Skim and Skip: Hierarchical Adaptive Inference for Efficient Multimodal Retrieval

Meng Gao[1,*] **Yizhen Zhang**[1,*] **Yang Ding**[1,*] **Ziqi Dai** **Shuoshuo Zhang**[1]
Junjie Wang[1] **Taiqiang Wu**[1] **Chufan Shi**[1] **Lei Ji**[2,†] **Jian Jiao**[2]
Linfeng Zhang[3,†] **Yeyun Gong**[2] **Yujiu Yang**[1,†]
[1]Tsinghua University [2]Microsoft [3]Shanghai Jiao Tong University
gaom24@mails.tsinghua.edu.cn leiji@microsoft.com
zhanglinfeng@sjtu.edu.cn yang.yujiu@sz.tsinghua.edu.cn

Abstract

Universal multimodal retrieval (UMR) increasingly adopts multimodal large language models (MLLMs) as unified embedding backbones, but their strong retrieval performance comes at substantial inference cost. Existing methods typically rely on uniformly dense inference, where all input tokens are processed through the entire model and matched using the final-layer [EOS] representation. However, this paradigm overlooks two key forms of heterogeneity in multimodal retrieval: token contributions to the final retrieval embedding are highly uneven, and different queries require markedly different amounts of inference depth. To address this, we propose **Skim and Skip (SAS)**, a hierarchical adaptive inference framework for efficient multimodal retrieval. SAS first performs token-level evidence selection to preserve only the input information most relevant to the final retrieval embedding, and then performs depth-adaptive inference to determine whether the current representation is already sufficient for reliable matching. Experiments on 12 MMEB retrieval tasks show that SAS retains about 99% of the dense baseline's average retrieval performance while reducing LLM-side FLOPs by up to **66.3%** and achieving up to **1.64**× end-to-end speedup. Our code is available at https://github.com/GaoMengGladys/SAS.

1 Introduction

Multimodal large language models (MLLMs) are increasingly used as unified backbones for multimodal retrieval (Jiang et al., 2024b; Zhang et al., 2025b; He et al., 2026). Benefiting from large-scale cross-modal pretraining, MLLM-based retrievers have achieved strong performance across zero-shot, fine-grained, and universal retrieval settings (Liu et al., 2024; Jiang et al., 2024b; He et al., 2026; Li et al., 2026; Huang et al., 2026; Yang et al., 2024b; Zhang et al., 2026b; Lin et al., 2024). However, this performance comes at substantial inference cost, as existing methods typically process every query with all tokens through the full depth of the model (Lyu et al., 2025; Xiao et al., 2025).

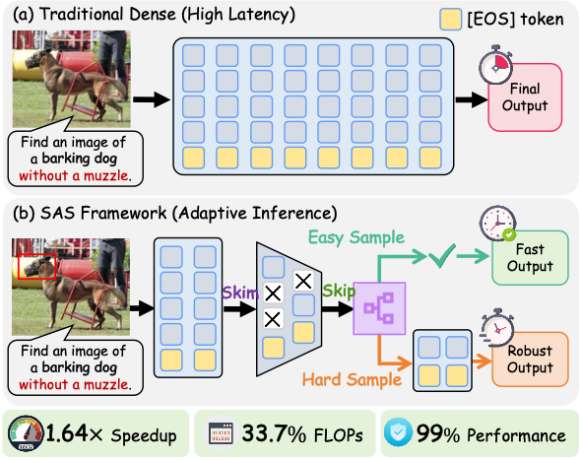


Figure 1: **Motivation of Skim and Skip (SAS).** Conventional MLLM retrieval applies uniformly dense inference to all tokens across all layers, resulting in high latency. In contrast, SAS performs hierarchical adaptive inference: it first filters retrieval-relevant information (*Skim*) and then allocates depth based on representation maturity (*Skip*), allowing easy queries to exit early while reserving deeper computation for harder ones.

The high inference cost primarily arises from two core computational factors: input information width and information processing depth. First, multimodal inputs—comprising visual and textual tokens—are long and processed uniformly, despite being inherently uneven, redundant, and noisy, which degrades the quality of the final retrieval embedding. Second, different queries require varying levels of processing depth, yet existing approaches often apply a fixed processing budget. Therefore, efficient retrieval hinges on two tightly coupled questions: which portions of the input are truly informative for constructing the representation, and

*Equal contribution. Meng Gao and Yizhen Zhang did this work during the internship at Microsoft Research Asia.
†Corresponding authors.

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 43873–43890
October 24-29, 2026 ©2026 Association for Computational Linguistics

how much computation is necessary for that representation to become sufficiently reliable.

Existing acceleration methods typically address these two issues in isolation, either compressing the input sequence (Yang et al., 2024b; Chen et al., 2024; Li et al., 2026; Wu et al., 2026a; Deniz et al., 2025; He and Chen, 2026), or executing fewer layers (Lyu et al., 2025; Yoo et al., 2026; Huang et al., 2024; Yang et al., 2026). However, standard pruning strategies developed for multimodal tasks such as VQA are poorly suited to retrieval: existing token-reduction methods largely overlook the critical role of the EOS embedding, causing severe performance degradation even at low compression ratios. Moreover, naive depth reduction also harms performance: while easy queries already achieve 55.0% Hit@1 at intermediate layer 22, hard queries reach only 7.2% on MSCOCO, indicating substantial variability in required processing depth. Furthermore, jointly learning what to preserve and when a representation suffices is inherently challenging, as it deviates from standard full-token supervision.

Motivated by this view, we propose **Skim and Skip (SAS)**, a hierarchical adaptive inference framework for efficient multimodal retrieval. SAS consists of two stages. **Skim** performs [EOS]-centric cross-modal information selection, preserving only the input information most relevant to the final retrieval embedding. **Skip** performs adaptive depth allocation, deciding whether the current representation is already mature enough for reliable retrieval. To motivate this design, we revisit the assumptions behind uniformly dense inference in Section 2, and then present SAS in Section 3.

Our main contributions are as follows:

- We identify two key forms of heterogeneity overlooked by uniform dense inference in MLLM retrieval: uneven token contributions to the final retrieval representation, and varying depth requirements across queries.
- We propose **Skim and Skip (SAS)**, a hierarchical adaptive inference framework that unifies [EOS]-centric cross-modal information selection and adaptive depth allocation for multimodal retrieval.
- We evaluate SAS on 12 multimodal retrieval benchmarks spanning five task types. SAS retains about 99% of the dense baseline's average performance while reducing LLM-side FLOPs by up to 66.3% and achieving up to 1.64× end-to-end speedup.

Top-k	Shallow (L0)	Middle (L14)	Deep (L27)
Top-10	69.6%	**77.8%**	58.3%
Top-20	86.3%	**87.9%**	75.1%

Table 1: Average [EOS] attention on 11 MMEB tasks.

Drop Ratio	CIRR	FashionIQ	NIGHTS	OVEN
0%	0.572	0.197	0.685	0.645
10%	0.591	0.204	0.659	0.647
20%	0.577	0.191	0.667	0.648
30%	0.586	0.194	0.660	0.633

Table 2: Negligible impact of random drops.

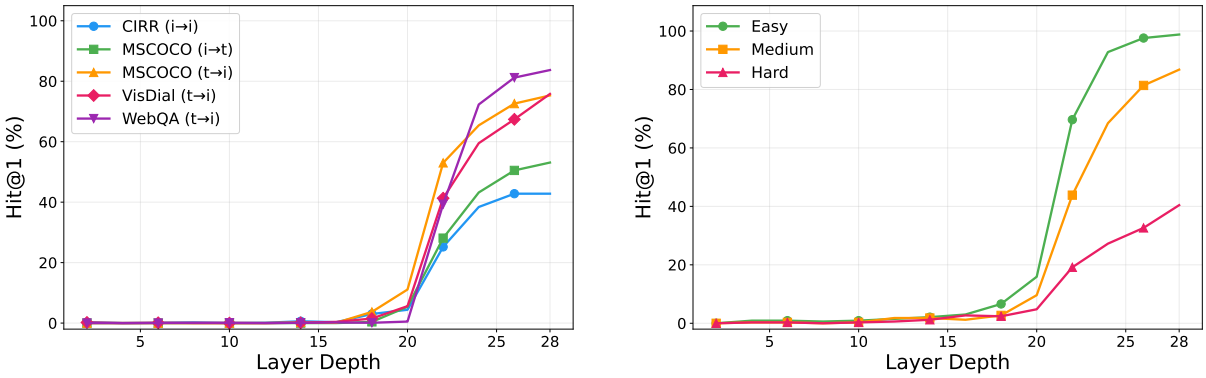
2 Revisiting Uniform Dense Inference in MLLM Retrieval

In MLLM-based retrieval, most approaches follow a uniformly dense inference paradigm, processing every query with the full set of input tokens through the entire model. While effective, this design implicitly assumes that all input information warrants equal computational effort and that all queries require the same depth of reasoning. To revisit this paradigm, we examine it from two perspectives: whether all tokens contribute equally to the final [EOS] retrieval embedding, and whether all queries require the same inference depth to form a reliable retrieval representation.

2.1 Are All Tokens Equally Informative?

We first examine how the [EOS] token aggregates retrieval evidence. Specifically, we analyze the attention from [EOS] to all input tokens across layers of VLM2Vec-V2.0. As shown in Table 1, at the middle layer (L14), where cross-modal interaction is strongest, the top-10 tokens capture 77.8% of the total [EOS] attention mass and the top-20 capture 87.9%. These results suggest that the final [EOS] embedding is driven primarily by a small subset of tokens, while the majority contribute little.

This view is reinforced by random token removal experiments. We randomly drop different proportions of tokens across multiple retrieval tasks (Table 2) and find that performance remains largely stable under 10–30% pruning, with slight gains on some tasks. This suggests that retrieval evidence is inherently sparse and that some tokens may even introduce distracting or irrelevant signals. Efficient retrieval inference should therefore identify the information that materially contributes to the final [EOS] retrieval embedding and preserve only that



(a) Per-layer retrieval performance across five tasks.

