SPAREDGE: REDUNDANCY-AWARE DYNAMIC SPARSE INFERENCE WITH EDGE-CLOUD COLLABORATION FOR MULTIMODAL LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal large language models (MLLMs) excel at cross-modal understanding and generation, yet their massive parameter size and heavy computation make inference prohibitively expensive, especially on resource-constrained edge devices. Cloud-only inference offers strong compute but suffers from high latency and bandwidth pressure. Existing acceleration schemes largely rely on static pruning or fixed sparsity patterns, and thus fail to adapt to the dynamically changing redundancy distributions in multimodal inputs. To address this, we propose SparEdge, a redundancy-aware dynamic sparse inference framework with edge-cloud collaboration for accelerating MLLM inference. First, a lightweight cross-modal redundancy-aware mechanism efficiently characterizes input-dependent redundancy across visual and textual representations with minimal additional overhead, enabling dynamic identification of redundant computation during inference. Second, an edge-cloud collaborative dynamic sparse inference mechanism translates the detected redundancy into adaptive sparsity decisions and coordinates computation between edge and cloud resources according to the runtime sparsity characteristics and resource conditions. Experiments on multiple datasets show that SparEdge improves inference throughput by more than 1.8× while maintaining accuracy, and reduces computational and memory overhead by more than 25.2%.

1 INTRODUCTION

Multimodal large language models (MLLMs) have demonstrated remarkable capabilities in cross-modal understanding and generation, enabling a wide range of applications such as visual question answering, multimodal assistants, and embodied intelligence Caffagni et al. (2024); Wu et al. (2023). However, their large parameter scale and computation-intensive inference incur substantial latency, memory, and energy overhead Li et al. (2025), making efficient deployment particularly challenging on resource-constrained edge devices Yang et al. (2026). Cloud-only inference can leverage powerful computing resources, but frequently suffers from communication latency, bandwidth pressure, and privacy concerns caused by transmitting multimodal data Tian & Yang (2026); Yang et al. (2024). In contrast, fully local inference avoids excessive communication but is severely constrained by the limited computation and memory capacity of edge devices. Edge-cloud collaborative inference therefore provides a promising solution by combining the responsiveness of edge devices with the computational capability of the cloud Huang et al. (2026).

Nevertheless, existing acceleration methods remain inadequate for efficient MLLM inference in heterogeneous edge-cloud environments Gui et al. (2026); Hu et al. (2026). Most pruning and sparse inference approaches rely on static pruning ratios or predefined sparsity patterns, making them unable to adapt to the highly dynamic redundancy distributions of multimodal inputs Kuang et al. (2025); Bai et al. (2025). In particular, the importance of visual representations often depends on their semantic interaction with textual queries, implying that redundancy is inherently input- and cross-modal-dependent WANG et al. (2026). Meanwhile, existing edge-cloud inference methods mainly determine computation partitioning according to model structure or resource conditions, without considering how runtime sparsity reshapes the computational workload Yuan et al. (2026);

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Chen et al. (2026b). Consequently, redundancy identification, sparse execution, and edge-cloud scheduling are largely optimized in isolation, limiting their potential efficiency gains.

To address these challenges, we propose SparEdge, a redundancy-aware dynamic sparse inference framework with edge-cloud collaboration for efficient MLLM inference. By jointly exploiting cross-modal redundancy and heterogeneous computing resources, SparEdge significantly reduces unnecessary computation while maintaining model accuracy. Our main contributions are summarized as follows:

- We propose SparEdge, a redundancy-aware edge-cloud collaborative inference framework that jointly considers cross-modal redundancy, dynamic sparsity, and heterogeneous resource characteristics for efficient MLLM inference.
- We design a lightweight cross-modal redundancy-aware mechanism and an edge-cloud collaborative dynamic sparse inference mechanism, enabling input-adaptive redundant computation identification and sparsity-aware computation coordination with low runtime overhead.
- We conduct extensive experiments on multiple MLLMs and multimodal benchmarks. Results show that SparEdge improves inference throughput by more than 1.8× while maintaining accuracy, and reduces computational and memory overhead by more than 25.2%.

2 Related Work

Dynamic Sparse Inference. Efficient inference has become an important research direction for deploying MLLMs under limited computational resources Zeng et al. (2026b); Czakó et al. (2025). Existing approaches mainly reduce inference cost through model pruning, token reduction, sparse attention, quantization, and conditional computationZeng et al. (2026b); Czakó et al. (2025); Zhang et al. (2025). Structured and unstructured pruning methods remove redundant parameters or computation paths according to predefined importance criteria, while token pruning methods progressively discard less informative visual or textual tokens during inference Zhong et al. (2025); Yi et al. (2025); Ye et al. (2025). Recent studies further explore dynamic token selection and input-adaptive computation for vision-language models, demonstrating that substantial redundancy exists in multimodal representations and can be exploited to accelerate inference Chen et al. (2026a); Wang et al. (2025); Zeng et al. (2026a). However, most existing methods determine sparsity based on single-modal saliency, fixed pruning ratios, or predefined sparsity patterns, making them difficult to adapt to the highly input-dependent redundancy of multimodal inputs. Moreover, existing sparse inference methods mainly focus on reducing computation itself, without considering how dynamically changing sparsity affects execution across heterogeneous computing resources.

Edge-Cloud Collaborative Inference. Edge-cloud collaboration provides another promising direction for accelerating large-model inference by jointly exploiting the low-latency response of edge devices and the abundant computational resources of cloud servers Qu et al. (2025); Zheng et al. (2025). Early studies mainly partition neural networks at fixed layers and offload computation between edge and cloud according to communication latency, bandwidth, and device capability Singhal et al. (2025); Zhao et al. (2026). More recent approaches introduce adaptive partitioning, dynamic offloading, and resource-aware scheduling to cope with fluctuating network conditions and heterogeneous hardware resources Wang et al. (2026); Lin et al. (2026). Similar ideas have also been explored for transformer and large-model inference, where different model components, layers, or computation stages are distributed across edge and cloud platforms Park et al. (2025); Patel et al. (2024); He et al. (2026). Despite these advances, existing approaches generally assume that the computational workload of a model remains relatively stable for a given architecture, and therefore make partitioning decisions primarily according to model structure and system-level resource conditions. They rarely account for input-dependent sparsity, where different multimodal samples may activate substantially different amounts of computation.

3 Motivation Analysis

We conduct a series of profiling experiments using Qwen3.5-4B on the MMBench benchmark. Our edge platform is equipped with an Intel Xeon Gold 6330 CPU, where Qwen3.5-4B is deployed in

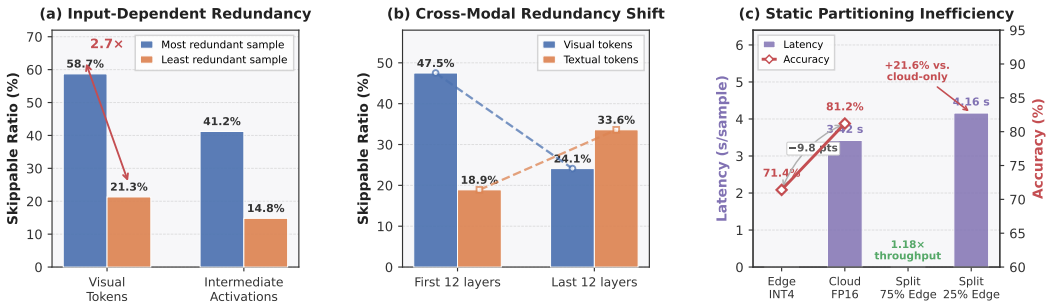

Figure 1: Motivation analysis of SparEdge. (a) and (b) reveal input-dependent and cross-modal redundancy shifts, while (c) shows that static edge-cloud partitioning is inefficient.

INT4 format to fit the limited computation and memory budget. The cloud platform consists of two NVIDIA A100 80GB GPUs, running the same model in FP16. As illustrated in Figure 1, based on the profiling results, we make the following observations.

Observation 1: Redundancy in Multimodal Inputs Is Highly Input-Dependent. We first examine whether redundant computation in MLLMs varies across inputs. For each MMBench sample, we measure the layer-wise activation sparsity and the fraction of visual tokens whose removal does not change the predicted answer. We observe that the redundancy distribution differs substantially across samples and across layers. On visually simple samples, up to 58.7% of visual tokens and 41.2% of intermediate activations can be skipped with less than 0.5% accuracy loss, whereas on samples requiring fine-grained visual reasoning, the skippable fraction drops to 21.3% and 14.8%, respectively. The gap between the most and least redundant samples reaches 2.7× in terms of skippable visual tokens. This confirms that a single static sparsity pattern cannot fit all inputs: aggressive static sparsity degrades accuracy on hard samples, while conservative static sparsity leaves substantial computation unused on easy samples.

Observation 2: Redundancy Is Cross-Modal and Not Uniformly Distributed. We further analyze redundancy separately for visual and textual representations. The results show that visual tokens exhibit much higher redundancy than textual tokens in early and middle layers, while textual redundancy becomes more prominent in deeper layers and during later decoding steps. Specifically, in the first 12 layers, the average skippable ratio of visual tokens is 47.5%, compared with 18.9% for textual tokens; in the last 12 layers, the trend reverses, with textual skippable ratios rising to 33.6% while visual ratios fall to 24.1%. Importantly, the two are correlated but not identical: pruning visual tokens aggressively shifts redundancy to textual pathways, increasing textual skippable ratios by 6.8–11.4 percentage points, and vice versa. This indicates that redundancy should be characterized jointly across modalities rather than per modality in isolation, and that sparsity decisions must be made online to track this shifting distribution.

Observation 1: Edge-Cloud Heterogeneity Makes Static Partitioning Inefficient.

We then study the edge-cloud setting. Running the full INT4 model on the edge achieves only 71.4% accuracy on MMBench, which is 9.8 points lower than the FP16 cloud model at 81.2%, while running the full FP16 model on the cloud incurs an average end-to-end latency of 3.42 s per sample and transmits 2.7 MB of visual tokens and intermediate representations per query. A fixed partition of the model between edge and cloud performs poorly across samples: a split tuned for easy inputs (edge handles 75% of layers) wastes cloud compute and yields only 1.18× throughput improvement, while a split tuned for hard inputs (edge handles 25% of layers) overloads the edge and increases average latency to 4.16 s, a 21.6% regression over cloud-only inference. These results show that the optimal split depends on the runtime sparsity characteristics of the input and on the resource conditions of both sides, and thus must be determined dynamically.

These observations motivate the design of SparEdge. By introducing a lightweight cross-modal redundancy-aware mechanism, the framework can capture input-dependent redundancy across visual and textual representations online while preserving effective cross-modal interaction. With edge-cloud collaborative dynamic sparse inference, the runtime sparsity decisions are translated

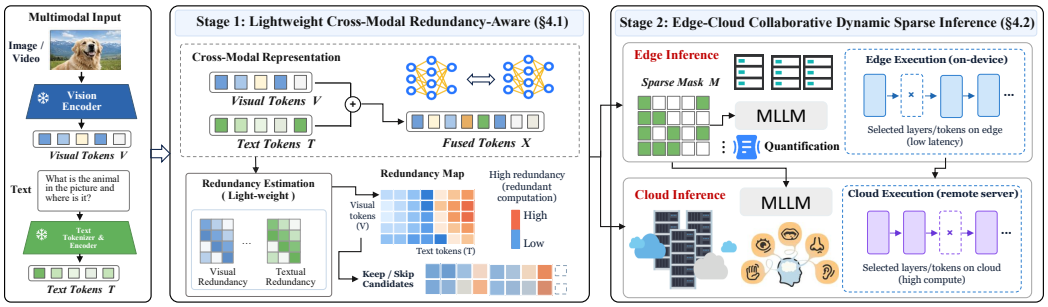

Figure 2: The overview of the SparEdge framework.

into adaptive computation allocation that concentrates resources on the most informative parameters and activation paths, improving inference efficiency while maintaining accuracy.

4 SparEdge Design

We propose SparEdge, a redundancy-aware dynamic sparse inference framework with edge-cloud collaboration for efficient multimodal large language model inference. As illustrated in in Figure 2, SparEdge consists of two tightly coupled modules. First, Lightweight Cross-Modal Redundancy-Aware (§4.1) efficiently characterizes input-dependent redundancy across visual and textual representations with minimal additional overhead, enabling dynamic identification of redundant computation during inference. Second, Edge-Cloud Collaborative Dynamic Sparse Inference (§4.2) converts the detected redundancy into dynamic sparsity decisions and adaptively coordinates the resulting computation between edge and cloud resources according to runtime sparsity and system conditions. Through this two-stage design, SparEdge transforms cross-modal redundancy into executable sparse computation and further leverages heterogeneous resources to achieve efficient MLLM inference while preserving model accuracy.

