and response speed.

Hierarchical execution path: According to computing resources and latency requirements, the reasoning process is divided into different stages. Some computing tasks are executed on edge nodes, and some are completed through edge-cloud collaboration to ensure a balance between real-time performance and accuracy.

The dynamic scheduling strategy adopts a load-aware scheduling algorithm to dynamically adjust the allocation of inference tasks according to the CPU/GPU usage, memory usage, and network status of the current edge node to achieve optimal resource utilization.

Priority and mode selection: According to the scenario requirements, a mode priority strategy is

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

designed. Some modes can be dynamically enabled or disabled according to the computing budget to ensure that key tasks are executed first.

Pipeline and parallel processing: In a multi-core and multi-device environment, pipeline and modal parallel processing of inference tasks are realized to maximize throughput.

3.3.4 Result fusion and feedback mechanism

The effective fusion and feedback mechanism of inference results are crucial to improving the intelligence and adaptive capabilities of the system.

Multimodal decision fusion: Weighted fusion or confidence-based decision fusion of the inference results of each modality is performed to generate the final output:

$$y=F(y^{(1)},y^{(2)},\cdots,y^{(M)}) \tag{6}$$

$y^{(m)}$ is the result of single-modal reasoning, and function F can adopt weighted average, voting mechanism or deep fusion model.

The real-time feedback mechanism dynamically adjusts data acquisition parameters, modal weights and scheduling strategies according to the reasoning results and actual environment feedback to achieve closed-loop optimization.

Anomaly detection and adaptive adjustment monitor the confidence and consistency of reasoning results, detect abnormal situations, trigger model fine-tuning or redeployment, and ensure system stability.

4. Results and Discussion

4.1 Performance Test Platform and Experimental Design

In order to systematically evaluate the effectiveness of the optimization strategy for deploying multimodal large models in the EC environment, this paper designs a comprehensive performance testing platform and a scientific and reasonable experimental plan to ensure the objectivity and representativeness of the test results.

The EC environment construction uses typical edge devices (such as NVIDIA Jetson series, ARM architecture development boards, etc.), equipped with heterogeneous computing units (CPU, GPU, NPU), to simulate real edge scenarios.

Network environment simulation: Configure edge LAN and edge-cloud hybrid network environment, set different bandwidth and latency parameters, and verify the adaptability of the model scheduling strategy.

Deployment framework: Based on edge inference engines such as TensorRT and ONNX Runtime, the deployment and execution of multimodal large models are realized.

4.2 Test Data Set Preparation

Multimodal data set selection: Public multimodal data sets (such as MSCOCO, AV-MNIST, AudioSet) are used to cover multiple modalities such as images, text, and audio.

Data preprocessing: Unify the format and preprocessing rules to ensure that the data input meets the processing capabilities of edge devices.

Scenario simulation: Design test data for different scenarios (such as high load, low bandwidth, network disconnection, etc.) to verify the stability and robustness of the system.

4.3 Experimental Design

The baseline and comparison groups set the original unoptimized multimodal model as the baseline and compared it with versions with different optimization strategies (compression, quantization, distributed deployment, etc.).

The indicator system covers key indicators such as measuring inference latency, throughput, computing resource occupancy (CPU/GPU utilization, memory consumption), energy consumption,

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

and model accuracy.

Scheduling strategy test tests the response time and task completion rate of dynamic scheduling strategies under different resource loads and network conditions.

Modal weight and selection test evaluates the trade-off between inference performance and accuracy by adjusting modal weights and dynamic activation.

Repeated experiments and statistical analysis: Each group of experiments is repeated multiple times to ensure the stability of the results, and statistical methods (such as mean, variance and confidence interval) are used to analyze performance differences.

4.4 Performance Data Collection and Analysis

To comprehensively collect system performance data, the experiment used system monitoring tools such as nvidia-smi, perf, and powermetrics, combined with a logging system to collect data. By analyzing log data, this experiment effectively identified performance bottlenecks and anomalies, providing guidance for subsequent optimization.