(b) Easy/Medium/Hard stratification (MSCOCO t→i).

Figure 2: Evidence against uniform depth allocation.

evidence for subsequent reasoning.

Importantly, this token selection problem cannot be reduced to a modality-specific heuristic. Token composition varies substantially across tasks. Such variability makes fixed modality-dependent rules brittle. Instead, retrieval-oriented token selection should rely on a unified cross-modal criterion that directly measures a token's relevance to the final [EOS] representation.

2.2 Do All Queries Require the Same Inference Depth?

We next examine whether all queries require the same inference depth. Layer-wise probing on VLM2Vec-V2.0 shows that retrieval signals do not emerge only at the final layer, but develop progressively in intermediate and late layers (Figure 2a). For example, on VisDial, the model already reaches 50.9% Hit@1 at layer 22, compared to 82.1% at the final layer, indicating that intermediate layers already encode substantial retrieval-relevant semantics before full-depth inference is completed.

However, the benefit of deeper computation is far from uniform across queries. To examine this heterogeneity, we partition queries into Easy, Medium, and Hard groups according to the final-layer Top-1/Top-2 similarity margin, and compare their layer-wise retrieval performance (Figure 2b). On MSCOCO text-to-image retrieval, Easy queries already achieve 55.0% Hit@1 at layer 22, whereas Hard queries reach only 7.2%. This disparity shows that retrieval representations mature at very different rates across samples.

At the same time, the presence of useful intermediate-layer signals does not by itself yield a practical adaptive-depth mechanism. Standard retrieval backbones are supervised only at the final layer, so intermediate representations, although informative, are not explicitly trained to function as standalone retrieval embeddings. Converting this latent depth redundancy into real inference savings therefore requires two steps: first, activating retrieval capability at a target intermediate layer through additional supervision; and second, deciding whether inference can terminate there based on the maturity of the current representation.

3 Methodology

The analysis in Section 2 shows that efficient multimodal retrieval requires two coupled decisions: what information to preserve and how much computation to allocate. Motivated by this view, we propose **Skim and Skip (SAS)**, a hierarchical adaptive inference framework for multimodal retrieval. For each query, SAS first applies **Skim** to retain the input information most relevant to the final [EOS] retrieval embedding, and then applies **Skip** to determine whether the current representation is already mature enough for early termination. Accordingly, Skim addresses *information selection*, while Skip addresses *depth allocation*.

3.1 Skim: [EOS]-centric Cross-modal Information Selection

The goal of **Skim** is to retain only the input information that materially contributes to the final [EOS] retrieval embedding, so that subsequent computation is concentrated on informative tokens. Because visual and textual token composition varies substantially across retrieval tasks, token selection must be defined under a unified cross-modal criterion. Rather than relying on local saliency or heuristic thresholds, Skim measures token importance by the

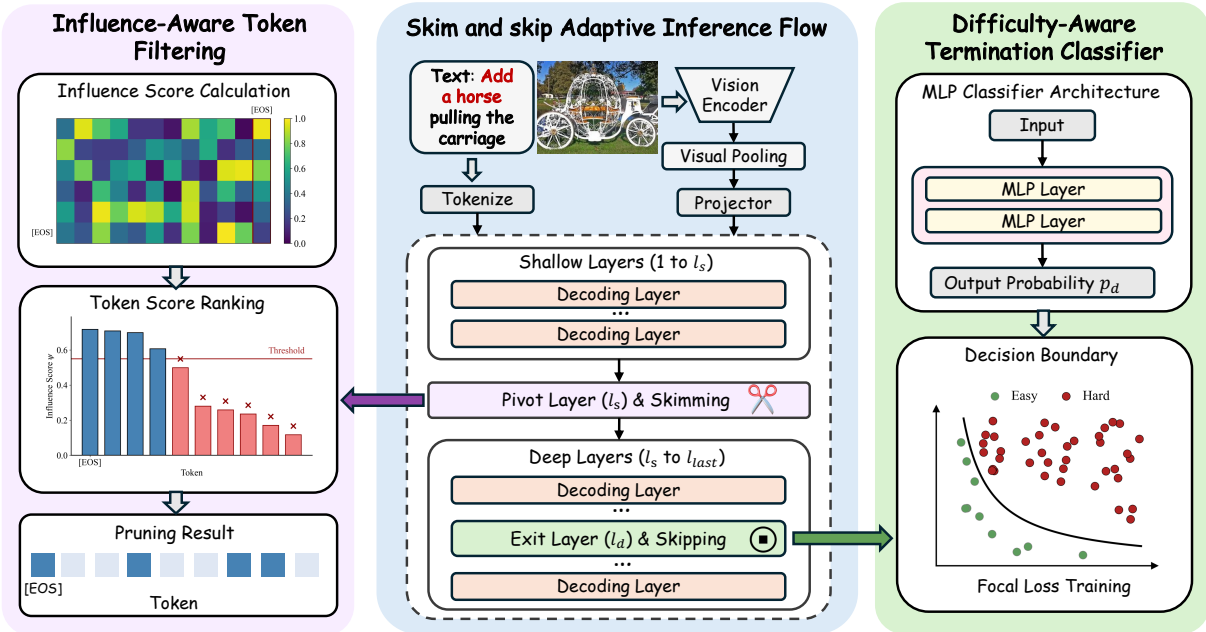


Figure 3: **Overview of SAS.** SAS treats retrieval inference as progressive evidence accumulation. **Left: Token-level evidence selection.** At the skim layer l_s, the Influence-Aware Filter estimates each token's marginal influence on the [EOS] embedding and removes low-contribution tokens, preserving only evidence-bearing information. **Right: Layer-level evidence sufficiency.** At the decision layer l_d, a learned sufficiency classifier determines whether the accumulated evidence is adequate for reliable retrieval; sufficiently mature queries exit early, while the rest continue to deeper layers.

perturbation that removing a token would induce on the [EOS] representation.

In practice, we first apply pooling to the ViT-encoded visual tokens to shorten the initial visual sequence before it is fed into the LLM. We then perform cross-modal token selection at layer l_s.

Local Perturbation of Token Removal. Consider attention head h at layer l. Let $o_{l,h}(e)$ denote the attention output at the [EOS] position, and let $\alpha_{l,h}(e \to j)$ denote the attention weight from [EOS] to token j. The output at [EOS] is

$$o_{l,h}(e) = \sum_i \alpha_{l,h}(e \to i)\, v_{l,h,i}, \tag{1}$$

where $v_{l,h,i}$ is the value vector of token i.

If token j is removed, the attention weights over the remaining tokens are re-normalized, perturbing the head output at [EOS]. The resulting exact perturbation is

$$\Delta o_{l,h}(e) = \frac{\alpha_{l,h}(e \to j)}{1 - \alpha_{l,h}(e \to j)} \Big(o_{l,h}(e) - v_{l,h,j} \Big). \tag{2}$$

This expression captures the immediate effect of removing a token on the [EOS] representation in the current layer. Importantly, the perturbation depends not only on the attention mass assigned to token j, but also on the discrepancy between its value vector and the aggregated head output. Attention alone is therefore insufficient to characterize a token's contribution to retrieval.

A Computable Upper Bound on Token Influence. The local perturbation first affects a single attention head and then propagates through the output projection $W_O^{l,h}$ to subsequent layers. To quantify its downstream impact, we consider the magnitude of the projected perturbation, $\|W_O^{l,h} \Delta o_{l,h}(e)\|_2$.

Computing this quantity exactly during inference is inefficient. Instead, we upper-bound it using the spectral norm $\sigma_O^{l,h}$ of $W_O^{l,h}$:

$$\|W_O^{l,h} \Delta o_{l,h}(e)\|_2 \le \sigma_O^{l,h} \frac{\alpha_j}{1 - \alpha_j} \times \Big(\|o_{l,h}(e)\|_2 + \|v_{l,h,j}\|_2 \Big), \tag{3}$$

where α_j abbreviates $\alpha_{l,h}(e \to j)$.

To obtain an inference-time computable score, we further bound $\|o_{l,h}(e)\|_2$ and $\|v_{l,h,j}\|_2$ using the spectral norm $\sigma_V^{l,h}$ of the value projection $W_V^{l,h}$ and its bias term $\mathbf{b}_V^{l,h}$. Aggregating the resulting bound over all H heads yields the following influence

score for token j at layer l:

$$\psi_l(j) = \sum_{h=1}^{H} \sigma_O^{l,h} \frac{\alpha_j}{1-\alpha_j} \cdot \Big[\sigma_V^{l,h}\big(S_{l,h} + \|\mathbf{h}_l[j]\|_2\big) + 2\|\mathbf{b}_V^{l,h}\|_2\Big], \tag{4}$$

where

$$S_{l,h} := \sum_i \alpha_{l,h}(e \to i)\, \|\mathbf{h}_l[i]\|_2, \tag{5}$$

and $\mathbf{h}_l[j]$ is the input hidden state of token j at layer l. The full derivation is provided in Appendix B.

This score can be interpreted as a computable upper bound on the perturbation that removing token j may induce on the subsequent evolution of the [EOS] representation. A larger score indicates greater importance to the final retrieval embedding, while a smaller score suggests that the token can be removed with limited effect.

Unified Cross-modal Selection. At skim layer l_s, we rank all tokens by their influence scores and remove those with the lowest values, preserving the tokens most important to the final [EOS] retrieval embedding. Because the score is defined directly by a token's effect on [EOS], visual and textual tokens can be ranked under the same criterion.

Moreover, the influence score is computed from attention weights and hidden states already available during the forward pass, introducing negligible overhead.

3.2 Skip: Adaptive Depth Allocation by Representation Maturity

The goal of **Skip** is to determine whether the current depth is already sufficient for reliable retrieval and to allocate additional computation only when necessary. This is non-trivial because standard retrieval backbones are supervised only on the final-layer [EOS] embedding. As a result, intermediate layers may already contain retrieval signals, but are not explicitly trained to serve as standalone retrieval embeddings.

To address this, Skip adopts a two-stage training paradigm: we first activate retrieval capability at the target decision layer, and then train a lightweight assessor to decide whether inference can stop there.