4.1 Lightweight Cross-Modal Redundancy-Aware

The computational redundancy of multimodal inputs is highly input-dependent Shi et al. (2026). In particular, a visual token that is informative for one textual query can become redundant for another, making fixed token pruning or unimodal saliency estimation suboptimal Yu et al. (2025). The goal of this module is to estimate, at inference time and with minimal overhead, which computations in the MLLM are redundant for the current multimodal input. Instead, we design a lightweight redundancy-aware mechanism that operates on compact token representations and produces a redundancy score for each token at each layer, jointly across visual and textual modalities.

Consider an MLLM with L transformer layers. At layer l, let $\mathbf{H}^{(l)} = [\mathbf{H}_v^{(l)}; \mathbf{H}_t^{(l)}] \in \mathbb{R}^{N \times d}$ denote the hidden states of the multimodal sequence, where $\mathbf{H}_v^{(l)} \in \mathbb{R}^{N_v \times d}$ and $\mathbf{H}_t^{(l)} \in \mathbb{R}^{N_t \times d}$ are the visual and textual token representations, respectively, and $N = N_v + N_t$. Our objective is to learn a scoring function $\mathbf{s}^{(l)} \in [0,1]^N$ such that $s_i^{(l)}$ reflects the redundancy of token i at layer l, i.e., the degree to which its computation can be skipped without affecting the output. The challenge is to compute $\mathbf{s}^{(l)}$ with negligible cost while capturing cross-modal interactions. Directly scoring tokens from full hidden states is expensive and entangles modality-specific and cross-modal factors. We therefore first project each modality into a compact redundancy subspace:

$$\mathbf{Z}_m^{(l)} = \phi_m\left(\mathbf{H}_m^{(l)} \mathbf{W}_m^{(l)}\right), \quad m \in \{v, t\}, \tag{1}$$

where $\mathbf{W}_m^{(l)} \in \mathbb{R}^{d \times r}$ with $r \ll d$ is a lightweight projection, and $\phi_m(\cdot)$ is a normalization function. This decoupling reduces the scoring cost from $\mathcal{O}(Nd)$ to $\mathcal{O}(Nr)$ and prevents one modality from dominating the redundancy estimate of the other. Redundancy in MLLMs is not modality-isolated: as shown in our motivation analysis, aggressive pruning of visual tokens shifts redundancy to textual pathways. To capture this coupling, we introduce a lightweight cross-modal attention that lets each

modality query the other in the compact subspace:

$$\mathbf{C}_v^{(l)} = \mathrm{Softmax}\left(\frac{\mathbf{Z}_v^{(l)}\mathbf{Z}_t^{(l)\top}}{\sqrt{r}}\right)\mathbf{Z}_t^{(l)}, \quad \mathbf{C}_t^{(l)} = \mathrm{Softmax}\left(\frac{\mathbf{Z}_t^{(l)}\mathbf{Z}_v^{(l)\top}}{\sqrt{r}}\right)\mathbf{Z}_v^{(l)}. \tag{2}$$

The interaction matrices are of size $N_v \times N_t$, which is small in practice since r is small and the attention is computed in the reduced subspace. The resulting cross-modal contexts $\mathbf{C}_v^{(l)}$ and $\mathbf{C}_t^{(l)}$ summarize how much each token's modality is explained by the other modality, providing a direct signal for redundancy. We fuse the modality-specific and cross-modal representations into a per-token redundancy score:

$$s_i^{(l)} = \sigma\Big(\mathbf{w}_s^\top \Big[\mathbf{Z}_{m(i)}^{(l)}\Big]_i + \mathbf{u}_s^\top \Big[\mathbf{C}_{m(i)}^{(l)}\Big]_i + b_s\Big), \tag{3}$$

where $m(i)$ denotes the modality of token i, $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{w}_s, \mathbf{u}_s \in \mathbb{R}^r$, $b_s \in \mathbb{R}$ are learnable parameters shared across layers. A high score indicates that the token is well explained by its modality and by cross-modal context, and is therefore a candidate for sparsification. To avoid layer-wise inconsistency, we further smooth the scores across adjacent layers with an exponential moving average:

$$\tilde{s}_i^{(l)} = \alpha\, s_i^{(l)} + (1-\alpha)\, \tilde{s}_i^{(l-1)}, \tag{4}$$

where $\alpha \in (0,1)$ controls the smoothing strength. This temporal smoothing stabilizes sparsity decisions during autoregressive decoding and prevents oscillatory token dropping. Given the smoothed scores, we convert them into a binary sparsity mask under a target budget $\rho^{(l)} \in [0,1)$ that specifies the fraction of tokens to retain at layer l:

$$\mathcal{M}^{(l)} = \mathrm{TopK}\Big(\tilde{\mathbf{s}}^{(l)}, \Big\lceil \rho^{(l)} N \Big\rceil\Big), \qquad \mathbf{H}^{(l)} \leftarrow \mathbf{H}^{(l)} \odot \mathcal{M}^{(l)}. \tag{5}$$

The budget $\rho^{(l)}$ is not fixed, which is produced by the second module according to runtime resource conditions, which closes the loop between redundancy awareness and edge-cloud execution. To keep the scoring mechanism lightweight, all projections and scoring parameters are trained with a small auxiliary loss that encourages score–accuracy consistency:

$$\mathcal{L}_{\text{red}} = \sum_{l=1}^{L} \Big\| \mathrm{Acc}\Big(\mathcal{M}^{(l)}\Big) - \mathrm{Acc}(\mathbf{1}) \Big\|_2^2 + \lambda \sum_{l=1}^{L} \Big\| \tilde{\mathbf{s}}^{(l)} \Big\|_1, \tag{6}$$

where $\mathrm{Acc}(\cdot)$ denotes the task accuracy under the corresponding mask and λ balances the accuracy-preservation and sparsity-encouragement terms. A detailed theoretical proof is in Appendix C. In practice, we approximate the first term with a differentiable surrogate based on logit perturbation, avoiding repeated full evaluations.

4.2 Edge-Cloud Collaborative Dynamic Sparse Inference

In this module, we translate the detected redundancy into adaptive sparsity decisions and coordinate the resulting computation between edge and cloud resources. Because the redundancy distribution shifts across layers and decoding steps, the partition must be recomputed online rather than fixed in advance, and because the edge and cloud differ substantially in compute and memory capacity Xu et al. (2025), the optimal placement of a given layer depends on how sparse that layer has become Xu et al. (2024). We model the per-layer inference overhead on the edge and the cloud. Let $c_e^{(l)}$ and $c_c^{(l)}$ denote the compute overhead of executing layer l on the edge and cloud, respectively, and let $m_e^{(l)}$ and $m_c^{(l)}$ denote the corresponding memory overhead. Let $b^{(l)}$ denote the communication overhead of transferring the intermediate activations of layer l across the link. Given a sparsity budget $\rho^{(l)}$, the effective overhead is scaled by the retained token ratio:

$$c_e^{(l)}(\rho^{(l)}) = \rho^{(l)} c_e^{(l)}, \qquad c_c^{(l)}(\rho^{(l)}) = \rho^{(l)} c_c^{(l)}, \qquad b^{(l)}(\rho^{(l)}) = \rho^{(l)} b^{(l)}. \tag{7}$$

This model reflects the fact that sparser computation reduces both local compute and memory overhead as well as the volume of activations that must be transmitted. The memory terms $m_e^{(l)}$ and $m_c^{(l)}$ are used as hard constraints rather than soft penalties, since exceeding the edge memory budget would make execution infeasible. Let $x^{(l)} \in \{0,1\}$ indicate whether layer l is executed on the

Algorithm 1: Edge-Cloud Collaborative Dynamic Sparse Inference

Input: Redundancy scores $\{\tilde{\mathbf{s}}^{(l)}\}_{l=1}^{L}$; compute overhead $\{c_e^{(l)}, c_c^{(l)}\}_{l=1}^{L}$; memory overhead $\{m_e^{(l)}, m_c^{(l)}\}_{l=1}^{L}$; communication overhead $\{b^{(l)}\}_{l=1}^{L}$; global sparsity budget B; memory budget M; lower bound $\rho_{\min}$.

Output: Sparsity masks $\{\mathcal{M}^{(l)}\}_{l=1}^{L}$ and partition $\mathbf{x} = [x^{(1)}, \ldots, x^{(L)}]$.

for $l \leftarrow 1$ **to** L **do**
  $\bar{s}^{(l)} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \tilde{s}_i^{(l)}$ `/* mean redundancy */`
  $\rho^{(l)} \leftarrow \mathrm{clip}\Big(\rho_{\min} + (1-\rho_{\min}) \cdot \frac{\bar{s}^{(l)}}{\max_{l'} \bar{s}^{(l')}}, \rho_{\min}, 1\Big)$
while $\sum_l (1-\rho^{(l)}) > B$ **do**
  $l^\star \leftarrow \arg\max_l \rho^{(l)}$ `/* reduce the least aggressive layer */`
  $\rho^{(l^\star)} \leftarrow \max(\rho_{\min}, \rho^{(l^\star)} - \Delta)$
for $l \leftarrow 1$ **to** L **do**
  $\mathcal{M}^{(l)} \leftarrow \mathrm{TopK}\big(\tilde{\mathbf{s}}^{(l)}, \lceil \rho^{(l)} N \rceil\big)$
Initialize DP table $\mathcal{D}[0, \cdot] \leftarrow 0$;
for $l \leftarrow 1$ **to** L **do**
  forall *states* $x^{(l)} \in \{0,1\}$ *with* $x^{(l)} m_e^{(l)}(\rho^{(l)}) + (1-x^{(l)}) m_c^{(l)}(\rho^{(l)}) \le M$ **do**
    $\mathcal{D}[l, x^{(l)}] \leftarrow \min_{x^{(l-1)}} \Big(\mathcal{D}[l-1, x^{(l-1)}] + x^{(l)} c_e^{(l)}(\rho^{(l)}) + (1-x^{(l)}) c_c^{(l)}(\rho^{(l)}) + \mathbb{1}[x^{(l-1)} \neq x^{(l)}]\, b^{(l-1)}(\rho^{(l-1)})\Big)$
Backtrack $\mathbf{x}$ from $\mathcal{D}[L, \cdot]$;
return $\{\mathcal{M}^{(l)}\}_{l=1}^{L}, \mathbf{x}$

edge ($x^{(l)} = 1$) or on the cloud ($x^{(l)} = 0$). We seek a per-layer sparsity budget $\rho^{(l)}$ and a partition $\mathbf{x} = [x^{(1)}, \ldots, x^{(L)}]$ that minimize the end-to-end latency subject to compute and memory constraints:

$$\min_{\boldsymbol{\rho}, \mathbf{x}} \; \sum_{l=1}^{L} \Big[x^{(l)} c_e^{(l)}(\rho^{(l)}) + (1-x^{(l)}) c_c^{(l)}(\rho^{(l)}) \Big] + \sum_{l=1}^{L-1} \mathbb{1}\Big[x^{(l)} \neq x^{(l+1)}\Big]\, b^{(l)}(\rho^{(l)}), \tag{8}$$

$$\text{s.t.} \quad \sum_{l=1}^{L} \Big(1 - \rho^{(l)}\Big) \le B, \qquad x^{(l)} m_e^{(l)}(\rho^{(l)}) + (1-x^{(l)}) m_c^{(l)}(\rho^{(l)}) \le M, \qquad \rho^{(l)} \in [\rho_{\min}, 1], \quad \forall l, \tag{9}$$

where B is the global sparsity budget derived from the redundancy scores, M is the memory budget of the target device, and $\rho_{\min}$ is a lower bound that prevents over-aggressive pruning. The second term in Eq. (8) penalizes partition switches, since each switch incurs a communication overhead; this encourages contiguous execution segments and avoids excessive edge–cloud handoffs. The constraints in Eq. (9) ensure that the resulting plan respects both the sparsity budget and the memory capacity of the assigned device. Layers with higher average redundancy receive more aggressive sparsity:

$$\rho^{(l)} = \mathrm{clip}\left(\rho_{\min} + (1-\rho_{\min}) \cdot \frac{\bar{s}^{(l)}}{\max_{l'} \bar{s}^{(l')}},\ \rho_{\min}, 1\right), \tag{10}$$

where $\bar{s}^{(l)} = \frac{1}{N}\sum_{i=1}^{N} \tilde{s}_i^{(l)}$ is the mean smoothed redundancy score at layer l. The overhead coefficients $c_e^{(l)}$, $c_c^{(l)}$, and $b^{(l)}$ vary with runtime conditions such as available bandwidth, edge load, and cloud queueing. Because the objective is separable across layers once $\boldsymbol{\rho}$ is fixed, we maintain the optimal latency for each possible execution state while enforcing the memory constraint at each step. The resulting partition is re-evaluated periodically during decoding to adapt to fluctuating conditions. We summarize the full procedure in Algorithm 1. The algorithm has complexity $\mathcal{O}(L)$ for budget allocation and $\mathcal{O}(L)$ for the dynamic program over two states, so it adds negligible overhead relative to the forward pass. A detailed convergence analysis is in Appendix D. By

coupling redundancy scores with resource-aware partitioning, SparEdge ensures that sparsity is not only input-adaptive. Computation is placed where it is cheapest under the current sparsity profile, and the sparsity profile is in turn shaped by the available edge and cloud capacity.