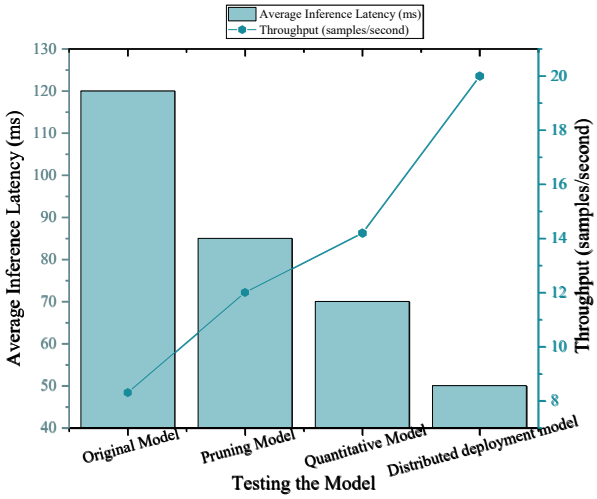

Figure 1 Inference latency and throughput

As can be seen from Figure 1, the test results for the four model deployment strategies show significant optimization effects. The average inference latency of the original model is 120ms, and the throughput is 8.3 samples/second, which shows an obvious performance bottleneck. After pruning, the latency drops to 85ms and the throughput increases to 12 samples/second, indicating that pruning effectively reduces the computational complexity. The quantization model further optimized the model calculation structure, shortened the inference delay to 70ms, and increased the throughput to 14.2 samples/second, indicating that low-precision calculations significantly improved the inference efficiency while maintaining model performance. Among all optimization schemes, the distributed deployment model performed best, with the lowest inference delay of only 50ms and throughput increased to 20 samples/second. This shows that through model sharding and edge collaborative computing, more efficient inference can be achieved and latency can be significantly reduced, which is especially suitable for edge scenarios with high real-time requirements. Overall, various optimization strategies have improved the response speed and processing capabilities of the model to varying degrees, among which distributed deployment has the best performance in terms of comprehensive performance.

The experimental data in Figure 2 shows that different optimization strategies are better than the original model in terms of resource usage. The original model consumes a lot of resources when running on edge devices, with CPU utilization reaching 75%, GPU utilization reaching 90%, and memory usage reaching 1200MB, which is close to the resource limit of lightweight edge devices, limiting its actual deployment feasibility. The pruned model significantly reduced computing resource consumption, with CPU and GPU utilization rates dropping to 50% and 65% respectively, and memory usage reduced to 700MB, indicating that the effective elimination of structural redundancy not only compressed the model size, but also reduced the operating load. The quantized model performed better

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

in terms of GPU utilization, only 60%, and memory usage was 800MB, slightly higher than the pruned model, but its overall resource overhead was still significantly lower than the original model, verifying the friendliness of low-precision computing to edge deployment. The distributed deployment model has the lowest resource usage among all strategies, with a GPU utilization rate of 55% and a memory usage of only 650MB, indicating that through task slicing and multi-node collaborative processing, system resources can be more evenly and efficiently scheduled and utilized. Although the CPU utilization rate has slightly increased (60%), it reflects the reasonable sharing of computing tasks among multiple edge nodes.