Stage I: Enabling Retrieval at the Intermediate Layer. To enable retrieval at the target layer l_d, we supervise its representation using feature-level distillation and contrastive learning. Intuitively, this stage encourages the intermediate embedding to both align with the final-layer semantic space and remain discriminative for retrieval.

Specifically, we treat the final-layer embedding $e^{(L)}$ as the teacher and the embedding at layer l_d, denoted by $e^{(l_d)}$, as the student. We first align them using mean squared error:

$$\mathcal{L}_{distill} = \mathbb{E}_{(q,c)\sim\mathcal{D}}\Big[\|e_q^{(L)} - e_q^{(l_d)}\|_2^2 + \|e_c^{(L)} - e_c^{(l_d)}\|_2^2\Big]. \tag{6}$$

This term reduces the representation gap between intermediate and final embeddings.

To ensure retrieval discriminability, we further apply InfoNCE supervision (Oord et al., 2018) to the intermediate embedding:

$$\mathcal{L}_{con}^{(l_d)} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(\mathrm{sim}(e_{q_i}^{(l_d)}, e_{c_i}^{(l_d)})/\tau)}{\sum_{j=1}^{N}\exp(\mathrm{sim}(e_{q_i}^{(l_d)}, e_{c_j}^{(l_d)})/\tau)}, \tag{7}$$

where $\mathrm{sim}(\cdot,\cdot)$ denotes cosine similarity and τ is a temperature parameter.

We combine the two objectives with a dynamic weight:

$$\mathcal{L}_{inter}(t) = \lambda(t)\mathcal{L}_{con}^{(l_d)} + (1-\lambda(t))\mathcal{L}_{distill}, \tag{8}$$

where $\lambda(t)$ increases monotonically during training. Early training emphasizes semantic alignment, while later training shifts toward retrieval discriminability.

To preserve final-layer performance, we retain the standard contrastive objective at layer L. The overall Stage-I loss is

$$\mathcal{L}_{\text{StageI}} = \mathcal{L}_{con}^{(L)} + \gamma\mathcal{L}_{inter}(t), \tag{9}$$

where γ controls the strength of intermediate supervision. After this stage, layer l_d is explicitly trained to serve as a usable retrieval representation, rather than merely implicitly carrying retrieval signals.

Stage II: Learning a Depth Sufficiency Assessor. Once the intermediate representation at layer l_d has matured sufficiently, we train a depth assessor to determine whether it is sufficient for reliable retrieval. In this stage, the backbone is frozen, and only a lightweight decision module is optimized.

The assessor does not predict the retrieval result itself. Instead, it predicts whether the current representation has accumulated enough evidence for

Model	Avg. (Ret.)	Time ↓	FLOPs ↓	Speedup ↑	Exit Rate ↑	Vis. Tok. ↓	Txt. Tok. ↓
Qwen2.5-VL-3B + CL	68.9 (100%)	678.4s	100%	1.00×	0%	100%	100%
Qwen2.5-VL-3B + SAS	**68.0 (99%)**	**471.3s** ↓30.5%	**36.6%** ↓63.4%	**1.44×**	**20%** ↑20%	**25%** ↓75%	**50%** ↓50%
Qwen2.5-VL-7B + CL	71.7 (100%)	1058.0s	100%	1.00×	0%	100%	100%
Qwen2.5-VL-7B + SAS	**70.9 (99%)**	**644.2s** ↓39.1%	**33.7%** ↓66.3%	**1.64×**	**26%** ↑26%	**25%** ↓75%	**50%** ↓50%

Table 3: Efficiency analysis of Qwen2.5-VL-3B/7B with standard contrastive learning (CL) and their SAS-accelerated counterparts. Avg. (Ret.) reports the mean Hit@1 and the corresponding performance retention rate relative to the CL baseline. Exit Rate indicates the proportion of early-exited samples, while Vis./Txt. Tok. denote the ratios of retained visual and textual tokens after skimming.

reliable matching, or whether deeper refinement is still needed. We formulate this as a binary classification problem over whether the sample should terminate at layer l_d.

The assessor uses three types of features. First, *uncertainty scalars*, including the Top-1 similarity score and the Top-1/Top-2 similarity margin from intermediate retrieval results, capture ranking confidence. Second, *modality indicators* distinguish text, image, and multimodal inputs, allowing the classifier to adapt to task-dependent distribution shifts. Third, *semantic features* are obtained by projecting the intermediate `[EOS]` representation to encode the current semantic context. Full architectural and training details are provided in Appendix C.

4 Experiment

Implementation Details. We instantiate SAS on top of Qwen2.5-VL-Instruct (3B and 7B) (Bai et al., 2025), and additionally evaluate its generality on the LLaVA-style training architecture (Liu et al., 2023). Stage I is conducted on 8 NVIDIA H100 GPUs with LoRA (Hu et al., 2022) ($r = 16$) for parameter-efficient fine-tuning. We train on the retrieval subset of MMEB (Jiang et al., 2024b) for 5,000 steps with a learning rate of 5×10^{-5}, a global batch size of 1,024, and a contrastive temperature of $\tau = 0.02$. Stage II is performed on a single NVIDIA A100 GPU, where we freeze the backbone and train the termination classifier for 1,000 steps using Focal Loss (Lin et al., 2017) with $\alpha = 0.8$ and $\gamma = 3.0$, a learning rate of 5×10^{-4}, and a batch size of 512. $\lambda(t)$ is initialized at 0 and annealed to 1.0 with a cosine schedule, gradually shifting the objective from distillation-based alignment to contrastive discrimination. We apply Vision Pooling at layer 1, Influence-Aware Token Filtering at layer 10 (l_s), and place adaptive depth allocation at layer 12 (l_d). For text tokens, we prune 50% of the sequence while preserving a 16-token guardrail.

Evaluation. We evaluate SAS on the MMEB retrieval split, covering 12 benchmarks across five representative multimodal retrieval task types. Following standard practice, we report Hit@1 as the accuracy metric and measure end-to-end inference latency on a single NVIDIA A100 GPU.

5 Main Results

We analyze the results from two perspectives: (1) efficiency relative to our reproduced dense models, and (2) retrieval performance compared with representative baselines.

5.1 Efficiency Comparison

To isolate the effect of SAS, we train dense 3B and 7B baselines under the same setting but without any adaptive acceleration, and use them as strict references for efficiency–accuracy comparison.

Efficiency–Accuracy Trade-off. As shown in Table 3, SAS substantially reduces computation while preserving retrieval quality. Compared with dense models, SAS retains nearly 99% of average retrieval performance (e.g., 71.7% → 70.9% for 7B, and 68.9% → 68.0% for 3B), while reducing FLOPs by up to 66.3%. This supports our claim that a considerable portion of token- and depth-level computation in dense MLLM retrieval is redundant and can be removed adaptively.

Revisiting the Parameter–Latency Trade-off. A notable finding is that the SAS-accelerated 7B model is even faster than the dense 3B model in end-to-end latency. In addition to outperforming the dense 3B model by about 2.0% in retrieval accuracy, the accelerated 7B model also achieves lower inference time. We attribute this to stronger representation quality in the larger model, which enables more reliable identification of easy queries and therefore more frequent early termination at l_d. This result suggests that, under adaptive inference, a larger model can simultaneously maintain better

Methods	$q^t \to c^i$				$q^i \to c^t$		$q^i \to c^i$	$q^t \to (c^i, c^t)$		$(q^i, q^t) \to c^i$		$(q^i, q^t) \to (c^i, c^t)$	Avg.
	VisDial	VN	COCO	Wiki-SS	VN	COCO	NIGHTS	WebQA	EDIS	CIRR	FIQ	OVEN	
UniIR (Wei et al., 2024)	42.2	74.3	68.5	12.2	76.8	72.1	66.2	89.6	79.2	51.3	40.2	69.4	61.8
GME (Zhang et al., 2024b)	48.1	74.7	68.1	**73.9**	78.3	63.1	67.0	88.8	91.8	44.2	32.9	72.3	66.9
VLM2Vec-2B (Jiang et al., 2024b)	74.3	73.1	73.4	54.2	73.7	68.5	66.3	85.9	81.2	46.8	14.0	68.3	65.0
VLM2Vec-7B (Jiang et al., 2024b)	81.9	80.5	77.2	62.3	81.2	73.9	67.6	88.3	85.7	51.1	17.1	66.5	69.4
VLM2Vec-V2.0 (Meng et al., 2025)	82.7	74.5	75.3	66.9	78.2	71.4	68.6	90.6	84.1	57.5	19.5	64.3	69.6
LamRA-Qwen2-7B (Liu et al., 2024)	61.3	70.4	72.2	69.7	**83.9**	73.7	65.6	81.0	85.9	51.7	**42.0**	82.0	70.0
LamRA-Qwen2.5-7B (Liu et al., 2024)	62.5	70.1	65.7	67.0	74.2	71.1	64.4	85.7	78.7	44.7	33.4	**84.8**	66.9
LLaVA-7B + CL	81.6	**88.9**	76.3	59.7	80.1	**75.4**	68.7	75.9	83.1	58.3	17.4	50.8	68.0
LLaVA-7B + SAS (ours)	81.6	75.6	76.3	57.3	80.0	73.5	**69.7**	87.2	75.2	56.2	17.4	54.5	67.1
Qwen2.5-VL-3B + CL	83.3	73.8	76.2	61.3	76.5	73.7	62.0	89.6	90.3	59.3	19.9	61.0	68.9
Qwen2.5-VL-3B + SAS (ours)	83.9	71.3	76.5	55.1	76.2	72.2	68.2	**90.7**	85.8	54.8	20.5	60.6	68.0
Qwen2.5-VL-7B + CL	83.9	77.4	**78.4**	71.1	78.6	73.9	68.0	90.5	**93.0**	54.1	25.0	66.8	**71.7**
Qwen2.5-VL-7B + SAS (ours)	**84.6**	75.1	77.8	64.5	78.8	74.4	68.0	90.1	90.0	**59.7**	22.5	65.6	70.9

Table 4: Performance comparison among Qwen2.5-VL-3B/7B with contrastive learning (CL), our SAS-accelerated variants, and state-of-the-art retrieval methods. All results are reported in Hit@1.

accuracy and achieve lower latency than a smaller dense counterpart.