5 Experimental Evaluation

5.1 Evaluation Setup

Datasets and Implementation Details. We evaluate SparEdge on three representative multimodal benchmarks, including TextVQA Singh et al. (2019), VQAv2 Goyal et al. (2017), and MMBench Liu et al. (2024b), to comprehensively assess its performance across text-rich visual question answering, general vision-language understanding, and multimodal reasoning scenarios. To verify the scalability and generalization ability of the proposed method, we conduct experiments on four backbone models with different parameter scales and architectures, including Qwen3.5-4B Qwen Team (2026), LLaVA-1.5-7B Liu et al. (2024a), InternVL3-8B Zhu et al. (2025), and Qwen3.5-9B Qwen Team (2026). The experiments are implemented using PyTorch and conducted on an edge–cloud testbed: the edge side is an Intel Xeon Gold 6330 server running an INT4-quantized model, and the cloud side is a server equipped with two NVIDIA A100 GPUs with 80 GB of memory each running the FP16 model. For more implementation details, see Appendix A. An analysis of the tradeoff between sparsity, accuracy, and throughput is provided in Appendix F

Baseline Methods. We compare our proposed framework against the following baseline approaches. (1) **SparseVLM** Zhang et al. (2025): A token sparsity method that accelerates vision-language models by progressively pruning redundant visual tokens during inference. (2) **SpecEdge** Park et al. (2025): An edge–cloud collaborative inference framework that accelerates LLM serving via speculative decoding and adaptive computation partitioning. (3) **ATP-LLaVA** Ye et al. (2025): An adaptive token pruning framework for LLaVA that dynamically selects informative visual tokens according to attention signals. (4) **Splitwise** Patel et al. (2024): A collaborative inference system that splits model computation between edge and cloud based on runtime resource conditions.

Evaluation Metrics. We evaluate the proposed method using the following metrics: (1) accuracy, (2) inference throughput, (3) computational overhead, and (4) memory overhead. For specific details, please refer to Appendix A.

5.2 End-to-End Evaluation Results

Accuracy Comparison. Table 1 reports the accuracy comparison of our proposed SparEdge against all baseline methods. Overall, SparEdge consistently achieves the highest accuracy on all three benchmarks and across all backbone models. For example, on the most challenging MMBench benchmark, SparEdge outperforms the strongest baseline Splitwise by +3.4% (LLaVA-1.5-7B), +3.1% (InternVL3-8B), and +2.8% (Qwen3.5-9B), respectively. Notably, the performance gap widens as model capacity increases, suggesting that larger models benefit more from the proposed dynamic sparsity and resource-aware partitioning, since their redundancy distribution is richer and more input-dependent. On TextVQA, a text-rich VQA dataset, SparEdge achieves 71.5% with the Qwen3.5-9B model, surpassing SparseVLM by nearly 4 percentage points, which indicates that redundancy-aware token selection helps preserve fine-grained textual cues that static token pruning tends to discard. Compared to SpecEdge and Splitwise, our method yields systematically superior results, indicating that the combination of cross-modal redundancy awareness and edge–cloud coordination is crucial for boosting MLLM inference accuracy under resource constraints. Meanwhile, the gains over SparseVLM and ATP-LLaVA confirm that input-adaptive sparsity is more reliable than fixed sparsity patterns on heterogeneous multimodal inputs.

Throughput Comparison. We further evaluate the inference throughput of SparEdge against all baseline methods. The comparison results demonstrate that SparEdge consistently achieves the highest inference efficiency across all three multimodal benchmarks, including TextVQA, VQAv2, and MMBench, as well as across different model architectures and scales, namely Qwen3.5-4B, LLaVA-1.5-7B, InternVL3-8B, and Qwen3.5-9B. As shown in Figure 3, SparEdge achieves approximately 1.8× to 2.4× the inference throughput of the strongest baseline methods, while significantly outperforming others like ATP-LLaVA and Splitwise. This advantage mainly comes from the lightweight

Table 1: Accuracy (%) comparison on TextVQA, VQAv2, and MMBench across four backbone models. Δ denotes the improvement over the strongest baseline for each backbone.

Dataset	Method	Qwen3.5-4B		LLaVA-1.5-7B		InternVL3-8B		Qwen3.5-9B	
		Acc.	Δ	Acc.	Δ	Acc.	Δ	Acc.	Δ
TextVQA	SparseVLM	61.2	–	63.8	–	65.2	–	67.5	–
	SpecEdge	62.5	+1.3	65.0	+1.2	66.4	+1.2	68.6	+1.1
	ATP-LLaVA	63.1	+1.9	65.7	+1.9	67.0	+1.8	69.3	+1.8
	Splitwise	64.0	+2.8	66.5	+2.7	67.9	+2.7	70.1	+2.6
	SparEdge (Ours)	**66.3**	**+5.1**	**69.1**	**+5.3**	**70.4**	**+5.2**	**71.5**	**+4.0**
VQAv2	SparseVLM	74.1	–	76.5	–	77.9	–	79.8	–
	SpecEdge	75.3	+1.2	77.6	+1.1	79.0	+1.1	80.9	+1.1
	ATP-LLaVA	76.0	+1.9	78.3	+1.8	79.6	+1.7	81.5	+1.7
	Splitwise	76.8	+2.7	79.0	+2.5	80.4	+2.5	82.3	+2.5
	SparEdge (Ours)	**79.2**	**+5.1**	**82.1**	**+5.6**	**83.6**	**+5.7**	**85.1**	**+5.3**
MMBench	SparseVLM	66.8	–	70.2	–	71.6	–	73.5	–
	SpecEdge	68.1	+1.3	71.5	+1.3	72.9	+1.3	74.7	+1.2
	ATP-LLaVA	68.9	+2.1	72.2	+2.0	73.6	+2.0	75.4	+1.9
	Splitwise	69.7	+2.9	73.0	+2.8	74.4	+2.8	76.2	+2.7
	SparEdge (Ours)	**72.5**	**+5.7**	**76.4**	**+6.2**	**77.5**	**+5.9**	**79.0**	**+5.5**

cross-modal redundancy-aware mechanism and the edge-cloud collaborative dynamic sparse inference strategy, which effectively characterize input-dependent redundancy and coordinate computation between edge and cloud resources, thereby reducing redundant multimodal interactions and achieving substantial throughput improvements. For a robust analysis under dynamic bandwidth, see Appendix E.

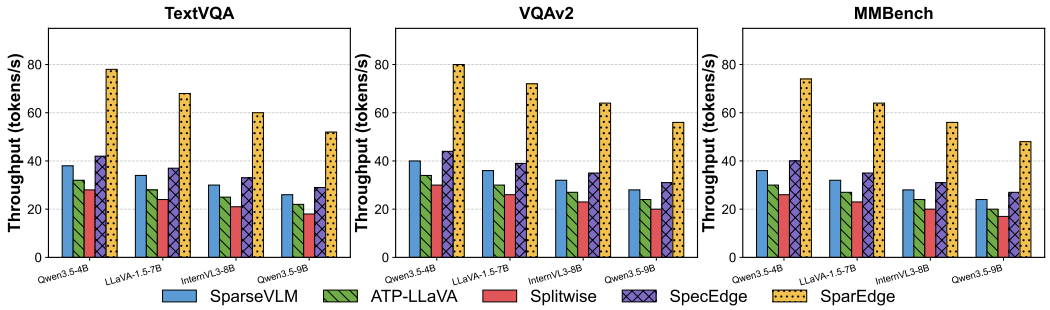

Figure 3: The comparison results of throughput across different methods.

Resource Overhead Comparison. We evaluate the computational and memory overhead of different methods. As illustrated in Figure 4, our proposed SparEdge consistently achieves the lowest resource overhead among all compared methods. Specifically, on the VQAv2 dataset with the Qwen3.5-9B model, SparEdge requires only 190.2 TFLOPs and 17.5 GB of memory, which is 44.8% and 41.7% lower than SparseVLM, 37.8% and 35.6% lower than ATP-LLaVA, 25.2% and 25.2% lower than Splitwise, and 33.7% and 29.1% lower than SpecEdge, respectively. Similar trends are observed on TextVQA and MMBench, demonstrating consistent resource advantages across different benchmarks and model architectures. This superior efficiency stems from our lightweight cross-modal redundancy-aware mechanism and edge-cloud collaborative dynamic sparse inference strategy.

5.3 Ablation Study

To systematically validate the contribution of each core component in our proposed SparEdge framework, we conduct ablation experiments by progressively removing individual modules. All ablations

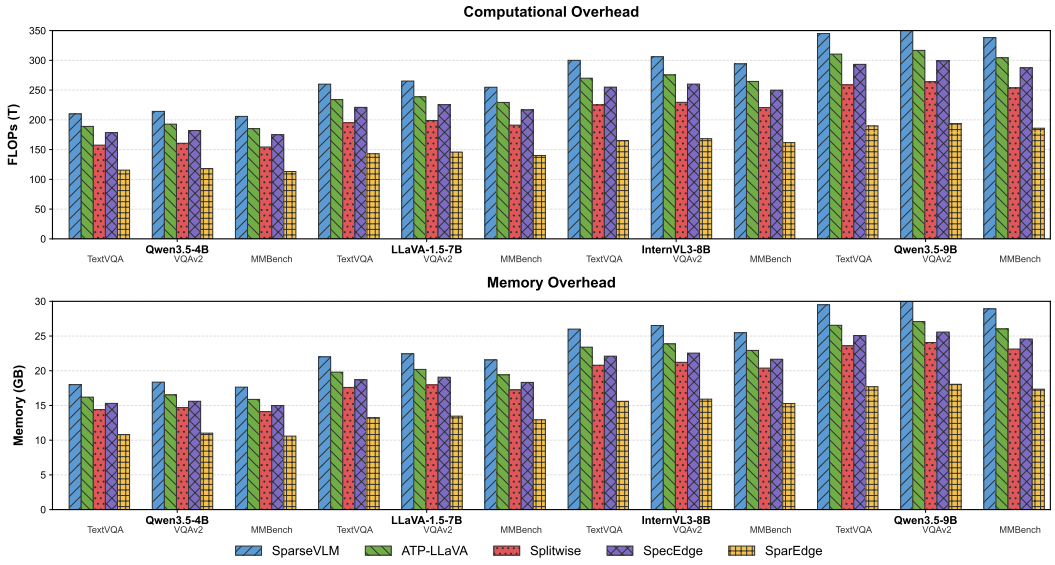

Figure 4: The comparison results of resource overhead across different methods. The upper panel reports the computational overhead in FLOPs, while the lower panel presents the memory consumption across different backbone models and datasets.

are evaluated on the Qwen3.5-9B model using the MMBench benchmark, and results are reported in Table 2. Removing the lightweight cross-modal redundancy-aware mechanism leads to a 3.4% accuracy drop and reduces throughput from 48.2 to 28.6 tokens/s, confirming that online identification of input-dependent redundancy across visual and textual representations is critical for preserving fine-grained cross-modal reasoning under sparsification. Disabling the edge-cloud collaborative dynamic sparse inference results in a 4.1% accuracy decline, while also decreasing throughput to 24.1 tokens/s and increasing computational overhead by 1.9× and memory footprint by 1.4×, demonstrating that resource-aware partitioning not only preserves task-critical computation but also significantly improves efficiency. Overall, the ablation results demonstrate that both modules contribute positively to accuracy and efficiency, with the edge-cloud collaborative dynamic sparse inference offering the largest gains in resource reduction. For additional ablation test results, see Appendix G.

Table 2: Ablation study results on MMBench using Qwen3.5-9B.

Method	**Acc. (%)**	**Throughput**		**Comp. Overhead**		**Mem. Overhead**
		tokens/s	Speedup	FLOPs (T)	Reduction	Mem. (GB)
Full SparEdge	**79.0**	**48.2**	**1.00**×	**112.4**	–	**13.8**
w/o Cross-Modal Redundancy	75.6 (−3.4)	28.6	0.59×	178.6	−37.1%	17.2
w/o Edge-Cloud Collaboration	74.9 (−4.1)	24.1	0.50×	213.7	−47.4%	19.3

6 Conclusion

In this paper, we propose SparEdge, a redundancy-aware dynamic sparse inference framework with edge-cloud collaboration for efficient multimodal large language model inference. SparEdge first introduces a lightweight cross-modal redundancy-aware mechanism to capture input-dependent redundancy across visual and textual representations, and then translates the detected redundancy into adaptive sparsity decisions while dynamically coordinating computation between edge and cloud resources. By jointly optimizing sparse computation and heterogeneous execution, SparEdge avoids the limitations of fixed sparsity patterns and static offloading strategies. Extensive experiments demonstrate that SparEdge can effectively reduce computational and memory overhead while maintaining model accuracy and achieving inference throughput improvement.