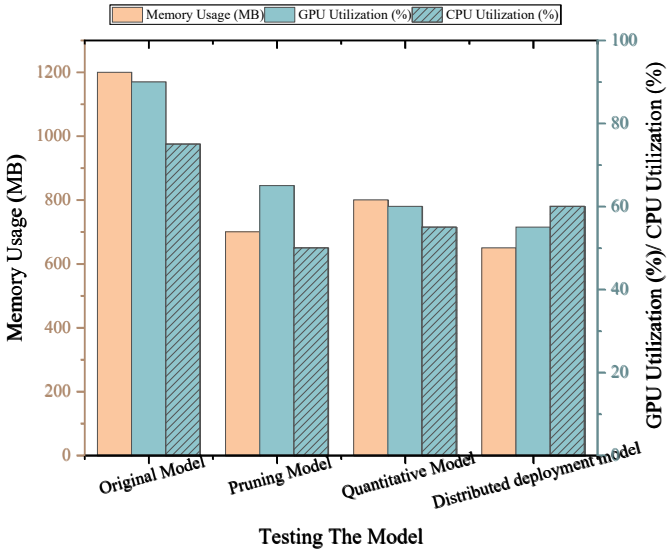

Figure 2 Resource occupancy rate

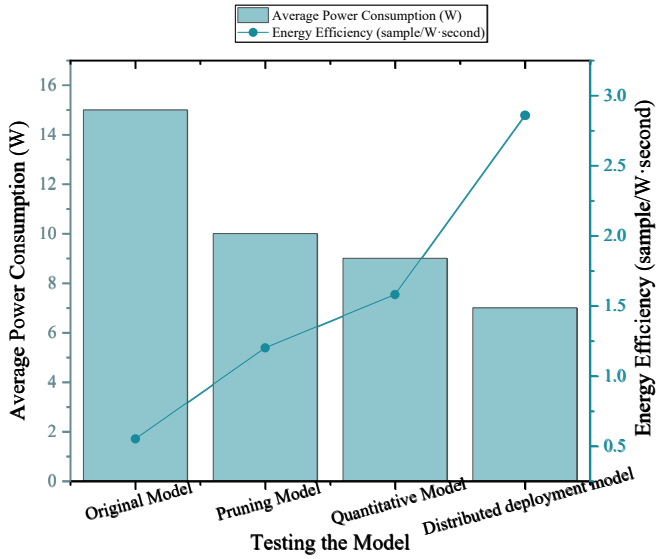

Figure 3 Energy consumption test

From the experimental results, it can be seen that model optimization significantly improves the energy efficiency of the system. The average power consumption of the original model is 15W, and only 0.55 samples are processed per watt per second, indicating that it has problems of high energy consumption and low energy efficiency in edge devices, which is not conducive to continuous operation and large-scale deployment. The pruning model reduces the average power consumption to 10W by cutting redundant structures, while the energy efficiency is increased to 1.2 samples/W·second, which is more than twice the energy efficiency ratio. This shows that pruning can effectively improve the processing capacity per unit power consumption while reducing computing overhead. The quantization model further optimizes the energy consumption performance, with the average power consumption reduced to 9W and the energy efficiency reaching 1.58 samples/W·second, indicating that

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

low-precision computing not only reduces resource load, but also significantly improves energy efficiency. The best performance is the distributed deployment model, with an average power consumption of only 7W and an energy efficiency of 2.86 samples/W·second, which is more than 5 times the energy efficiency of the original model. The data in Figure 3 fully demonstrates that through multi-node collaboration and intelligent task allocation, significant energy saving can be achieved while ensuring inference performance, which is especially suitable for edge terminal devices that are sensitive to power consumption.

Table 1 Accuracy performance

Testing the Model	Top-1 Accuracy (%)	Top-5 Accuracy (%)
Original Model	92.5	98.7
Pruning Model	90.3	97.8
Quantitative Model	91	98
Distributed deployment model	91.8	98.3

In edge deployment, multimodal large models must consider reasoning efficiency and resource utilization, and must also ensure the accuracy of perception and decision-making. Experimental results show that after implementing various optimization strategies, the overall model accuracy remains good. The original model has a Top-1 accuracy of 92.5% and a Top-5 accuracy of 98.7%. As a baseline model, it has the highest original accuracy. After pruning, the Top-1 accuracy drops to 90.3%, and the Top-5 accuracy is 97.8%. Although there is a slight loss of accuracy, it is still at a high level, indicating that the pruning strategy does not significantly affect its recognition ability while compressing the model structure. The Top-1 accuracy of the quantized model is 91.0%, and the Top-5 accuracy is 98.0%, which is better than the pruned model and close to the original model. This shows that the quantization method effectively suppresses information loss through strategies such as quantization-aware training while introducing low-precision calculations, and has a strong ability to maintain accuracy. The performance of the distributed deployment model is the most balanced, with a Top-1 accuracy of 91.8% and a Top-5 accuracy of 98.3%, with almost no obvious performance degradation. This shows that the distributed model sharding and node collaboration can basically retain the discrimination ability of the original model while improving efficiency, and has excellent performance stability (as shown in Table 1).