5.2 Performance Comparison

Table 4 compares SAS with representative baselines, including dual-encoder retrievers and recent fine-tuned MLLM-based methods.

Despite aggressive token reduction and adaptive depth allocation, SAS remains highly competitive across diverse retrieval tasks. Our 7B model achieves an average Hit@1 of 70.9%, on par with strong recent MLLM-based retrieval baselines. This shows that adaptive sparse inference substantially improves efficiency at negligible accuracy cost, preserving the representational strength of large retrieval models.

SAS is particularly effective on tasks requiring fine-grained semantic matching. On benchmarks such as VisDial and WebQA, it consistently outperforms prior methods. We attribute this to the complementary roles of Skim and Skip: Skim suppresses irrelevant information to sharpen semantic focus, while Skip preserves deeper reasoning for genuinely difficult queries.

6 Discussion and Ablation

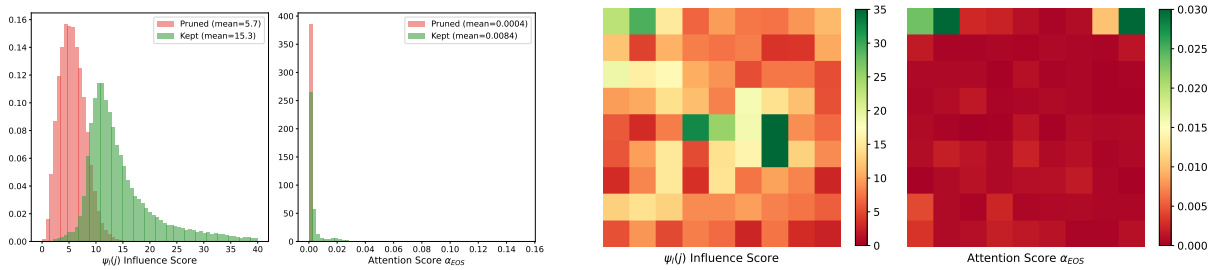
6.1 Qualitative Analysis of Influence-Aware Token Scoring

To better understand the behavior of the influence score $\psi_l(j)$ used in Skim, we visualize both its distributional behavior and its instance-level spatial selection pattern on the OVEN dataset, based on SAS-3B under 75% visual-token reduction, collecting score statistics from 1,000 queries (112,554 visual tokens in total).

As shown in Figure 4a, the proposed influence score yields a much clearer separation between kept and pruned tokens, with limited distributional overlap. In contrast, raw [EOS] attention collapses most tokens into a narrow range near zero, making the two groups barely distinguishable. This suggests that raw attention, as an instantaneous per-head signal, does not provide a stable pruning boundary. By aggregating perturbation effects across heads and accounting for downstream propagation, the influence score more faithfully captures each token's marginal contribution to the retrieval embedding.

Figure 4b presents a representative spatial score map. The influence score highlights broader and more spatially coherent high-score regions, whereas raw attention responds only to a few isolated peaks. This indicates that retrieval-relevant evidence is often distributed across multiple tokens that jointly shape the [EOS] representation, rather than concentrated in a small number of extreme attention maxima—a pattern that generic attention-based pruning can easily miss.

We further ablate two design choices behind Skim. The selection layer l_s determines when evidence filtering is applied: pruning too early can discard tokens before sufficient cross-modal integration, whereas later pruning reduces the potential computational savings. We select $l_s = 10$, which achieves the highest final-layer Hit@1 in this ablation; full results are provided in Appendix D. We also compare Skim against alternative token-reduction strategies in Appendix E. Under the evaluated configurations, SAS achieves 68.0% Hit@1 with 75% visual-token and 50% textual-token reduction, while FastV and DART achieve 60.6%



(a) Score distribution.

(b) Spatial score map.

Figure 4: Qualitative comparison of influence-aware scoring vs. raw [EOS] attention.

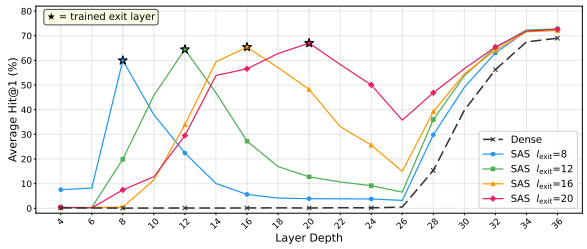


Figure 5: Performance under different l_d choices.

Components		Performance	Efficiency	
Skim	Skip	Avg. ↑	FLOPs ↓	Speedup ↑
✗	✗	68.9	100.0%	1.00×
✓	✗	68.0	42.2%	1.26×
✗	✓	68.1	87.9%	1.17×
✓	✓	**68.0**	**36.6%**	**1.44×**

Table 5: Ablation Study of SAS Components.

and 59.0%, respectively, and random text pruning achieves 58.3%.

6.2 Exit Layer Selection

The exit layer l_d determines where the depth sufficiency check is applied. A shallower exit saves more computation but may act on immature representations, whereas a deeper exit is more reliable but reduces the opportunity for early termination.

Figure 5 compares per-layer retrieval performance for SAS variants trained with $l_d \in \{8, 12, 16, 20\}$. Supervising a specific layer consistently boosts retrieval performance at that depth (marked by $\star$), confirming that Stage I effectively activates retrieval capability at the target layer. Among all choices, $l_d = 12$ gives the best trade-off: the intermediate layer reaches 50.6% average Hit@1, while the final layer achieves 69.4%. Earlier exits yield weaker intermediate retrieval representations, whereas overly late exits diminish the practical benefit of adaptive termination. In particular, setting $l_d = 13$ performs poorly, likely because there is too little depth remaining after the selection layer.

The full quantitative comparison is provided in Table 8. We further verify this effect through embedding-space visualization in Appendix J: after Stage I training, queries and their ground-truth candidates form much more coherent semantic clusters at l_d, whereas they are clearly separated before training. Additional comparisons of different supervision strategies are also included there.

6.3 Component Analysis

Table 5 reports a progressive ablation of the two hierarchical decisions in SAS. Evidence selection alone brings a 1.26× speedup with minimal performance loss (68.0% vs. 68.9%), indicating substantial token redundancy, while depth adaptation alone yields a 1.17× speedup, suggesting that many queries can terminate before the final layer. Combining the two leads to a 1.44× speedup at the same 68.0% accuracy, showing that token selection and depth allocation address complementary sources of computation and work effectively together.

7 Conclusion

In this paper, we introduce **Skim and Skip (SAS)**, a hierarchical adaptive inference framework designed for universal multimodal retrieval. To overcome the computational bottlenecks inherent in standard dense retrieval, SAS reconfigures the inference paradigm by integrating two complementary strategies: *Influence-Aware Token Filtering*, which effectively prunes redundant tokens to mitigate quadratic complexity, and *Difficulty-Aware Early Exit*, which allows easy samples to terminate at intermediate layers. Comparisons with strong dense MLLM backbones across multiple benchmarks demonstrate that SAS substantially improves inference efficiency while preserving competitive retrieval performance.

Limitations

Despite the encouraging results, we have not yet fully optimized the framework along two practical dimensions.

First, our prototype relies on standard dense attention kernels. Token skimming yields variable-length sequences, so smaller GEMMs and padding waste make wall-clock gains fall short of the theoretical FLOPs reduction. This is a hardware-efficiency gap rather than a method limitation, and can be narrowed by length-bucketed batching or variable-length attention kernels.

Second, we evaluate SAS only on retrieval and embedding workloads, not multimodal generation tasks. Extending to VQA or captioning complicates KV-cache handling and requires systematic choices of pruning and exit locations across backbones. Since SAS adds only lightweight control without modifying the backbone, we expect limited additional calibration to enable robust configurations for generative settings.

Acknowledgements

This work was partly supported by the National Key Research and Development Program of China (No. 2024YFB2808903), the Shenzhen Science and Technology Program (ZDCY20250901103533010).

References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2022. Token merging: Your vit but faster. *ArXiv*, abs/2210.09461.

Haonan Chen, Liang Wang, Nan Yang, Yutao Zhu, Ziliang Zhao, Furu Wei, and Zhicheng Dou. 2025. mme5: Improving multimodal multilingual embeddings via high-quality synthetic data. *arXiv preprint arXiv:2502.08468*.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*.

Xuanming Cui, Hong-You Chen, Hao Yu, Hao Yuan, Zihao Wang, Shlok Kumar Mishra, Hanchao Yu, Yonghuan Yang, Jun Xiao, Ser-Nam Lim, and 1 others. 2025. Reason to contrast: A cascaded multimodal retrieval framework. *arXiv preprint arXiv:2602.23369*.

Omer Faruk Deniz, Ruiyu Mao, Ruochen Li, Yapeng Tian, and Latifur Khan. 2025. Vision token reduction via attention-driven self-compression for efficient multimodal large language models. In *IEEE International Conference on Big Data (BigData)*.

Yuxiang Duan, Ao Li, Yingqin Li, Luyu Li, and Pengwei Wang. 2025. Gridprune: From" where to look" to" what to select" in visual token pruning for mllms. *arXiv preprint arXiv:2511.10081*.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, C'eline Hudelot, and Pierre Colombo. 2024. Colpali: Efficient document retrieval with vision language models. *ArXiv*, abs/2407.01449.

Chenwei He, Xiangzhao Hao, Tianyu Yang, Yuxiang Ma, Yuheng Jia, Lingxiang Wu, Chaoyang Zhao, Haiyun Guo, and Jinqiao Wang. 2026. Plume: Latent reasoning based universal multimodal embedding. *arXiv preprint arXiv:2604.02073*.