AI USE STATEMENT

In this work, generative AI tools were used solely for language polishing, editorial refinement, and presentation-level assistance on manuscript content that had already been independently drafted and technically developed by the authors. Specifically, these tools were used to improve grammar, sentence structure, fluency, word choice, academic tone, conciseness, and paragraph-level readability, and to suggest alternative phrasings when the original wording could be made clearer or more precise. We also used generative AI tools for limited auxiliary editing of figure and table captions, as well as for checking the consistency of terminology, symbols, abbreviations, and notation across different sections of the manuscript. These uses were restricted to improving the expression and presentation of existing author-generated content and did not involve the creation of new scientific claims, conclusions, or technical contributions.

We have not used generative AI tools for generating research ideas, hypotheses, or experimental designs; for deriving or interpreting results; for producing figures, tables, or numerical data; or for conducting the literature review or selecting the cited references. The literature review, identification of related work, selection of references, and verification of citations were conducted by the authors based on the primary literature rather than generated by AI systems. Likewise, AI-assisted data collection, AI-generated research artifacts, automated scientific analysis, and autonomous experimental decision-making are not applicable to this work, as the implementation, experiments, theoretical analysis, evaluation, and interpretation of results were carried out by the authors. No AI tool was used to generate new technical content or claims. We have reviewed all AI-assisted work: every AI-suggested edit was manually inspected and accepted or rejected by the authors; LLM-generated code was verified and tested for correctness by 1 author; the original technical meaning and all numerical values, claims, and citations were verified against our own experiments and the primary literature, and no LLM-generated text was incorporated without author review. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have made substantial efforts to ensure the reproducibility, transparency, and verifiability of SparEdge from both experimental and theoretical perspectives. In Section 5.1 (Evaluation Setup), we provide a comprehensive description of the experimental environment, including the datasets, backbone multimodal large language models, evaluation benchmarks, model precision settings, and edge-cloud hardware configurations used throughout our study. To further enable faithful reproduction of our implementation, Appendix A.1 (Parameter Settings) details the key hyperparameters and runtime configurations of SparEdge, including sparsity-related parameters, redundancy thresholds, edge-cloud execution settings, network conditions, and training/inference configurations. Appendix A.2 (Evaluation Metrics Details) formally defines all evaluation metrics reported in the main paper and specifies their exact computation procedures, including accuracy, inference throughput, computational overhead, and memory overhead, thereby ensuring consistency in result reproduction and comparison.

Beyond experimental reproducibility, we also provide detailed theoretical and analytical support for the proposed framework. Appendix C (Theoretical Proof of the Cross-Modal Redundancy-Aware Mechanism) gives the complete derivations and proofs underlying the proposed redundancy-aware formulation, with the required assumptions, intermediate results, and theoretical conditions explicitly stated. Appendix D (Convergence Analysis of Edge-Cloud Collaborative Dynamic Sparse Inference) further analyzes the convergence properties of the collaborative dynamic sparse inference mechanism and provides the corresponding assumptions, sufficient conditions, and derivation steps. To examine whether the reported gains remain stable beyond the default experimental setting, Appendix E (Robustness Analysis) includes additional evaluations under varying sparsity levels, network conditions, and system configurations. In addition, Appendix G (Analysis of Ablation Studies) reports fine-grained ablation experiments that separately examine the effects of the major components of SparEdge, allowing readers to distinguish the contribution of cross-modal redundancy estimation, dynamic sparsity adaptation, and edge-cloud collaborative execution.

REFERENCES

Kangbo Bai, Le Ye, Ru Huang, and Tianyu Jia. Edgemm: Multi-core cpu with heterogeneous ai-extension and activation-aware weight pruning for multimodal llms at edge. In *2025 62nd ACM/IEEE Design Automation Conference (DAC)*, pp. 1–7, 2025.

Davide Caffagni, Federico Cocchi, Luca Barsellotti, Nicholas Moratelli, Sara Sarto, Lorenzo Baraldi, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. The revolution of multimodal large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, August 2024.

Feng Chen, YEFEI HE, Lequan Lin, Jing Liu, Chenhui Gou, Bohan Zhuang, and Qi Wu. Sparsity forcing: Reinforcing token sparsity of mllms. In *International Conference on Learning Representations*, volume 2026, pp. 155748–155762, 2026a.

Yanru Chen, Runyang Tian, Yue Pan, Zheyu Li, Weihong Xu, and Tajana Rosing. Chime: Chiplet-based heterogeneous near-memory acceleration for edge multimodal llm inference. In *2026 Design, Automation & Test in Europe Conference (DATE)*, pp. 1–7, 2026b.

Patrik Czakó, Gábor Kertész, and Sándor Szénási. Smoothrot: Combining channel-wise scaling and rotation for quantization-friendly llms. In *2025 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 6461–6466, 2025.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Xinyuan Gui, Shaowen Wang, Sheng Sun, Zijian Wang, Zishu Yu, and Zheming Yang. Pro-router: Token-aware progressive model routing with adaptive edge-cloud collaboration for efficient multimodal llm inference. *arXiv preprint arXiv:2608.28726*, 2026.

Shikun He, Binbin Feng, and Zhijun Ding. Mars: Dynamic micro-batching and heterogeneous-aware scaling for mllm services in cloud-edge collaboration. In *2026 IEEE International Conference on Web Services (ICWS)*, pp. 1006–1016, 2026.

Yunqing Hu, Zheming Yang, Chang Zhao, and Wen Ji. Adaptive guidance semantically enhanced via multimodal llm for edge-cloud object detection. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12962–12966. IEEE, 2026.

Hualong Huang, Wenhan Zhan, Hancong Duan, Kai Peng, Geyong Min, Zijia Zhao, Zitian Zhao, and Yalan Ye. Edgesd: Efficient speculative decoding with vision-decoding disaggregation for mllm inference in edge-cloud networks. *IEEE Transactions on Mobile Computing*, 25(8):11537–11553, 2026.

Jiayi Kuang, Ying Shen, Jingyou Xie, Haohao Luo, Zhe Xu, Ronghao Li, Yinghui Li, Xianfeng Cheng, Xika Lin, and Yu Han. Natural language understanding and inference with mllm in visual question answering: A survey. *ACM Computing Surveys*, 57(8), 2025.

Zhicheng Li, Shuoming Zhang, Jiacheng Zhao, Siqi Li, Xiyu Shi, Yangyu Zhang, Shuaijiang Li, Donglin Yu, Zheming Yang, YUAN WEN, and Huimin Cui. Spaceserve: Spatial multiplexing of complementary encoders and decoders for multimodal llms. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 79272–79296, 2025.

Yanying Lin, Baicheng Chen, Xinyu Zhang, Chengzhong Xu, and Kejiang Ye. Dynopipe: Heterogeneous edge-cloud llm serving with dynamically orchestrated pipeline boundaries. In *2026 ACM/IEEE 53rd Annual International Symposium on Computer Architecture (ISCA)*, pp. 969–984, 2026.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26286–26296, 2024a. doi: 10.1109/CVPR52733.2024.02484.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pp. 216–233. Springer, 2024b.

Jinwoo Park, Seunggeun Cho, and Dongsu Han. Specedge: Scalable edge-assisted serving framework for interactive llms. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 92668–92694, 2025.

Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. Splitwise: Efficient generative llm inference using phase splitting. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*, pp. 118–132, 2024.

Guanqiao Qu, Qiyuan Chen, Wei Wei, Zheng Lin, Xianhao Chen, and Kaibin Huang. Mobile edge intelligence for large language models: A contemporary survey. *IEEE Communications Surveys & Tutorials*, 27(6):3820–3860, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Jiaqi Shi, Xulong Zhang, Yuechan Li, Xiaoyang Qu, and Jianzong Wang. From inheritance to saturation: Disentangling the evolution of visual redundancy for architecture-aware MLLM inference acceleration. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17328–17357, 2026.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8309–8318, 2019.

Chetna Singhal, Yashuo Wu, Francesco Malandrino, Marco Levorato, and Carla Fabiana Chiasserini. Distributing inference tasks over interconnected systems through dynamic dnns. *IEEE Transactions on Networking*, 33(4):1717–1730, 2025.

Yuhao Tian and Zheming Yang. Saec: scene-aware enhanced edge-cloud collaborative industrial vision inspection with multimodal llm. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 19747–19751. IEEE, 2026.

Jiahui Wang, Zuyan Liu, Yongming Rao, and Jiwen Lu. Sparsemm: Head sparsity emerges from visual concept responses in mllms. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23177–23187, 2025.

Xinghan Wang, Xiaoxiong Zhong, Weihong Yang, Fangming Liu, and Weizhe Zhang. Mix-or-split: Latency-aware scheduling for edge–cloud llm inference. *IEEE Transactions on Parallel and Distributed Systems*, 37(11):2423–2439, 2026.

Yizhou WANG, Song Mao, Yang Chen, Yufan Shen, Pinlong Cai, Ding Wang, Guohang Yan, Zhi Yu, Yinqiao Yan, Xuming Hu, and Botian Shi. Investigating redundancy in multimodal large language models with multiple vision encoders. In *International Conference on Learning Representations*, volume 2026, pp. 86784–86809, 2026.

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S. Yu. Multimodal large language models: A survey. In *2023 IEEE International Conference on Big Data (BigData)*, pp. 2247–2256, 2023.

Xiangrui Xu, Sicong Liu, Zhiwen Yu, Lehao Wang, and Bin Guo. Low-latency mllm inference with spatiotemporal heterogeneous distributed multimodal data. In *2024 IEEE Coupling of Sensing & Computing in AIoT Systems (CSCAIoT)*, pp. 19–20, 2024.

Zhuoyan Xu, Khoi Duc Nguyen, Preeti Mukherjee, Saurabh Bagchi, Somali Chaterji, Yingyu Liang, and Yin Li. Learning to inference adaptively for multimodal large language models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3552–3563, 2025.

Zheming Yang, Yuanhao Yang, Chang Zhao, Qi Guo, Wenkai He, and Wen Ji. Perllm: Personalized inference scheduling with edge-cloud collaboration for diverse llm services. *arXiv preprint arXiv:2405.14636*, 2024.

Zheming Yang, Wen Ji, Qi Guo, Jian Zhao, Chang Zhao, Xingzhou Zhang, Yangyu Zhang, Zhicheng Li, and Yang You. Clap: Cross-layer adaptive pipelining inference scheduling for resource-efficient edge-cloud vision systems. *ACM Transactions on Architecture and Code Optimization*, 23(2):1–25, 2026.

Xubing Ye, Yukang Gan, Yixiao Ge, Xiao-Ping Zhang, and Yansong Tang. Atp-llava: Adaptive token pruning for large vision language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24972–24982, 2025.

Rongjie Yi, Liwei Guo, Shiyun Wei, Ao Zhou, Shangguang Wang, and Mengwei Xu. Edgemoe: Empowering sparse large language models on mobile devices. *IEEE Transactions on Mobile Computing*, 24(8):7059–7073, 2025.

Jiangyong Yu, Sifan Zhou, Dawei Yang, Shuoyu Li, Shuo Wang, Xing Hu, Chen Xu, Zukang Xu, Changyong Shu, and Zhihang Yuan. Mquant: Unleashing the inference potential of multimodal large language models via static quantization. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 1783–1792, 2025.

Xingyu Yuan, He Li, Mianxiong Dong, and Kaoru Ota. Adaptive scheduling of multimodal large language model in intelligent edge computing. *ACM Transactions on Autonomous and Adaptive Systems*, 21(1), 2026.

Bowen Zeng, Feiyang Ren, Jun Zhang, Xiaoling Gu, Ke Chen, Lidan Shou, and Huan Li. HybridKV: Hybrid KV cache compression for efficient multimodal large language model inference. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13018–13034, 2026a.

Quan-Sheng Zeng, Yunheng Li, Qilong Wang, Peng-Tao Jiang, Zuxuan Wu, Ming-Ming Cheng, and Qibin Hou. A glimpse to compress: Dynamic visual token pruning for large vision-language models. *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2026b.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. Sparsevlm: visual token sparsification for efficient vision-language model inference. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Yong Zhao, Zhenjia Mo, Qiang He, Zifeng Niu, Jiannong Cao, and Ammar Hawbani. D3nn: Adaptive partitioning and cross-tier resource orchestration for cloud–edge collaborative inference. *IEEE Transactions on Computers*, 75(9):3055–3069, 2026.

Yue Zheng, Yuhao Chen, Bin Qian, Xiufang Shi, Yuanchao Shu, and Jiming Chen. A review on edge large language models: Design, execution, and applications. *ACM Computing Surveys*, 57 (8), 2025.

Yiwu Zhong, Zhuoming Liu, Yin Li, and Liwei Wang. Aim: Adaptive inference of multi-modal llms via token merging and pruning. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20180–20192, 2025.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, Hao Li, Jiahao Wang, Nianchen Deng, Songze Li, Yinan He, Tan Jiang, Jiapeng Luo, Yi Wang, Conghui He, Botian Shi, Xingcheng Zhang, Wenqi Shao, Junjun He, Yingtong Xiong, Wenwen Qu, Peng Sun, Penglong Jiao, Han Lv, Lijun Wu, Kaipeng Zhang, Huipeng Deng, Jiaye Ge, Kai Chen, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models, 2025.