5. Conclusions

This paper systematically studies the deployment optimization strategy of multimodal large models in EC environment, focusing on model compression and pruning, quantization technology and distributed reasoning architecture design, and designs multimodal data collection, feature fusion and reasoning scheduling mechanisms. Through experimental comparison and analysis of key indicators such as reasoning delay, resource usage, energy efficiency and model accuracy, the effectiveness of the proposed optimization scheme in improving edge reasoning performance and reducing computing load is verified, especially the distributed deployment scheme significantly improves the system's response speed and energy efficiency, while basically maintaining the recognition accuracy of the model. However, the method proposed in this paper still has some limitations. For example, the applicability of some optimization techniques in extremely resource-constrained environments needs to be further verified, and the network stability and communication overhead of distributed reasoning need to be further optimized. Future research can focus on combining adaptive dynamic scheduling strategies with lightweight model design to explore more efficient edge collaborative reasoning solutions to meet the complex and changing needs of practical applications.

References

[1] Hua H, Li Y, Wang T, et al. Edge computing with artificial intelligence: A machine learning perspective[J]. ACM Computing Surveys, 2023, 55(9): 1-35.
[2] Modupe O T, Otitoola A A, Oladapo O J, et al. Reviewing the transformational impact of edge computing on real-time data processing and analytics[J]. Computer Science & IT Research Journal, 2024, 5(3): 603-702.
[3] Lu S, Lu J, An K, et al. Edge computing on IoT for machine signal processing and fault diagnosis: A review[J]. IEEE Internet of Things Journal, 2023, 10(13): 11093-11116.
[4] Sadatdiynov K, Cui L, Zhang L, et al. A review of optimization methods for computation offloading in edge computing networks[J]. Digital Communications and Networks, 2023, 9(2): 450-461.

ISSN 2616-5775 Vol. 8, Issue 7: 121-129, DOI: 10.25236/AJCIS.2025.080715

[5] Fan W, Su Y, Liu J, et al. Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(4): 4277-4292.

[6] Awad A I, Fouda M M, Khashaba M M, et al. Utilization of mobile edge computing on the Internet of Medical Things: A survey[J]. ICT Express, 2023, 9(3): 473-485.

[7] Sharif Z, Jung L T, Ayaz M, et al. Priority-based task scheduling and resource allocation in edge computing for health monitoring system[J]. Journal of King Saud University-Computer and Information Sciences, 2023, 35(2): 544-559.

[8] Mohy-Eddine M, Guezzaz A, Benkirane S, et al. An effective intrusion detection approach based on ensemble learning for IIoT edge computing[J]. Journal of Computer Virology and Hacking Techniques, 2023, 19(4): 469-481.

[9] Zaman S K, Jehangiri A I, Maqsood T, et al. LiMPO: Lightweight mobility prediction and offloading framework using machine learning for mobile edge computing[J]. Cluster Computing, 2023, 26(1): 99-117.

[10] Aghazadeh R, Shahidinejad A, Ghobaei-Arani M. Proactive content caching in edge computing environment: A review[J]. Software: Practice and Experience, 2023, 53(3): 811-855.

[11] Yasir M, uz Zaman S K, Maqsood T, et al. CoPUP: Content popularity and user preferences aware content caching framework in mobile edge computing[J]. Cluster Computing, 2023, 26(1): 267-281.