Jialuo He and Huangxun Chen. 2026. Energy-driven adaptive visual token pruning for efficient vision-language models. *arXiv preprint arXiv:2603.05950*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Haohang Huang, Xuan Lu, Mingyi Su, Xuan Zhang, Ziyan Jiang, Ping Nie, Kai Zou, Tomas Pfister, Wenhu Chen, Wei Zhang, and 1 others. 2026. Mmeb-v3: Measuring the performance gaps of omni-modality embedding models. *arXiv preprint arXiv:2604.23321*.

Lianming Huang, Shangyu Wu, Yufei Cui, Ying Xiong, Xue Liu, Tei-Wei Kuo, Nan Guan, and Chun Jason Xue. 2024. Raee: A robust retrieval-augmented early exiting framework for efficient inference. *arXiv preprint arXiv:2405.15198*.

Yatai Ji, Junjie Wang, Yuan Gong, Lin Zhang, Yanru Zhu, Hongfa Wang, Jiaxing Zhang, Tetsuya Sakai, and Yujiu Yang. 2023. MAP: Multimodal uncertainty-aware vision-language pre-training model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023*, pages 23262–23271. IEEE.

Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. 2024a. E5-v: Universal embeddings with multimodal large language models. *ArXiv*, abs/2407.12580.

Ziyan Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhu Chen. 2024b. Vlm2vec: Training vision-language models for massive multimodal embedding tasks. *ArXiv*, abs/2410.05160.

Jiwan Kim, Kibum Kim, Wonjoong Kim, Byung-Kwan Lee, and Chanyoung Park. 2026. Why and when visual token pruning fails? a study on relevant visual information shift in mllms decoding. *arXiv preprint arXiv:2604.12358*.

Fanheng Kong, Jingyuan Zhang, Yahui Liu, Hongzhi Zhang, Shi Feng, Xiaocui Yang, Daling Wang, Yu Tian, Qi Wang, Fuzheng Zhang, and Guorui Zhou. 2025. Modality curation: Building universal embeddings for advanced multimodal information retrieval. *ArXiv*, abs/2505.19650.

Zhibin Lan, Liqiang Niu, Fandong Meng, Jie Zhou, and Jinsong Su. 2025. Llave: Large language and vision embedding models with hardness-weighted contrastive learning. *ArXiv*, abs/2503.04812.

Qi Li, Yanzhe Zhao, Yongxin Zhou, Yameng Wang, Yandong Yang, Yuanjia Zhou, Jue Wang, Zuojian Wang, and Jinxiang Liu. 2026. Magic-mm-embedding: Towards visual-token-efficient universal multimodal embedding with mllms. *arXiv preprint arXiv:2602.05275*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and 1 others. 2024. Rho-1: Not all tokens are what you need. *arXiv preprint arXiv:2404.07965*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pages 34892–34916.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. 2020. Fastbert: a self-distilling bert with adaptive inference time. *ArXiv*, abs/2004.02178.

Yikun Liu, Pingan Chen, Jiayin Cai, Xiaolong Jiang, Yao Hu, Jiangchao Yao, Yanfeng Wang, and Weidi Xie. 2024. Lamra: Large multimodal model as your advanced retrieval assistant. *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4015–4025.

Bo Lv, Yasheng Sun, Junjie Wang, and Haoxiang Shi. 2026. OneLatent: Single-token compression for visual latent reasoning. *CoRR*, abs/2602.13738.

Yibo Lyu, Rui Shao, Gongwei Chen, Yijie Zhu, Weili Guan, and Liqiang Nie. 2025. Puma: Layer-pruned language model for efficient unified multimodal retrieval with modality-adaptive learning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 7653–7662.

Rui Meng, Ziyan Jiang, Ye Liu, Mingyi Su, Xinyi Yang, Yuepeng Fu, Can Qin, Zeyuan Chen, Ran Xu, Caiming Xiong, Yingbo Zhou, Wenhu Chen, and Semih Yavuz. 2025. Vlm2vec-v2: Advancing multimodal embedding for videos, images, and visual documents. *ArXiv*, abs/2507.04590.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*.

Yiming Ren, Yujiu Yang, and Junjie Wang. 2026. Mitigating the reasoning tax in vision-language fine-tuning with input-adaptive depth aggregation. *CoRR*, abs/2603.26330.

Siyu Sun, Jing Ren, Zhaohe Liao, Dongxiao Mao, Xiangyuan Ren, Yiyi Zhang, Haohua Zhao, Weixiong Lin, Jiang Shaohua, Liqing Zhang, and 1 others. 2026. Bottleneck tokens for unified multimodal retrieval. *arXiv preprint arXiv:2604.11095*.

Raghuveer Thirukovalluru, Rui Meng, Ye Liu, K Karthikeyan, Mingyi Su, Ping Nie, Semih Yavuz, Yingbo Zhou, Wenhu Chen, and Bhuwan Dhingra. 2025. Breaking the batch barrier (b3) of contrastive learning via smart batch mining. *ArXiv*, abs/2505.11293.

Cong Wei, Yang Chen, Haonan Chen, Hexiang Hu, Ge Zhang, Jie Fu, Alan Ritter, and Wenhu Chen. 2024. Uniir: Training and benchmarking universal multimodal information retrievers. In *European Conference on Computer Vision*, pages 387–404. Springer.

Zichen Wen, Yifeng Gao, Shaobo Wang, Junyuan Zhang, Qintong Zhang, Weijia Li, Conghui He, and Linfeng Zhang. 2025. Stop looking for important tokens in multimodal language models: Duplication matters more. *ArXiv*, abs/2502.11494.

Hao Wu, Yingqi Fan, Jinyang Dai, Junlong Tong, Yunpu Ma, and Xiaoyu Shen. 2026a. Hidrop: Hierarchical vision token reduction in mllms via late injection, concave pyramid pruning, and early exit. *arXiv preprint arXiv:2602.23699*.

Jie Wu, Haoling Li, Xin Zhang, Jiani Guo, Jane Luo, Steven Liu, Yangyu Huang, Ruihang Chu, Scarlett Li, and Yujiu Yang. 2026b. X-coder: Advancing competitive programming with fully synthetic tasks, solutions, and tests. *Preprint*, arXiv:2601.06953.

Zilin Xiao, Qi Ma, Mengting Gu, Chun-cheng Jason Chen, Xintao Chen, Vicente Ordonez, and Vijai Mohan. 2025. Metaembed: Scaling multimodal retrieval at test-time with flexible late interaction. *arXiv preprint arXiv:2509.18095*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy J. Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. In *Annual Meeting of the Association for Computational Linguistics*.

Runming Yang, Taiqiang Wu, Jiahao Wang, Pengfei Hu, Yik-Chung Wu, Ngai Wong, and Yujiu Yang. 2024a. Llm-neo: Parameter efficient knowledge distillation for large language models. *arXiv preprint arXiv:2411.06839*.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. 2024b. Visionzip: Longer is better but not necessary in vision language models. *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19792–19802.

Zebin Yang, Yijiahao Qi, Tong Xie, Bo Yu, Shaoshan Liu, and Meng Li. 2026. Dysl-vla: Efficient vision-language-action model inference via dynamic-static layer-skipping for robot manipulation. *arXiv preprint arXiv:2602.22896*.

Sangmin Yoo, Srikanth Malla, Chiho Choi, Wei D Lu, and Joon Hee Choi. 2026. Adept: Adaptive dynamic early-exit process for transformers. *arXiv preprint arXiv:2601.03700*.

Jianrui Zhang, Anirudh Sundara Rajan, Brandon Han, Soochahn Lee, Sukanta Ganguly, and Yong Jae Lee. 2026a. Reasoning-augmented representations for multimodal retrieval. *arXiv preprint arXiv:2602.07125*.

Kai Zhang, Yi Luan, Hexiang Hu, Kenton Lee, Siyuan Qiao, Wenhu Chen, Yu Su, and Ming-Wei Chang. 2024a. Magiclens: Self-supervised image retrieval with open-ended instructions. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pages 59403–59420. PMLR / OpenReview.net.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. 2025a. Beyond attention or similarity: Maximizing conditional diversity for token pruning in mllms. *ArXiv*, abs/2506.10967.

Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. 2024b. Gme: Improving universal multimodal retrieval by multimodal llms. *ArXiv*, abs/2412.16855.

Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. 2025b. Bridging modalities: Improving universal multimodal retrieval by multimodal large language models. *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9274–9285.

Xinchen Zhang, Bowei Liu, Jiale Liu, Chufan Shi, Yizhen Zhang, Junhong Liu, Youliang Zhang, Zhiheng Li, Yujiu Yang, and Ling Yang. 2026b. Omniverifier-m1: Multimodal meta-verifier with explicit structured recalibration. *arXiv preprint arXiv:2605.28805*.

Xinchen Zhang, Xiaoying Zhang, Youbin Wu, Yanbin Cao, Renrui Zhang, Ruihang Chu, Ling Yang, Yujiu Yang, and Guang Shi. 2026c. Generative universal verifier as multimodal meta-reasoner. In *International Conference on Learning Representations*, volume 2026, pages 109211–109243.

Yuan Zhang, Chunkai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. 2024c. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *ArXiv*, abs/2410.04417.

Junjie Zhou, Zheng Liu, Ze Liu, Shitao Xiao, Yueze Wang, Bo Zhao, Chen Jason Zhang, Defu Lian, and Yongping Xiong. 2024. Megapairs: Massive data synthesis for universal multimodal retrieval. *ArXiv*, abs/2412.14475.

Xin Zou, Di Lu, Yizhou Wang, Yibo Yan, Yuanhuiyi Lyu, Xu Zheng, Linfeng Zhang, and Xuming Hu. 2025. Don't just chase "highlighted tokens" in mllms: Revisiting visual holistic context retention. *ArXiv*, abs/2510.02912.