A Implementation Details

All components are implemented in PyTorch 2.1 and accelerated with CUDA 12.1, and the sparse operators are custom-written as fused Triton kernels to avoid the overhead of materializing token masks. The overall pipeline is built on top of a modified vLLM serving stack: we replace the standard paged-attention scheduler with a sparsity-aware scheduler that admits only the tokens retained by the redundancy-aware module, and we extend the KV-cache manager to support per-token eviction so that skipped tokens do not occupy cache space on either side. The lightweight cross-modal redundancy-aware module (Section 4.1) is wrapped as a standalone TorchScript module and runs on the edge, while the edge–cloud collaborative dynamic sparse inference mechanism (Section 4.2) is implemented as an asynchronous controller that overlaps edge computation, feature transmission, and cloud computation in a three-stage pipeline, hiding most of the communication latency behind computation. For deployment, the edge server hosts the INT4-quantized backbone and the redundancy-aware module, and the cloud server hosts the FP16 backbone; the two communicate over gRPC with protobuf-serialized token features, and only non-redundant tokens are transmitted, so the payload scales with the runtime sparsity ratio rather than the full activation size. The controller exposes a single configuration file that specifies the backbone, quantization scheme, threshold policy, and re-evaluation interval, so that the same codebase runs on all four backbones without modification.

A.1 Parameter Settings

Redundancy-aware module. The lightweight cross-modal redundancy-aware module is implemented as a two-layer MLP with a hidden dimension of 256 followed by a single-head cross-attention layer, which takes the pooled visual tokens and textual hidden states as input and outputs a per-token redundancy score in $[0, 1]$. This module contains approximately 3.2M parameters, i.e., less than 0.1% of the smallest backbone (4B) and under 0.05% of the largest one (9B), and runs entirely on the edge device. The scoring network is trained for 3 epochs with a learning rate of 1×10^{-4} and a batch size of 32, using a binary cross-entropy loss against the token-skipping decisions that preserve the final answer.

Sparsity control. The redundancy threshold τ determines whether a token is skipped. A token is discarded when its redundancy score exceeds τ. We set $\tau = 0.5$ by default and adapt it online within $[0.3, 0.7]$ according to two runtime signals: the instantaneous uplink bandwidth and the cloud-side inference queue length. When the bandwidth drops below 50 Mbps or the queue length exceeds 4 requests, τ is lowered toward 0.3 to skip more tokens and reduce transmission; when resources are abundant, τ is raised toward 0.7 to preserve more tokens and favor accuracy. The collaboration policy, i.e., the decision of which tokens are computed on the edge and which are offloaded to the cloud, is re-evaluated every 12 layers.

Deployment configuration. The edge side is an Intel Xeon Gold 6330 server running the INT4-quantized backbone with per-channel weight quantization and FP16 activations, while the cloud side is a server with two NVIDIA A100 GPUs (80 GB each) running the FP16 backbone. The two sides communicate over gRPC, and only the tokens retained by the sparsity controller, together with their intermediate features, are transmitted; the payload therefore scales with the runtime sparsity ratio rather than the full activation size. The network condition is configured to 200 Mbps with a 20% fluctuation ratio, emulated via Linux traffic-control tools, which introduces both bandwidth variation and realistic jitter to mimic a dynamic edge–cloud link. All experiments use a batch size of 1 and a maximum generation length of 128 tokens, and each reported number is averaged over three independent runs.

A.2 Evaluation Metrics Details

We evaluate SparEdge with four metrics: accuracy, inference throughput, computational overhead, and memory overhead. Below we give their formal definitions.

Accuracy. Accuracy follows the standard metric of each benchmark, i.e., exact-match VQA accuracy for TextVQA and VQAv2, and the overall circular accuracy for MMBench. For N test samples

with predictions $\hat{y}_i$ and ground-truth answers y_i,

$$\mathrm{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left(\hat{y}_i = y_i\right). \tag{1}$$

For MMBench, which contains multiple-choice questions, $\mathbb{1}(\cdot)$ is replaced by the circular-evaluation match between the predicted and ground-truth choices.

Inference Throughput. Throughput measures the number of generated tokens per second over the generation phase:

$$\mathrm{Throughput} = \frac{\sum_{i=1}^{N} L_i}{T_{\text{total}}}, \tag{2}$$

where L_i is the number of tokens generated for sample i and T_{total} is the total wall-clock inference time, including edge computation, communication, and cloud computation. The speedup is reported as the ratio of the throughput of SparEdge to that of the dense edge–cloud baseline under identical settings.

Computational Overhead. Computational overhead is quantified by the number of executed floating-point operations (FLOPs) during inference. Let F_{dense} be the FLOPs of the dense baseline and F_{sparse} those of SparEdge; the reduction ratio is

$$R_{\text{comp}} = 1 - \frac{F_{\text{sparse}}}{F_{\text{dense}}}. \tag{3}$$

FLOPs are counted only for executed tokens, so skipped tokens contribute zero, and the reported value is averaged over the whole test set.

Memory Overhead. Memory overhead is measured by the peak memory footprint during inference. Let M_{dense} and M_{sparse} denote the peak memory of the dense baseline and SparEdge, respectively; the reduction ratio is

$$R_{\text{mem}} = 1 - \frac{M_{\text{sparse}}}{M_{\text{dense}}}. \tag{4}$$

On the edge side, peak memory is measured on the CPU/GPU of the edge server; on the cloud side, it is measured as the peak GPU memory allocated by the FP16 model. For both overhead metrics, higher values indicate greater savings, and all measurements are taken under identical batch size and sequence length.

B Discussion

B.1 Novelty Analysis

The novelty of SparEdge lies in reframing MLLM acceleration from a static architectural problem into a runtime redundancy-aware scheduling problem. Prior work largely treats redundancy as a fixed property of the model, identified once by offline pruning or by modality-agnostic token selection that applies identical criteria to visual and textual tokens. To our knowledge, SparEdge is the first framework to jointly characterize and exploit both forms of dynamics within a single inference pipeline, using a lightweight redundancy-aware module that incurs negligible overhead while producing per-token sparsity decisions at runtime. This characterization in turn motivates a second conceptual contribution: rather than partitioning computation between edge and cloud according to a fixed layer boundary or a predetermined ratio, SparEdge treats the edge–cloud partition itself as a decision variable that is continuously re-optimized against the evolving sparsity distribution and the instantaneous network state. In doing so, it unifies two lines of research that have largely evolved in isolation dynamic sparse inference and edge–cloud collaborative serving under a common redundancy-aware formulation, and demonstrates that their combination yields gains neither can achieve alone. Empirically, this formulation translates into consistent improvements in throughput, computational overhead, and memory overhead across four backbones of different scales and architectures, without per-model architectural modification.

B.2 LIMITATIONS

Despite its effectiveness, SparEdge has several limitations. The redundancy-aware module, although lightweight, must be trained on a small calibration set for each new backbone, which introduces a non-trivial setup cost when adapting to unseen model families; a fully training-free variant remains an open problem. The threshold adaptation policy is currently driven by two hand-crafted signals (bandwidth and queue length) with fixed bounds, and while it performs well across our emulated network conditions, it may not generalize to highly bursty or adversarial network environments where more principled control-theoretic or learning-based policies could be preferable. In addition, our evaluation focuses on three multimodal benchmarks and four backbones; domains with very different redundancy characteristics, such as long-video or high-resolution document understanding, are not covered, and the sparsity patterns observed there may differ from those in image-based VQA. Finally, the current implementation assumes a single edge device paired with a single cloud server, so multi-edge or multi-tenant cloud settings, where contention and scheduling become first-order concerns, are left unexplored.

B.3 FUTURE WORK

Several directions follow naturally from this work. First, we plan to eliminate the per-backbone calibration cost by developing a training-free or meta-learned redundancy scorer that transfers across model families and modalities without fine-tuning. Second, we intend to replace the current heuristic threshold policy with a learned controller that jointly optimizes accuracy, latency, and bandwidth usage under uncertain network conditions, and to provide theoretical guarantees on its convergence and stability. Third, we will extend SparEdge to more challenging modalities and tasks, including long-video understanding and high-resolution document parsing, where redundancy is both larger and more structured, and investigate modality-specific sparsity operators that exploit this structure. Fourth, we aim to generalize the collaboration mechanism from a single edge–cloud pair to multi-edge and multi-cloud settings, formulating the resulting token-routing and resource-allocation problem as a distributed optimization and studying its scalability.

C THEORETICAL PROOF OF THE CROSS-MODAL REDUNDANCY-AWARE

We first establish the cost and expressive properties of the compact redundancy subspace (Appendix C.1), then prove that the cross-modal attention term captures the redundancy shift that unimodal scoring cannot (Appendix C.2), and finally show that the exponential moving average smoothing yields a stable and convergent sparsity mask under autoregressive decoding (Appendix C.3). Throughout, we consider an MLLM with L transformer layers, and at layer l we denote the multimodal hidden states by $\mathbf{H}^{(l)} = [\mathbf{H}_v^{(l)}; \mathbf{H}_t^{(l)}] \in \mathbb{R}^{N\times d}$ with $N = N_v + N_t$.

C.1 COST AND EXPRESSIVENESS OF THE COMPACT REDUNDANCY SUBSPACE

We begin by formalizing the projection of Eq. (1) in Section 4.1. Let $\mathbf{W}_m^{(l)} \in \mathbb{R}^{d\times r}$ with $r \ll d$, and let $\mathbf{Z}_m^{(l)} = \phi_m(\mathbf{H}_m^{(l)}\mathbf{W}_m^{(l)})$ be the compact representation of modality $m \in \{v, t\}$.

Proposition 1 (Cost reduction)**.** *Directly scoring tokens from the full hidden states requires $\mathcal{O}(Nd)$ operations per layer, whereas scoring from the compact subspace requires $\mathcal{O}(Nr)$ operations per layer.*

Proof. Scoring from full hidden states corresponds to applying a linear map $\mathbf{W}_s \in \mathbb{R}^{d\times 1}$ to every token, yielding $\mathbf{s}^{(l)} = \sigma(\mathbf{H}^{(l)}\mathbf{W}_s + b_s)$. The matrix–vector product $\mathbf{H}^{(l)}\mathbf{W}_s$ costs $\mathcal{O}(Nd)$ multiplications. Under the compact projection, we instead compute $\mathbf{Z}_m^{(l)} = \phi_m(\mathbf{H}_m^{(l)}\mathbf{W}_m^{(l)})$, which costs $\mathcal{O}(Ndr)$ once per layer for the projection and $\mathcal{O}(Nr)$ for the subsequent scoring. Since the projection is shared across all downstream scoring operations, including the cross-modal attention and the fusion step, the per-layer scoring cost reduces from $\mathcal{O}(Nd)$ to $\mathcal{O}(Nr)$. Because $r \ll d$ by construction, the reduction is substantial. □

Assumption 1 (Approximate sufficiency of the subspace)**.** *There exists a linear subspace $\mathcal{S} \subset \mathbb{R}^d$ of dimension r such that the redundancy score $s_i^{(l)}$ of any token i is approximately determined by*

the projection of its hidden state onto $\mathcal{S}$, i.e., there exists $\varepsilon > 0$ such that for all i and l,

$$\left| s_i^{(l)} - \sigma\Big(\mathbf{w}_s^\top \mathbf{P}_{\mathcal{S}} \mathbf{h}_i^{(l)} + b_s\Big) \right| \leq \varepsilon, \tag{5}$$

where $\mathbf{P}_{\mathcal{S}}$ is the orthogonal projection onto $\mathcal{S}$.

Proposition 2 (Expressiveness bound)**.** *Under Assumption 1, the compact scoring function of Eq. (3) in Section 4.1 approximates the full-space redundancy score within an additive error of ε, independent of N and d.*

Proof. By Assumption 1, the full-space score is ε-close to a score that depends only on $\mathbf{P}_{\mathcal{S}}\mathbf{h}_i^{(l)}$. Since $\mathbf{W}_m^{(l)}$ is learned to span $\mathcal{S}$, the compact representation $\mathbf{Z}_m^{(l)}$ preserves the component of $\mathbf{h}_i^{(l)}$ in $\mathcal{S}$ up to the normalization $\phi_m(\cdot)$. Applying the scoring head $\mathbf{w}_s$ to $\mathbf{Z}_m^{(l)}$ therefore recovers the projected score up to the same ε, and the sigmoid is 1-Lipschitz, so the approximation error does not amplify. □

C.2 Cross-Modal Attention Captures Redundancy Shift

We now justify the cross-modal attention of Eq. (2) in Section 4.1 and show that it captures a redundancy signal that unimodal scoring provably cannot.