A Detailed Related Work

A.1 Multimodal Retrieval

Multimodal retrieval has evolved from dual-encoder models such as CLIP (Radford et al., 2021) and uncertainty-aware extensions (Ji et al., 2023) toward MLLM-based universal embedding frameworks. Early efforts such as E5-V (Jiang et al., 2024a), VLM2Vec (Jiang et al., 2024b), and GME (Zhang et al., 2024b) demonstrated that MLLMs can be repurposed as strong zero-shot retrievers, while LamRA (Liu et al., 2024) further improved alignment through retrieval-oriented tuning. More recent works continue to strengthen MLLM retrieval through universal embeddings (Li et al., 2026; He et al., 2026), flexible late interaction (Xiao et al., 2025), bottleneck representations (Sun et al., 2026), and reasoning-enhanced retrieval features (Cui et al., 2025; Zhang et al., 2026a,c). For fine-grained scenarios, ColPali (Faysse et al., 2024) introduced multi-vector document representations derived directly from page images.

Beyond architectural design, another line of work improves retrieval representations through data scaling and training strategies. This includes expanding multimodal supervision via synthetic data (Zhou et al., 2024; Chen et al., 2025; Wu et al., 2026b) and self-supervised instruction-based retrieval (Zhang et al., 2024a), as well as improving contrastive learning with modality-aware curation (Kong et al., 2025), smart hard-negative mining (Thirukovalluru et al., 2025), and hardness-aware optimization (Lan et al., 2025). Benchmark efforts such as MMEB-V3 (Huang et al., 2026) further highlight the remaining performance gaps of omni-modality embedding models. However, these approaches generally retain uniform dense inference at test time, applying the same amount of computation to all queries regardless of difficulty. In contrast, our work focuses on adaptive inference for multimodal retrieval by dynamically adjusting computation based on the query's information content and depth requirements.

A.2 Token Compression and Adaptive Computation

To reduce the computational cost of long multimodal sequences, prior work has extensively studied token compression, especially on the vision side. Attention-based methods such as FastV (Chen et al., 2024), SparseVLM (Zhang et al., 2024c), and ADSC (Deniz et al., 2025) rank tokens by importance scores, but often remain sensitive to local attention patterns. Similarity-based methods, including ToMe (Bolya et al., 2022) and DART (Wen et al., 2025), merge or select representative tokens based on feature similarity, but are often weakly coupled to downstream retrieval objectives. More recent approaches incorporate stronger adaptivity through text conditioning, hierarchical reduction, or energy-aware scoring, as in CDPruner (Zhang et al., 2025a), GridPrune (Duan et al., 2025), HoloV (Zou et al., 2025), HiDrop (Wu et al., 2026a), and EAdaPrune (He and Chen, 2026). Nevertheless, these methods mainly focus on reducing token count, typically under a predefined compression budget. Recent analysis also shows that visual token pruning can fail when relevant evidence shifts during inference (Kim et al., 2026); selectivity beyond direct pruning has also been explored (Lv et al., 2026).

A complementary line of work studies adaptive computation through early exit or layer skipping. In language models, methods such as FastBERT (Liu et al., 2020), DeeBERT (Xin et al., 2020), and ADEPT (Liu et al., 2020; Xin et al., 2020; Yoo et al., 2026; Yang et al., 2024a) dynamically reduce inference depth based on prediction confidence. Related ideas have also been explored in retrieval-augmented and multimodal settings (Huang et al., 2024; Yang et al., 2026; Ren et al., 2026). However, directly transferring these techniques to multimodal retrieval is non-trivial, since retrieval depends on the stability of continuous embeddings rather than discrete label predictions. Unlike prior work that treats token reduction and depth reduction separately, our method formulates efficient multimodal retrieval as a unified adaptive inference problem that jointly determines what information to keep and how deeply it should be processed.

B Theoretical Analysis of Influence-Aware Token Skimming

In this section, we provide the complete derivation for the Influence-Aware Token Filtering score $\psi_l(j)$ (Eq. (4) in the main text) and the bounding analysis for retrieval stability under token pruning.

B.1 Derivation of Influence Score

Problem Formulation and Notation. We consider a Transformer-based MLLM backbone with L layers. For a specific query encoding process,

we focus on the self-attention mechanism at layer l with H heads. Let $\mathbf{h}_l[i] \in \mathbb{R}^D$ denote the hidden state of token i at layer l, and let e denote the index of the [EOS] token used for pooling the final query embedding. For a specific head h, the projection matrices are defined as $W_Q^{l,h}, W_K^{l,h}, W_V^{l,h} \in \mathbb{R}^{d\times D}$ and the output projection matrix as $W_O^{l,h} \in \mathbb{R}^{D\times d}$, where $d = D/H$. Crucially, we incorporate the bias term for the value projection, denoted as $\mathbf{b}_V^{l,h} \in \mathbb{R}^d$. Accordingly, the value vector for token i is computed as:

$$\mathbf{v}_{l,h,i} = W_V^{l,h}\mathbf{h}_l[i] + \mathbf{b}_V^{l,h}, \tag{10}$$

and the attention weight from the [EOS] token to token i is denoted by $\alpha_{l,h}(e \to i)$.

Local Perturbation Analysis. Consider the self-attention output at the [EOS] position for a single head h at layer l. The original output $\mathbf{o}_{l,h}(e)$ is a weighted sum of value vectors:

$$\mathbf{o}_{l,h}(e) = \sum_i \alpha_{l,h}(e \to i)\mathbf{v}_{l,h,i}. \tag{11}$$

When a token j is removed (skimmed), the attention mechanism re-normalizes the weights of the remaining tokens. Assuming the pre-softmax logits remain unchanged for the surviving tokens, the new attention weight for any remaining token i ($i \neq j$) becomes:

$$\alpha'_{l,h}(e \to i) = \frac{\alpha_{l,h}(e \to i)}{1 - \alpha_{l,h}(e \to j)}. \tag{12}$$

The perturbed output $\mathbf{o}'_{l,h}(e)$ after removing token j is:

$$\mathbf{o}'_{l,h}(e) = \sum_{i\neq j} \frac{\alpha_{l,h}(e \to i)}{1 - \alpha_{l,h}(e \to j)}\mathbf{v}_{l,h,i}. \tag{13}$$

The exact perturbation vector $\Delta\mathbf{o}_{l,h}(e) = \mathbf{o}'_{l,h}(e) - \mathbf{o}_{l,h}(e)$ is derived as follows:

$$\begin{aligned}\Delta\mathbf{o}_{l,h}(e) &= \frac{\sum_{i\neq j}\alpha_{l,h}(e \to i)\mathbf{v}_{l,h,i}}{1 - \alpha_{l,h}(e \to j)} - \mathbf{o}_{l,h}(e) \\ &= \frac{\mathbf{o}_{l,h}(e) - \alpha_{l,h}(e \to j)\mathbf{v}_{l,h,j}}{1 - \alpha_{l,h}(e \to j)} - \mathbf{o}_{l,h}(e) \\ &= \frac{\alpha_{l,h}(e \to j)}{1 - \alpha_{l,h}(e \to j)}(\mathbf{o}_{l,h}(e) - \mathbf{v}_{l,h,j}).\end{aligned} \tag{14}$$

This recovers Eq. (2) presented in the main text.

Upper Bound on Influence Score. To efficiently select tokens for pruning, we derive a computable upper bound for the magnitude of the perturbation projected by $W_O^{l,h}$. We utilize the spectral norms of the weight matrices, denoted as $\sigma_V^{l,h} = \|W_V^{l,h}\|_2$ and $\sigma_O^{l,h} = \|W_O^{l,h}\|_2$.

First, we bound the term $\|\mathbf{o}_{l,h}(e) - \mathbf{v}_{l,h,j}\|_2$ using the triangle inequality:

$$\|\mathbf{o}_{l,h}(e) - \mathbf{v}_{l,h,j}\|_2 \leq \|\mathbf{o}_{l,h}(e)\|_2 + \|\mathbf{v}_{l,h,j}\|_2. \tag{15}$$

We expand the norm of the value vector $\mathbf{v}_{l,h,j}$:

$$\begin{aligned}\|\mathbf{v}_{l,h,j}\|_2 &= \|W_V^{l,h}\mathbf{h}_l[j] + \mathbf{b}_V^{l,h}\|_2 \\ &\leq \sigma_V^{l,h}\|\mathbf{h}_l[j]\|_2 + \|\mathbf{b}_V^{l,h}\|_2.\end{aligned} \tag{16}$$

Next, we expand the norm of the original output $\mathbf{o}_{l,h}(e)$. Noting that $\sum_i \alpha_{l,h}(e \to i) = 1$, the bias term sums directly:

$$\begin{aligned}&\|\mathbf{o}_{l,h}(e)\|_2 \\ &= \left\|\sum_i \alpha_{l,h}(e \to i)(W_V^{l,h}\mathbf{h}_l[i] + \mathbf{b}_V^{l,h})\right\|_2 \\ &\leq \sigma_V^{l,h}\sum_i \alpha_{l,h}(e \to i)\|\mathbf{h}_l[i]\|_2 + \|\mathbf{b}_V^{l,h}\|_2 \\ &= \sigma_V^{l,h}S_{l,h} + \|\mathbf{b}_V^{l,h}\|_2,\end{aligned} \tag{17}$$

where we defined $S_{l,h} := \sum_i \alpha_{l,h}(e \to i)\|\mathbf{h}_l[i]\|_2$.

Substituting these bounds back into the perturbation equation:

$$\begin{aligned}&\|W_O^{l,h}\Delta\mathbf{o}_{l,h}(e)\|_2 \\ &\leq \sigma_O^{l,h}\frac{\alpha_{l,h}(e \to j)}{1 - \alpha_{l,h}(e \to j)}(\|\mathbf{o}_{l,h}(e)\|_2 + \|\mathbf{v}_{l,h,j}\|_2) \\ &\leq \sigma_O^{l,h}\frac{\alpha_{l,h}(e \to j)}{1 - \alpha_{l,h}(e \to j)}\Bigg[\left(\sigma_V^{l,h}S_{l,h} + \|\mathbf{b}_V^{l,h}\|_2\right) \\ &\quad + \left(\sigma_V^{l,h}\|\mathbf{h}_l[j]\|_2 + \|\mathbf{b}_V^{l,h}\|_2\right)\Bigg] \\ &= \sigma_O^{l,h}\frac{\alpha_{l,h}(e \to j)}{1 - \alpha_{l,h}(e \to j)} \\ &\quad \cdot \left(\sigma_V^{l,h}\left(S_{l,h} + \|\mathbf{h}_l[j]\|_2\right) + 2\|\mathbf{b}_V^{l,h}\|_2\right).\end{aligned} \tag{18}$$

Summing this upper bound across all H heads yields the final Influence Score $\psi_l(j)$, as used in the token filtering algorithm.

B.2 Theoretical Interpretation: Connection to Integrated Gradients

The derived influence score $\psi_l(j)$ aligns theoretically with feature attribution methods. Specifically, the importance of a token h_j to the output function F can be formalized using Integrated Gradients (IG):

$$IG_j(F) = (h_j - 0) \times \int_{\alpha=0}^{1} \frac{\partial F(\alpha h_j)}{\partial h_j} d\alpha. \quad (19)$$

In the context of the attention mechanism, assuming a single-step linear approximation ($\alpha = 1$) and the "Attention-as-Gradient" hypothesis, the gradient flow is proportional to the attention weight weighted by the value projection. Consequently, the perturbation term in Eq. 14 can be viewed as:

$$\|\Delta o_{l,h}(e)\| \approx \| \underbrace{\alpha_{l,h}(e \to j) \cdot v_{l,h,j}}_{\text{Attribution}} \|. \quad (20)$$

Therefore, our filtering strategy $\min_j \psi_l(j)$ is closely related to pruning the token with the minimal first-order contribution to the layer's output.

B.3 Global Stability Analysis via Neural ODEs

While the derivation above guarantees minimal *local* perturbation, we further demonstrate that this error does not propagate catastrophically. We view the residual connections in the MLLM backbone as an Euler discretization of a continuous Ordinary Differential Equation (ODE):

$$h_{l+1} = h_l + f(h_l) \iff \frac{dh(t)}{dt} = f(h(t), t) \quad (21)$$

Let $h(t)$ represent the trajectory of the full token sequence and $\tilde{h}(t)$ represent the trajectory after pruning a token at time t_l. The initial discrepancy introduced by pruning is bounded by our influence score: $\|\tilde{h}(t_l) - h(t_l)\| \leq \epsilon = \min_j \psi_l(j)$.

Assuming the network dynamics function f (Transformer block) is K-Lipschitz continuous (satisfied by bounded activation functions and LayerNorm), the divergence between the trajectories at the final time T (Layer L) is governed by **Gronwall's Inequality**:

$$\|\tilde{h}(T) - h(T)\| \leq \|\tilde{h}(t_l) - h(t_l)\| \cdot e^{K(T-t_l)}. \quad (22)$$

This inequality suggests that the error introduced by our skimming strategy remains bounded under the stated assumptions. By minimizing the initial perturbation $\|\tilde{h}(t_l) - h(t_l)\|$ via our Influence Score, the deviation in the final retrieval embedding can be kept under control, rather than growing chaotically.

C Implementation Details of Termination Classifier

In this section, we provide the technical specifications for the Termination Classifier $g(\cdot)$ utilized in Stage II.

C.1 Input Feature Construction

The classifier aggregates information from three distinct sources. We specifically design these features to capture inference uncertainty, input modality, and semantic context:

• **Confidence & Uncertainty Scalars ($\mathbf{x}_{stat} \in \mathbb{R}^{27}$):** We extract 27 scalar statistics from the intermediate layer's output. Crucially, to explicitly inform the classifier about ranking stability, we include:

(1) **Top-1 Similarity Score:** The cosine similarity of the highest-ranked candidate, serving as a proxy for absolute confidence.

(2) **Top-1/Top-2 Margin:** The difference between the similarity scores of the top-1 and top-2 candidates. A larger margin typically indicates a decisive prediction, while a narrow margin suggests potential ambiguity.

Other metrics include attention entropy, token density, and maximum attention scores. To rectify the highly skewed, long-tail distribution of these values (e.g., entropy varies significantly across layers), we apply a signed `Log1p` transformation followed by Layer Normalization:

$$\hat{\mathbf{x}}_{stat} = \text{LayerNorm}(\text{sign}(\mathbf{x}_{stat}) \cdot \ln(1 + |\mathbf{x}_{stat}|)). \quad (23)$$

• **Modality Embedding ($\mathbf{x}_{mod} \in \mathbb{R}^4$):** A learnable embedding layer maps the discrete modality index (representing text-only, image-only, or multimodal inputs) to a 4-dimensional continuous vector. This allows the classifier to adapt its decision boundary biases according to the specific input data type.

• **Semantic Projection ($\mathbf{x}_{sem} \in \mathbb{R}^{64}$):** To incorporate semantic context without incurring high computational costs, the high-dimensional backbone

embedding $\mathbf{e}_{l_{exit}}$ ($D = 3584$ for Qwen2.5-VL) is projected to a lower dimension via a non-linear head:

$$\mathbf{x}_{sem} = \text{ReLU}(\text{LayerNorm}(\mathbf{W}_p \mathbf{e}_{l_{exit}} + \mathbf{b}_p)), \tag{24}$$

where the projection dimension is set to 64.

C.2 Network Architecture

The processed features are concatenated into a single vector $\mathbf{h}_{in} \in \mathbb{R}^{95}$ and fed into the classifier. The architecture is a Multi-Layer Perceptron (MLP) defined as:

$$p_{exit} = \text{Sigmoid}(\text{MLP}_{\text{LN}}(\mathbf{h}_{in})). \tag{25}$$

Specifically, MLP_{LN} consists of two hidden layers with dimensions 128 and 64, respectively. We consistently employ Layer Normalization instead of Batch Normalization after each linear transformation, followed by ReLU activation and Dropout ($p = 0.2$). This design ensures that normalization statistics are computed independently for each instance, enhancing robustness against batch-size variations and distribution shifts during inference.

C.3 Training Objective

Since samples that are already sufficient at intermediate layers and those requiring deeper computation are typically imbalanced, we optimize the classifier using Focal Loss (Lin et al., 2017):

$$\mathcal{L}_{focal} = -\alpha(1 - p_t)^{\gamma} \log(p_t), \tag{26}$$

where p_t represents the estimated probability for the ground-truth class. This loss reduces the dominance of easy-to-classify samples, focusing training on boundary cases where the sufficiency decision is most ambiguous. We set $\alpha = 0.8$ and $\gamma = 3.0$ in all experiments.

D Hyperparameter Sensitivity: Selection Layer Analysis

D.1 Selection of the Selection Layer (l_s)

A key design choice in SAS is the position of the selection layer l_s, where Influence-Aware Token Filtering is applied. Choosing l_s requires balancing computational savings against representational maturity:

- **Too early ($l_s < 8$):** Multimodal representations may not yet be sufficiently mature. Pruning at this stage risks removing tokens that appear locally redundant but are still important for deeper cross-modal aggregation.
- **Too late ($l_s > 12$):** Although safer, delayed pruning preserves redundant tokens for too many layers and weakens the efficiency gains.

To identify an appropriate l_s, we conduct an ablation study on Qwen2.5-VL-3B trained for 2,000 steps on MMEB. We vary the selection layer over $l_s \in \{6, 8, 10, 11, 12, 13\}$ and measure retrieval performance (Hit@1) at both the intermediate layer (L_{12}) and the final layer (L_{36}).

Selection (l_s)	**Retrieval Performance (Hit@1)**	
	Inter. (L_{12})	**Final (L_{36})**
6	39.5%	66.3%
8	39.3%	66.4%
10	44.4%	**69.2%**
11	44.4%	68.1%
12	45.4%	68.7%
13	5.3%	68.2%

Table 6: **Impact of selection layer choice.** Retrieval performance at the intermediate and final layers under different l_s settings. $l_s = 10$ achieves the best balance.

Analysis. As shown in Table 6, applying pruning too early (layers 6 or 8) noticeably degrades final performance (around 66.3%–66.4% versus 69.2%), confirming that early representations are still too unstable for aggressive reduction.

Notably, $l_s = 10$ gives the best overall result, outperforming even later choices such as $l_s = 12$. We conjecture that selecting tokens at layer 10 leaves a useful buffer before the first depth sufficiency check at $l_d = 12$, allowing the remaining tokens to re-distribute attention and consolidate cross-modal semantics after pruning. By contrast, when filtering is applied too close to the first depth decision, the model has insufficient room to recover from the perturbation introduced by token removal. We therefore set $l_s = 10$ in all main experiments.

E Token Pruning Strategy Comparison

We compare SAS with two vision-token pruning baselines, FastV and DART, and a random text-pruning baseline. Table 7 reports retrieval Hit@1, modality-specific token reduction rates, and relative FLOPs for the evaluated configurations.

As shown in Table 7, FastV and DART achieve 60.6% and 59.0% Hit@1, respectively, with 75% visual-token reduction. Random text pruning achieves 58.3% with 50% textual-token reduction. SAS achieves 68.0% while reducing visual and textual tokens by 75% and 50%, respectively. These

Method	Avg. ↑	Vis. R. ↑	Text R. ↑	FLOPs ↓
DenseRet-3B	68.9	0%	0%	100%
Vision Pruning Baselines				
+ FastV	60.6	75%	0%	51.3%
+ DART	59.0	75%	0%	51.3%
Text Pruning Baseline				
+ Random	58.3	0%	50%	76.9%
Ours				
+ SAS	**68.0**	**75%**	**50%**	**36.6%**

Table 7: Comparison with vision and text token pruning strategies. Vis. R. and Text R. denote the reduction rates of visual and textual tokens, respectively.

l_d	Hit@1 at l_d	Hit@1 at L_{36}
8	47.7	66.8
10	50.9	69.4
11	50.5	68.2
12	50.6	**69.4**
13	18.8	67.7

Table 8: **Impact of decision layer choice.** Average Hit@1 at the selected decision layer l_d and at the final layer. We choose l_d=12 in the main experiments.

results compare the complete configurations and demonstrate that SAS preserves retrieval quality under joint visual and textual reduction.