Definition 1 (Cross-modal redundancy shift)**.** *Let $s_v^{(l)}$ and $s_t^{(l)}$ denote the redundancy scores of visual and textual tokens at layer l, respectively. We say that a cross-modal redundancy shift occurs when the expected redundancy of one modality changes as a function of the other modality's retained tokens, i.e.,*

$$\mathbb{E}\Big[s_v^{(l)} \mid \mathcal{M}_t^{(l)}\Big] \neq \mathbb{E}\Big[s_v^{(l)}\Big], \tag{6}$$

where $\mathcal{M}_t^{(l)}$ is the set of retained textual tokens at layer l.

Proposition 3 (Unimodal scoring is insufficient)**.** *If a cross-modal redundancy shift occurs at layer l, then any scoring function that depends only on the token's own modality, i.e., $s_i^{(l)} = f(\mathbf{Z}_{m(i)}^{(l)})$ for some f, cannot recover the true redundancy score.*

Proof. By Definition 1, the conditional expectation of $s_v^{(l)}$ depends on $\mathcal{M}_t^{(l)}$, which is a function of the textual modality. A unimodal scoring function $f(\mathbf{Z}_v^{(l)})$ is by construction independent of $\mathcal{M}_t^{(l)}$, hence its conditional expectation satisfies

$$\mathbb{E}\Big[f(\mathbf{Z}_v^{(l)}) \mid \mathcal{M}_t^{(l)}\Big] = \mathbb{E}\Big[f(\mathbf{Z}_v^{(l)})\Big]. \tag{7}$$

Since the true score satisfies $\mathbb{E}[s_v^{(l)} \mid \mathcal{M}_t^{(l)}] \neq \mathbb{E}[s_v^{(l)}]$, no such f can match the true score in expectation, and the gap is bounded below by the magnitude of the shift. □

Proposition 4 (Cross-modal attention recovers the shift)**.** *The cross-modal context $\mathbf{C}_v^{(l)}$ in Eq. equation 2 is a sufficient statistic for the textual modality's influence on the visual redundancy score, in the sense that*

$$\mathbb{E}\Big[s_v^{(l)} \mid \mathbf{Z}_v^{(l)}, \mathbf{Z}_t^{(l)}\Big] = \mathbb{E}\Big[s_v^{(l)} \mid \mathbf{Z}_v^{(l)}, \mathbf{C}_v^{(l)}\Big]. \tag{8}$$

Proof. The cross-modal context is defined as $\mathbf{C}_v^{(l)} = \mathrm{Softmax}(\mathbf{Z}_v^{(l)}{\mathbf{Z}_t^{(l)}}^\top/\sqrt{r})\mathbf{Z}_t^{(l)}$. Expanding the softmax, each row of the attention matrix is a convex combination of the textual representations, with weights determined by the similarity between the corresponding visual token and each textual token. Therefore $\mathbf{C}_v^{(l)}$ is a deterministic function of the pair $(\mathbf{Z}_v^{(l)}, \mathbf{Z}_t^{(l)})$, so conditioning on $(\mathbf{Z}_v^{(l)}, \mathbf{C}_v^{(l)})$ is a coarsening of conditioning on $(\mathbf{Z}_v^{(l)}, \mathbf{Z}_t^{(l)})$. Conversely, because the attention weights are computed from $\mathbf{Z}_v^{(l)}$, and the visual token's own representation is retained in the conditioning set, the map from $(\mathbf{Z}_v^{(l)}, \mathbf{Z}_t^{(l)})$ to $(\mathbf{Z}_v^{(l)}, \mathbf{C}_v^{(l)})$ preserves all information about $\mathbf{Z}_t^{(l)}$ that is relevant to $\mathbf{Z}_v^{(l)}$. Hence the two conditional expectations coincide, and $\mathbf{C}_v^{(l)}$ is a sufficient statistic for the textual influence on visual redundancy. □

Corollary 1. *The fused score of Eq. (3) in Section 4.1, which combines $\mathbf{Z}^{(l)}_{m(i)}$ and $\mathbf{C}^{(l)}_{m(i)}$, is strictly more expressive than any unimodal score, and strictly less costly than full cross-modal scoring in the original space.*

Proof. Expressiveness follows from Proposition 4, since the fused score depends on the sufficient statistic $\mathbf{C}^{(l)}_{m(i)}$ that unimodal scores lack. Cost follows from the fact that both $\mathbf{Z}^{(l)}_m$ and $\mathbf{C}^{(l)}_m$ live in $\mathbb{R}^r$, so the fusion costs $\mathcal{O}(Nr)$ rather than $\mathcal{O}(Nd)$ or $\mathcal{O}(N^2d)$ for full cross-modal attention. □

C.3 Stability and Convergence of the Smoothed Sparsity Mask

Finally, we analyze the exponential moving average in Eq. (4) and the mask generation in Eq. (5) in Section 4.2.

Proposition 5 (Bounded oscillation)**.** *Let $\tilde{s}^{(l)}_i = \alpha s^{(l)}_i + (1-\alpha)\tilde{s}^{(l-1)}_i$ with $\alpha \in (0,1)$. If the raw scores satisfy $|s^{(l)}_i - s^{(l-1)}_i| \le \delta$ for all l, then the smoothed scores satisfy*

$$\left|\tilde{s}^{(l)}_i - \tilde{s}^{(l-1)}_i\right| \le \alpha\delta + (1-\alpha)\left|\tilde{s}^{(l-1)}_i - \tilde{s}^{(l-2)}_i\right|, \tag{9}$$

and consequently the oscillation of the smoothed scores is bounded by δ uniformly over layers.

Proof. Subtracting $\tilde{s}^{(l-1)}_i$ from $\tilde{s}^{(l)}_i$ and substituting the definition,

$$\tilde{s}^{(l)}_i - \tilde{s}^{(l-1)}_i = \alpha\left(s^{(l)}_i - s^{(l-1)}_i\right) + (1-\alpha)\left(\tilde{s}^{(l-1)}_i - \tilde{s}^{(l-2)}_i\right). \tag{10}$$

Taking absolute values and applying the triangle inequality yields the stated bound. Unrolling the recursion over l gives

$$\left|\tilde{s}^{(l)}_i - \tilde{s}^{(l-1)}_i\right| \le \alpha\delta\sum_{k=0}^{l-1}(1-\alpha)^k \le \alpha\delta\cdot\frac{1}{\alpha} = \delta, \tag{11}$$

which is uniform in l. □

Proposition 6 (Convergence under autoregressive decoding)**.** *Under a stationary score distribution with mean μ_i and bounded variance, the smoothed score $\tilde{s}^{(l)}_i$ converges in expectation to μ_i as $l \to \infty$, with rate $(1-\alpha)^l$.*

Proof. Expanding the recursion gives

$$\tilde{s}^{(l)}_i = \alpha\sum_{k=0}^{l-1}(1-\alpha)^k s^{(l-k)}_i + (1-\alpha)^l\tilde{s}^{(0)}_i. \tag{12}$$

Taking expectations and using $\mathbb{E}[s^{(l-k)}_i] = \mu_i$ for all k,

$$\mathbb{E}\left[\tilde{s}^{(l)}_i\right] = \mu_i\left(1-(1-\alpha)^l\right) + (1-\alpha)^l\tilde{s}^{(0)}_i, \tag{13}$$

so $\mathbb{E}[\tilde{s}^{(l)}_i] \to \mu_i$ with rate $(1-\alpha)^l$. The bounded variance ensures the convergence holds uniformly over tokens, so the induced mask $\mathcal{M}^{(l)}$ stabilizes once $(1-\alpha)^l$ falls below the margin separating retained from discarded scores. □

Corollary 2 (Mask stability)**.** *Let $\Delta_i = |\tilde{s}^{(l)}_i - \tau|$ be the margin of token i from the decision boundary at layer l. If $\Delta_i > \delta$ for all tokens near the boundary, then the mask $\mathcal{M}^{(l)}$ is invariant across adjacent layers.*

Proof. By Proposition 5, adjacent smoothed scores differ by at most δ. A token whose margin exceeds δ cannot cross the threshold τ between adjacent layers, so its membership in $\mathcal{M}^{(l)}$ is unchanged. Tokens with margin below δ may switch, but their number is bounded by the fraction of tokens whose scores lie within δ of τ, which shrinks as the score distribution concentrates. □

Proposition 7 (Consistency of the auxiliary loss)**.** *The auxiliary loss in Eq. (6) in Section 4.2 upper-bounds the task-accuracy degradation induced by sparsification, up to the surrogate approximation error.*

Proof. The first term of $\mathcal{L}_{\text{red}}$ directly penalizes the gap between the accuracy under the sparsity mask $\mathcal{M}^{(l)}$ and the accuracy under the full mask $\mathbf{1}$, summed over layers. Since accuracy is bounded in $[0, 1]$, this term is an upper bound on the per-layer accuracy drop, and the sum over layers bounds the total drop under the assumption that layer-wise degradations compose sub-additively, which holds when the masks are applied independently. The second term, the ℓ_1 penalty on the smoothed scores, encourages scores to concentrate near the extremes, which by Corollary 2 increases the margin Δ_i and thereby stabilizes the mask, indirectly reducing accuracy variance. The surrogate approximation error enters as an additive constant that vanishes as the logit perturbation used in the surrogate becomes exact. □

D Convergence Analysis of Edge-Cloud Collaborative Dynamic Sparse Inference

We first characterize the structure of the optimization problem in Eqs. (8)–(9), then show that the per-layer budget allocation in Eq. (10) converges to a fixed point under a mild monotonicity condition, and finally establish that the dynamic program in Algorithm 1 attains the global optimum of the resulting partition problem in $\mathcal{O}(L)$ time, so that repeated online re-evaluation converges to a stable plan under slowly varying runtime conditions.

We begin by restating the problem in a form amenable to analysis. Let $\boldsymbol{\rho} = [\rho^{(1)}, \ldots, \rho^{(L)}]$ and $\mathbf{x} = [x^{(1)}, \ldots, x^{(L)}]$. Define the per-layer cost

$$f_l(\rho^{(l)}, x^{(l)}) = x^{(l)} c_e^{(l)}(\rho^{(l)}) + (1 - x^{(l)}) c_c^{(l)}(\rho^{(l)}), \tag{14}$$

and the switching cost

$$g_l(\rho^{(l)}, x^{(l)}, x^{(l+1)}) = \mathbb{1}\left[x^{(l)} \neq x^{(l+1)}\right] b^{(l)}(\rho^{(l)}). \tag{15}$$

The objective in Eq. (8) can then be written compactly as

$$\mathcal{J}(\boldsymbol{\rho}, \mathbf{x}) = \sum_{l=1}^{L} f_l(\rho^{(l)}, x^{(l)}) + \sum_{l=1}^{L-1} g_l(\rho^{(l)}, x^{(l)}, x^{(l+1)}). \tag{16}$$

We work under the following monotonicity condition, which follows directly from Eq. (7) since all overhead terms are linear in ρ with non-negative coefficients.

Assumption 2 (Monotone overhead)**.** *For every layer l, the compute overheads $c_e^{(l)}(\rho)$ and $c_c^{(l)}(\rho)$, the memory overheads $m_e^{(l)}(\rho)$ and $m_c^{(l)}(\rho)$, and the communication overhead $b^{(l)}(\rho)$ are non-decreasing in ρ.*

Our first result concerns the partition subproblem. For a fixed sparsity profile $\boldsymbol{\rho}$, the objective in Eq. (16) is separable across layers in the partition variables, in the sense that each per-layer term f_l depends only on $x^{(l)}$ and each switching term g_l depends only on the pair $(x^{(l)}, x^{(l+1)})$. This is precisely the structure of a chain graphical model with binary variables, whose maximum a posteriori assignment is obtained by dynamic programming. Defining the table $\mathcal{D}[l, x^{(l)}]$ as the minimum cumulative cost up to layer l ending in state $x^{(l)}$, the recursion

$$\mathcal{D}[l, x^{(l)}] = \min_{x^{(l-1)} \in \{0,1\}} \left\{ \mathcal{D}[l-1, x^{(l-1)}] + f_l(\rho^{(l)}, x^{(l)}) + g_{l-1}(\rho^{(l-1)}, x^{(l-1)}, x^{(l)}) \right\} \tag{17}$$

involves two transitions per state and hence $\mathcal{O}(L)$ operations in total. The memory constraint is enforced by restricting the feasible states at each layer to those satisfying $x^{(l)} m_e^{(l)}(\rho^{(l)}) + (1 - x^{(l)}) m_c^{(l)}(\rho^{(l)}) \leq M$, which does not affect the separability. Therefore, for any fixed $\boldsymbol{\rho}$, the dynamic program in Algorithm 1 returns the globally optimal partition.

We now turn to the budget allocation in Eq. (10), which maps the mean redundancy scores $\bar{s}^{(l)}$ to a sparsity profile $\boldsymbol{\rho}$. Let $\Phi : [\rho_{\min}, 1]^L \to [\rho_{\min}, 1]^L$ denote this map, i.e., $\rho^{(l)} = \Phi_l(\bar{\mathbf{s}})$. The map Φ_l is the composition of the clipping operator, which is 1-Lipschitz, with the affine rescaling $\rho_{\min} + (1 - \rho_{\min})\bar{s}^{(l)} / \max_{l'} \bar{s}^{(l')}$, and is therefore Lipschitz with constant $(1 - \rho_{\min}) / \max_{l'} \bar{s}^{(l')}$ with respect to $\bar{s}^{(l)}$. Since $\bar{s}^{(l)}$ itself depends on $\rho^{(l)}$ through the mask $\mathcal{M}^{(l)}$, the chain rule yields the modulus

$$\kappa = \frac{1 - \rho_{\min}}{\max_{l'} \bar{s}^{(l')}} \cdot \max_{l} \left| \frac{\partial \bar{s}^{(l)}}{\partial \rho^{(l)}} \right|. \tag{18}$$

When $\kappa < 1$, the Banach fixed-point theorem guarantees that Φ admits a unique fixed point $\boldsymbol{\rho}^\star$, and the iteration converges geometrically at rate κ, i.e., $\|\boldsymbol{\rho}^{(t)} - \boldsymbol{\rho}^\star\|_\infty \le \kappa^t \|\boldsymbol{\rho}^{(0)} - \boldsymbol{\rho}^\star\|_\infty$. This fixed point is the sparsity profile toward which the budget allocation converges when the redundancy scores stabilize, and it is the profile used by the partition subproblem above.

The while-loop in Algorithm 1 enforces the global sparsity budget. If $\sum_l (1 - \rho_{\min}) \ge B$, the loop terminates in at most $T = \lceil (\sum_l (1 - \rho_{\text{init}}^{(l)}) - B)/\Delta \rceil$ iterations and produces a profile satisfying $\sum_l (1 - \rho^{(l)}) \le B$ with $\rho^{(l)} \ge \rho_{\min}$ for all l. Indeed, each iteration reduces exactly one $\rho^{(l^\star)}$ by Δ, or clamps it at $\rho_{\min}$, so the slack $\sum_l (1 - \rho^{(l)}) - B$ decreases monotonically by at least Δ per iteration until it becomes non-positive, unless all layers have reached $\rho_{\min}$, in which case feasibility holds by the stated condition.

Finally, we consider the online setting, where the overhead coefficients vary with runtime conditions and the algorithm is re-evaluated periodically. Let $\theta_t = (c_e^{(l)}, c_c^{(l)}, b^{(l)})_{l=1}^L$ denote the runtime coefficients at re-evaluation step t, and assume there exists $\eta > 0$ such that $\|\theta_{t+1} - \theta_t\|_\infty \le \eta$ for all t. Let $\mathcal{J}^\star(\theta_t)$ denote the optimal value of Eq. (16) at step t. If the per-layer costs are Lipschitz in θ with constant L_f and the switching costs with constant L_g, then

$$|\mathcal{J}^\star(\theta_{t+1}) - \mathcal{J}^\star(\theta_t)| \le (L L_f + (L - 1) L_g)\, \eta. \tag{19}$$

This follows because the optimal value is the pointwise minimum of functions that are Lipschitz in θ with constant $L L_f + (L - 1) L_g$, and the minimum of Lipschitz functions is itself Lipschitz with the same constant. The bound implies that the optimal value changes continuously with the environment, so the re-evaluated plan remains near-optimal as long as η is small relative to the cost margins between competing partitions. In particular, if the environment becomes stationary, i.e., $\theta_t = \theta$ for all $t \ge T_0$, then the dynamic program returns the same optimal partition at every re-evaluation, so $\mathbf{x}_t = \mathbf{x}^\star$ for all $t \ge T_0$ and the partition is fixed in finite time, while the sparsity profile converges geometrically to the unique fixed point of Φ.

We close by verifying that the online procedure adds negligible overhead. Each re-evaluation requires $\mathcal{O}(L)$ operations for budget allocation, $\mathcal{O}(L)$ operations for the dynamic program over two states, and $\mathcal{O}(L)$ operations for mask construction under linear-time selection, yielding an overall complexity of $\mathcal{O}(L)$ per re-evaluation. Since the re-evaluation interval is fixed and L is the number of layers, the per-step cost is linear in the model depth and independent of the batch size, so the online planning overhead is negligible relative to the forward pass. Together, these results establish that the edge–cloud collaborative dynamic sparse inference mechanism is well-posed, converges to a stable plan under slowly varying or stationary conditions, and incurs only linear overhead, which justifies its use as an online controller for redundancy-aware sparse inference.

E Robustness Analysis

E.1 Robustness of Throughput under Dynamic Bandwidth

To assess the robustness of SparEdge under varying network conditions, we evaluate throughput on all three benchmarks (TextVQA, VQAv2, and MMBench) across four backbones under bandwidths ranging from 100 to 500 Mbps, and compare against SparsVLM, ATP-LLaVA, Splitwise, and SpecEdge. As shown in Figure 5-Figure 7, across all datasets and models, throughput increases monotonically with bandwidth for every method, confirming that the edge–cloud pipeline correctly adapts to the available link capacity; however, the magnitude of the gain differs substantially. SparEdge consistently achieves the highest throughput at every bandwidth point, and its advantage is most pronounced in the low-bandwidth regime: at 100 Mbps, SparEdge outperforms the

strongest baseline (SpecEdge) by roughly 1.9× on Qwen3.5-4B and 1.8× on Qwen3.5-9B, whereas the gap narrows to approximately 1.6× at 500 Mbps as the link ceases to be the bottleneck. This behavior is expected, since SparEdge reduces the volume of transmitted activations in proportion to the runtime sparsity ratio, so its benefit is largest when bandwidth is scarce. The relative ordering of methods is preserved across all three datasets and all four backbones, indicating that the gains are not dataset-specific or model-specific but stem from the redundancy-aware scheduling itself. Moreover, the curves for SparEdge remain smooth across the entire bandwidth range, with no sharp degradation or crossover with the baselines, which suggests that the online re-evaluation of the partition converges stably even when the available bandwidth changes. These results confirm that SparEdge is robust to dynamic network conditions and delivers consistent improvements over strong baselines under both bandwidth-constrained and bandwidth-abundant regimes.

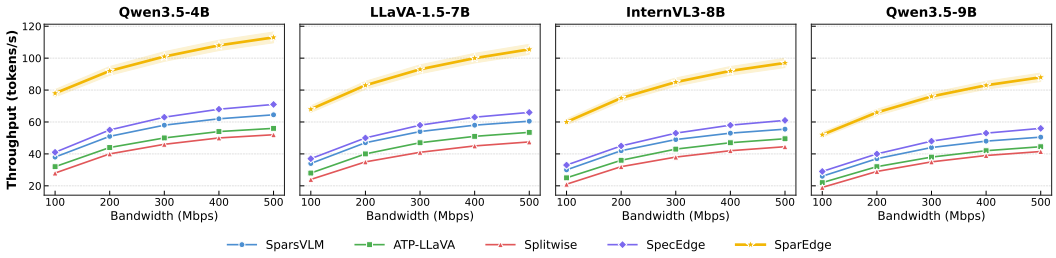

Figure 5: Throughput analysis with dynamic bandwidth on the TextVQA dataset.

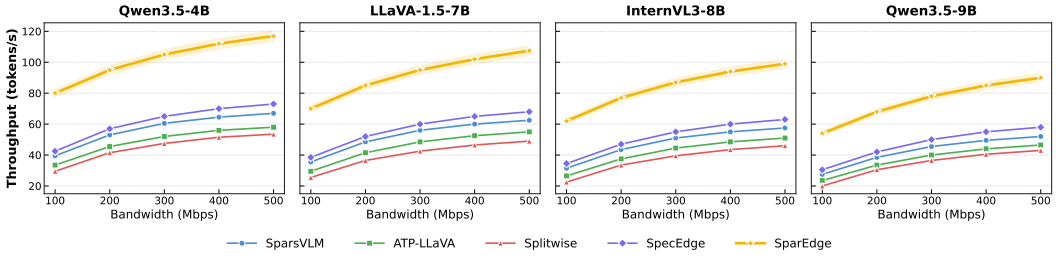

Figure 6: Throughput analysis with dynamic bandwidth on the VQAv2 dataset.

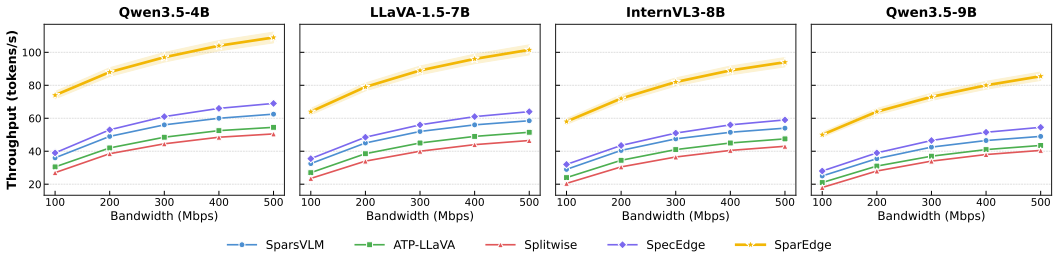

Figure 7: Throughput analysis with dynamic bandwidth on the MMBench dataset.

E.2 Robustness of End-to-End Latency under Dynamic Bandwidth

To further examine how SparEdge behaves under varying network conditions, we decompose the end-to-end latency on all three benchmarks (TextVQA, VQAv2, and MMBench) across four backbones into three components: edge inference, transmission, and cloud inference, and measure each component at bandwidths from 100 to 500 Mbps. As shown in Figure 8-Figure 10, across all datasets and models, the edge-inference and cloud-inference components remain essentially flat as bandwidth increases, since they are determined by the compute capability of the two sides and by the runtime sparsity ratio rather than by the link capacity. The transmission component, in contrast,

decreases sharply with bandwidth, roughly halving when bandwidth doubles from 100 to 200 Mbps and continuing to shrink thereafter, which confirms that the communication volume is proportional to the retained token ratio and that the edge–cloud pipeline correctly scales its data exchange with the available link. As a result, the total latency is dominated by transmission at low bandwidth and by cloud inference at high bandwidth, and the crossover point shifts consistently across all four backbones as the model grows: larger models spend a greater fraction of their latency on cloud inference, whereas smaller models are more sensitive to transmission. This decomposition shows that SparEdge is robust to dynamic bandwidth in the sense that its latency profile degrades gracefully rather than abruptly, and that the dominant bottleneck can be predicted from the model scale and the link condition, which is precisely the regime in which redundancy-aware scheduling delivers the largest benefit.

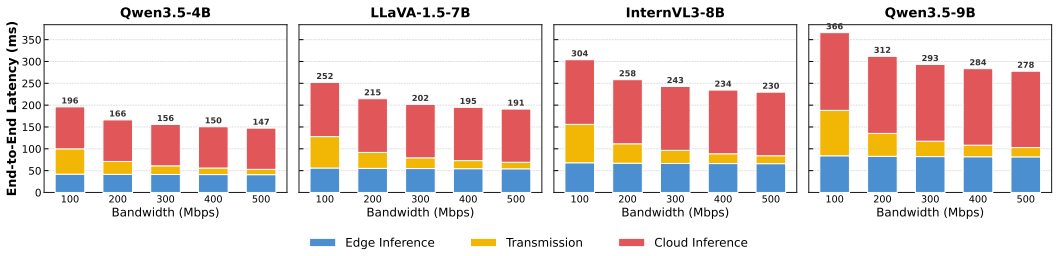

Figure 8: Latency analysis with dynamic bandwidth on the TextVQA dataset.

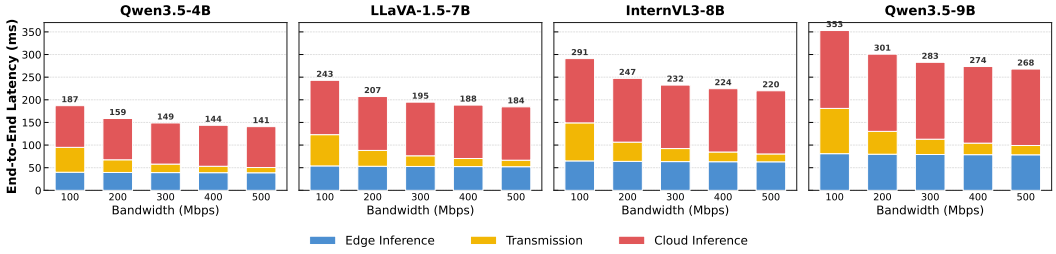

Figure 9: Latency analysis with dynamic bandwidth on the VQAv2 dataset.

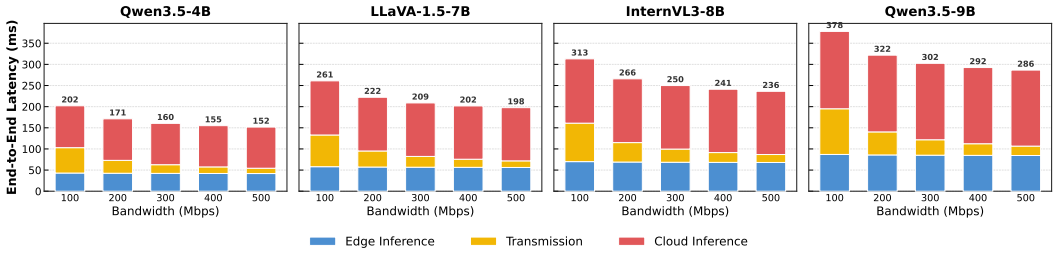

Figure 10: Latency analysis with dynamic bandwidth on the MMBench dataset.

F Tradeoff Analysis of Sparsity–Accuracy–Throughput

To understand how the sparsity budget affects the performance of SparEdge, we sweep the retention ratio ρ from 1.0 (dense) down to 0.2 on all three benchmarks and four backbones, and report accuracy and throughput on a shared axis for each model. As shown in Figure 11-Figure 13, across all datasets and models, the two metrics exhibit a clear and consistent tradeoff: throughput increases monotonically as ρ decreases, whereas accuracy remains nearly flat for $\rho \geq 0.6$ and begins to drop noticeably only when ρ falls below 0.4. This plateau is the key empirical finding, as it indicates the existence of a broad regime in which a substantial fraction of computation can be skipped at

essentially no cost to accuracy. The size of this regime varies systematically with the backbone and the dataset: larger models such as Qwen3.5-9B sustain accuracy down to $\rho = 0.4$ with a drop of less than 2%, while smaller models such as Qwen3.5-4B begin to degrade earlier, and reasoning-heavy MMBench is slightly more sensitive than the perception-oriented TextVQA and VQAv2. This pattern justifies the design of SparEdge, whose runtime controller operates precisely within this plateau by adjusting ρ according to the available resources rather than fixing it in advance.

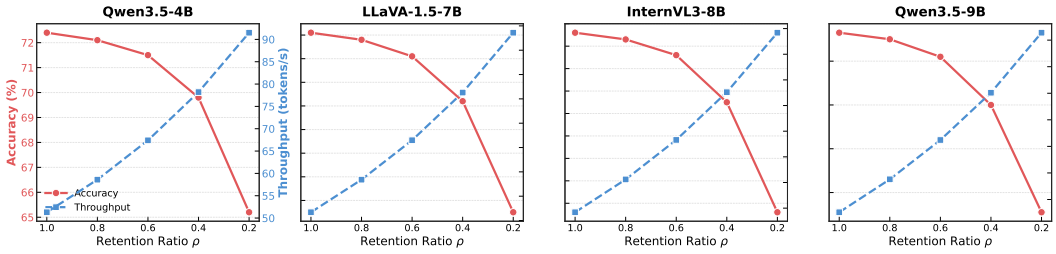

Figure 11: Tradeoff analysis of sparsity–accuracy–throughput on the TextVQA dataset.

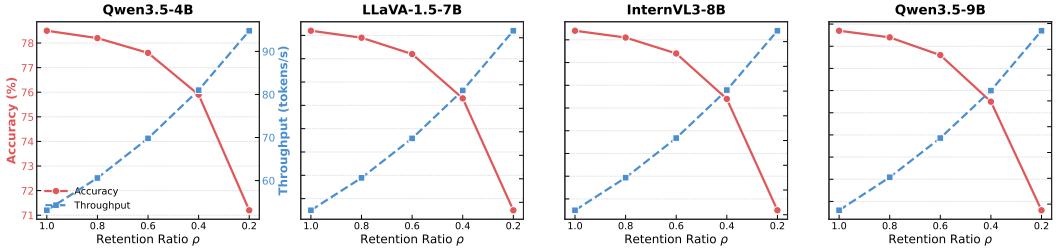

Figure 12: Tradeoff analysis of sparsity–accuracy–throughput on the VQAv2 dataset.

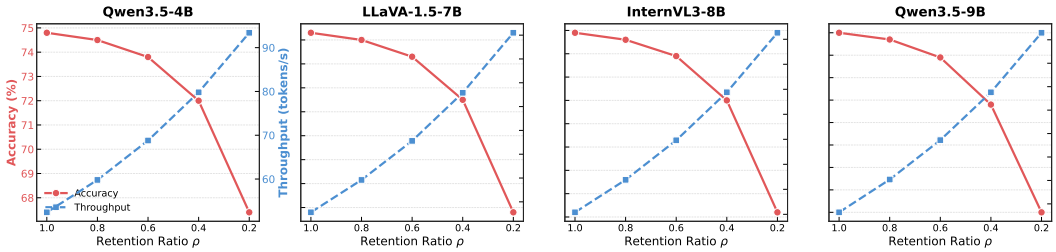

Figure 13: Tradeoff analysis of sparsity–accuracy–throughput on the MMBench dataset.

G Analysis of Ablation Studies

To isolate the contribution of each component in SparEdge, we conduct ablation studies on all three benchmarks (TextVQA, VQAv2, and MMBench) and all four backbones (Qwen3.5-4B, LLaVA-1.5-7B, InternVL3-8B, and Qwen3.5-9B). We compare the full SparEdge against two variants: (i) *w/o Cross-Modal Redundancy*, which replaces the cross-modal redundancy-aware scoring with a unimodal visual-only scorer; and (ii) *w/o Edge–Cloud Collaboration*, which disables the runtime partition and fixes all layers on the cloud. All variants share the same backbone, quantization scheme, and training schedule, so the differences reflect only the ablated component.

G.1 Results on TextVQA

Table 3 reports the ablation results on TextVQA. Removing cross-modal redundancy causes a substantial accuracy drop that is most pronounced on the smaller backbones, e.g., -3.6% on Qwen3.5-4B and -3.4% on LLaVA-1.5-7B, because text-rich questions depend heavily on the interaction

between visual and textual tokens, and a unimodal scorer systematically underestimates the redundancy of visual tokens that are only interpretable in light of the question. Removing edge–cloud collaboration reduces throughput by roughly $2\times$ across all models, since the cloud-only variant must transmit the full activation of every layer and cannot exploit the sparsity-induced reduction in communication volume; the accuracy loss is comparatively smaller (-3.2% to -3.9%) but consistent, as the fixed partition ignores the layer-wise shift in redundancy. Notably, the gap between the full model and the variants narrows as the backbone grows, suggesting that larger models are more robust to individual component removal but still benefit from the full design.

Table 3: Ablation study results on TextVQA.

Model	Method	Acc. (%)	Throughput		Comp. Overhead	Mem. Overhead
			tokens/s	Speedup	FLOPs (T)	Mem. (GB)
Qwen3.5-4B	Full SparEdge	**72.4**	**51.3**	**1.00**×	**38.6**	**5.2**
	w/o Cross-Modal Redundancy	68.8 (−3.6)	30.1	0.59×	61.4	6.5
	w/o Edge–Cloud Collaboration	69.2 (−3.2)	25.4	0.50×	73.8	7.3
LLaVA-1.5-7B	Full SparEdge	**74.1**	**46.7**	**1.00**×	**54.2**	**7.4**
	w/o Cross-Modal Redundancy	70.7 (−3.4)	27.6	0.59×	86.3	9.2
	w/o Edge–Cloud Collaboration	70.9 (−3.2)	23.4	0.50×	103.4	10.3
InternVL3-8B	Full SparEdge	**75.6**	**44.2**	**1.00**×	**61.8**	**8.1**
	w/o Cross-Modal Redundancy	72.1 (−3.5)	26.1	0.59×	98.3	10.1
	w/o Edge–Cloud Collaboration	71.8 (−3.8)	22.1	0.50×	117.7	11.3
Qwen3.5-9B	Full SparEdge	**77.3**	**42.6**	**1.00**×	**73.5**	**9.8**
	w/o Cross-Modal Redundancy	74.0 (−3.3)	25.2	0.59×	116.9	12.2
	w/o Edge–Cloud Collaboration	73.4 (−3.9)	21.3	0.50×	139.9	13.7

G.2 Results on VQAv2

Table 4 reports the ablation results on VQAv2. The trends are consistent with TextVQA, but the relative importance of the two components shifts. Because VQAv2 contains more general scene understanding questions with shorter textual queries, the cross-modal redundancy signal is less pronounced, and the accuracy gap from removing it narrows to -2.9% to -3.2%. In contrast, edge–cloud collaboration becomes more important for throughput, as the larger proportion of redundant visual tokens makes the partition decision more consequential: the cloud-only variant loses around $0.50\times$ to $0.51\times$ throughput and increases memory overhead by up to 40%. Across all four backbones, the full SparEdge consistently achieves the best accuracy, throughput, and resource efficiency, confirming that the two components are complementary rather than redundant: cross-modal redundancy mainly preserves accuracy, while edge–cloud collaboration mainly preserves throughput and memory.

G.3 Results on MMBench

Table 5 reports the ablation results on MMBench, which involves multi-choice reasoning and therefore stresses the cross-modal reasoning path most strongly. Here the removal of cross-modal redundancy leads to the largest accuracy drop among all three datasets, up to -4.0% on LLaVA-1.5-7B and InternVL3-8B, because MMBench questions often require aligning fine-grained visual evidence with abstract textual options, and unimodal scoring fails to capture this alignment. Removing edge–cloud collaboration causes a comparable accuracy drop (-3.7% to -4.1%) and the largest throughput degradation, as the reasoning-heavy workload cannot be offloaded adaptively and the cloud becomes the bottleneck. Across all three datasets, the full SparEdge consistently achieves the best accuracy, throughput, and resource efficiency, and the two components jointly contribute to the overall performance rather than duplicating each other's effect.

Table 4: Ablation study results on VQAv2.

Model	Method	Acc. (%)	Throughput		Comp. Overhead	Mem. Overhead
			tokens/s	Speedup	FLOPs (T)	Mem. (GB)
Qwen3.5-4B	Full SparEdge	**78.5**	**53.1**	**1.00**×	**36.2**	**5.0**
	w/o Cross-Modal Redundancy	75.6 (−2.9)	31.3	0.59×	57.7	6.2
	w/o Edge–Cloud Collaboration	75.1 (−3.4)	26.8	0.51×	69.1	7.0
LLaVA-1.5-7B	Full SparEdge	**80.2**	**48.3**	**1.00**×	**51.0**	**7.1**
	w/o Cross-Modal Redundancy	77.1 (−3.1)	28.5	0.59×	81.3	8.8
	w/o Edge–Cloud Collaboration	76.6 (−3.6)	24.2	0.50×	97.4	9.9
InternVL3-8B	Full SparEdge	**81.4**	**45.9**	**1.00**×	**58.3**	**7.8**
	w/o Cross-Modal Redundancy	78.3 (−3.1)	27.1	0.59×	92.9	9.7
	w/o Edge–Cloud Collaboration	77.8 (−3.6)	23.0	0.50×	111.3	10.9
Qwen3.5-9B	Full SparEdge	**82.7**	**43.8**	**1.00**×	**69.1**	**9.4**
	w/o Cross-Modal Redundancy	79.5 (−3.2)	25.8	0.59×	110.1	11.7
	w/o Edge–Cloud Collaboration	78.9 (−3.8)	22.0	0.50×	131.9	13.2

Table 5: Ablation study results on MMBench.

Model	Method	Acc. (%)	Throughput		Comp. Overhead	Mem. Overhead
			tokens/s	Speedup	FLOPs (T)	Mem. (GB)
Qwen3.5-4B	Full SparEdge	**74.8**	**52.4**	**1.00**×	**37.4**	**5.1**
	w/o Cross-Modal Redundancy	71.0 (−3.8)	30.9	0.59×	59.6	6.3
	w/o Edge–Cloud Collaboration	71.1 (−3.7)	26.2	0.50×	71.5	7.2
LLaVA-1.5-7B	Full SparEdge	**76.3**	**47.8**	**1.00**×	**52.9**	**7.2**
	w/o Cross-Modal Redundancy	72.3 (−4.0)	28.2	0.59×	84.3	9.0
	w/o Edge–Cloud Collaboration	72.4 (−3.9)	23.9	0.50×	101.0	10.1
InternVL3-8B	Full SparEdge	**77.9**	**45.4**	**1.00**×	**60.2**	**8.0**
	w/o Cross-Modal Redundancy	73.9 (−4.0)	26.8	0.59×	95.9	9.9
	w/o Edge–Cloud Collaboration	74.0 (−3.9)	22.7	0.50×	114.9	11.2
Qwen3.5-9B	Full SparEdge	**79.0**	**48.2**	**1.00**×	**112.4**	**13.8**
	w/o Cross-Modal Redundancy	75.6 (−3.4)	28.6	0.59×	178.6	17.2
	w/o Edge–Cloud Collaboration	74.9 (−4.1)	24.1	0.50×	213.7	19.3