F Exit Layer Ablation

Table 8 reports the average Hit@1 at the selected decision layer and at the final layer under different choices of l_d. Although $l_d = 10$ and $l_d = 12$ achieve comparable final-layer performance, we use $l_d = 12$ in the main experiments because it provides a more stable decision point after token selection and better practical early-exit behavior.

G Detailed Ablation on Supervision Strategies

Table 9: **Impact of Intermediate Supervision Strategies.** Comparison of Hit@1 accuracy at the intermediate (L_{12}) and final (L_{last}) layers.

Training Strategy	Exit (L_{12})	Final (L_{last})
Baseline (No Loss)	0.01	68.9
$+\mathcal{L}_{con}$ Only	49.4	69.0
$+\mathcal{L}_{distill}$ Only	**57.4**	69.2
Hybrid (Ours)	57.2	**69.5**

This section provides the detailed numerical comparison of different intermediate supervision strategies. Table 9 reports the retrieval accuracy (Hit@1) at both the intermediate exit layer (L_{12}) and the final layer (L_{last}). Further analysis reveals the limitations of single objectives. Applying solely $\mathcal{L}_{con}$ leads to under-fitting (49.4%) due to optimization difficulties at shallow layers, while solely $\mathcal{L}_{distill}$ over-regularizes the network, suppressing the final layer's potential (L_{last} drops to 69.2%). In contrast, our Hybrid Strategy resolves this dilemma, securing high utility for early exiting (57.2%) while boosting the final layer's peak performance to **69.5%**.

Metric	OVEN	CIRR	COCO_{i2t}	FIQ	Avg.
Forward to L_{12}	771	743	705	716	733.8
Influence Scoring	28.9	33.6	35.5	35.7	33.4
Dynamic Gather	8.4	12.0	4.0	4.6	7.3
Classifier Gating	7.9	6.4	8.5	9.0	6.9

Table 10: Per-batch control overhead at batch size 64 (all times in ms). The forward-to-L_{12} row is the standard computation up to the exit layer; the remaining rows are the additional costs of SAS. Influence scoring accounts for 3.7–5.0% of the forward time, and the total overhead (sum of the three components) is 5.9–7.0%.

H Additional Efficiency Analysis

H.1 Control Overhead of Skim and Skip

To verify that the speedup of SAS comes from real computation reduction rather than being offset by control overhead, we explicitly measure the wall-clock cost of the three auxiliary components: influence scoring (Skim), dynamic token gather (Skim), and classifier gating (Skip). Table 10 reports per-batch timing at batch size 64, measured up to the exit layer $l_d = 12$ on a single NVIDIA A100 GPU with synchronized CUDA timing.

These additional components account for approximately 6–7% of the runtime up to l_d at batch size 64, and are further amortized at larger batch sizes. For instance, the classifier gating overhead drops to approximately 0.1% at batch size 512. This confirms that the control overhead of SAS is modest and does not erode the FLOPs savings.

H.2 GPU Peak Memory

SAS reduces the number of tokens and layers processed per query, which also lowers memory consumption. Table 11 compares peak GPU memory between the dense baseline and SAS at batch size 64.

SAS consistently uses less memory than the dense baseline, with reductions of 8–19%. The

Dataset	Dense	SAS	Red.
CIRR	9.57	8.80	−8.0%
FIQ	10.21	8.93	−12.5%
COCO_{i2t}	9.58	8.82	−7.9%
OVEN	11.46	9.27	−19.1%

Table 11: GPU peak memory (GB) comparison at batch size 64. SAS consistently uses less memory than the dense baseline, as shorter sequences lead to smaller activations and KV caches.

Setting	Avg. Hit@1
Main paper (1 seed)	68.0
Setting A (5 seeds, classifier only)	68.49 ± 0.04
Setting B (3 seeds, 1,000 steps)	66.47 ± 0.36
Setting B (3 seeds, 2,000 steps)	67.87 ± 0.54

Table 12: Multi-seed robustness of SAS-3B. Setting A retrains only the classifier on a fixed backbone; Setting B retrains the full pipeline with a reduced step budget to save compute.

savings are larger for tasks with more input tokens (e.g., OVEN), as token skimming more substantially reduces activation sizes and KV caches. The only additional memory in SAS is a lightweight MLP classifier (<0.1M parameters) and a one-time spectral-norm cache, both of which are negligible compared to the backbone.

I Multi-Seed Robustness Analysis

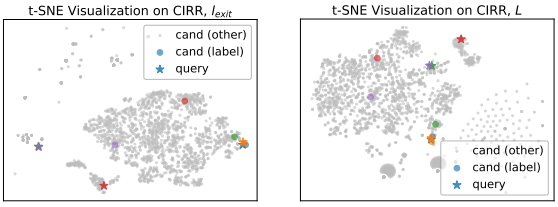
To assess the robustness of our main results to random initialization, we re-ran the SAS-3B experiment (12-task Hit@1) under two settings:

- **Setting A**: keep the trained backbone fixed and retrain the early-exit classifier with 5 random seeds.
- **Setting B**: retrain the full pipeline (backbone + classifier) with 3 random seeds, evaluated at 1,000 and 2,000 training steps.

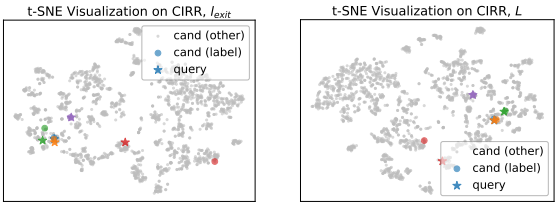
These results lead to three observations. (1) Stage II is nearly seed-insensitive: retraining only the classifier (Setting A) yields 68.49 ± 0.04, with most per-task standard deviations essentially zero. (2) The conclusion remains stable under full-pipeline retraining (Setting B), with averages of 66.47 ± 0.36 and 67.87 ± 0.54 at 1,000 and 2,000 steps. (3) Larger per-task variance appears only in under-trained settings on harder tasks and shrinks with more training (e.g., CIRR ±3.1 → ±0.8, OVEN ±4.4 → ±2.5), indicating under-training noise rather than instability of the method itself.

J More Visualization Results

To further verify that Stage I training activates the retrieval capability of the exit layer, we visualize query and candidate embeddings via t-SNE, before and after training at both the exit layer l_d and the final layer L. Before training, queries and their ground-truth candidates are widely separated at l_d; after Stage I training, they converge into semantically cohesive clusters, confirming that the intermediate supervision endows layer l_d with discriminative retrieval power. This pattern holds consistently across all eight datasets.

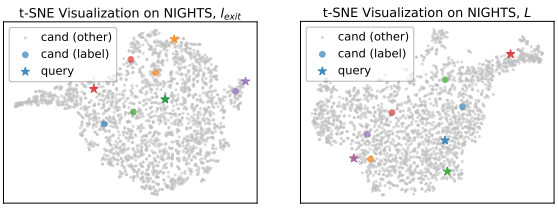


(a) Before training

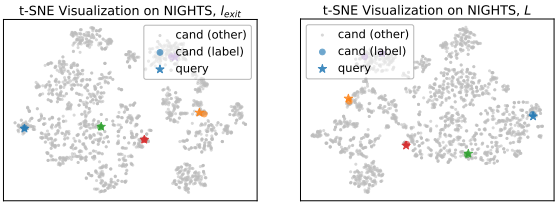


(b) After training

Figure 6: t-SNE visualizations on CIRR

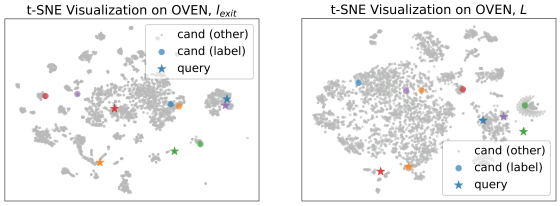


(a) Before training

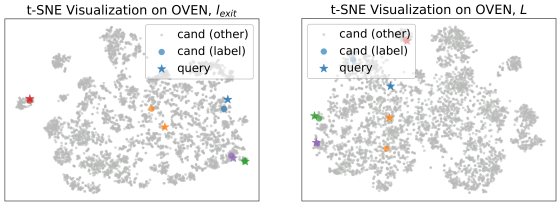


(b) After training

Figure 7: t-SNE visualizations on NIGHTS

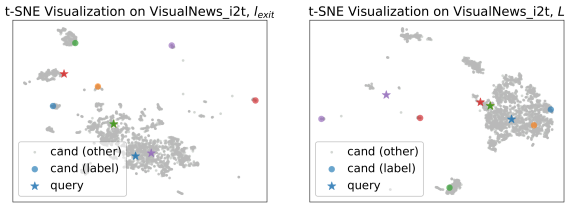


(a) Before training

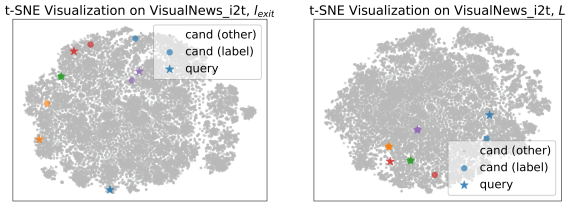


(b) After training

Figure 8: t-SNE visualizations on OVEN

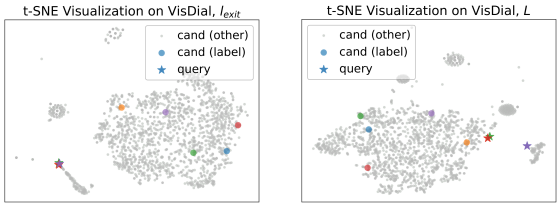


(a) Before training

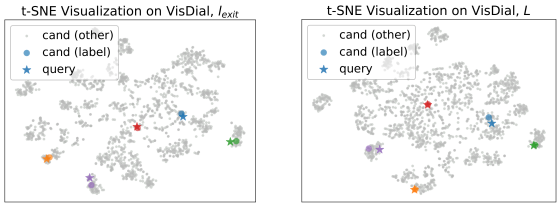


(b) After training

Figure 9: t-SNE visualizations on VisualNews_i2t

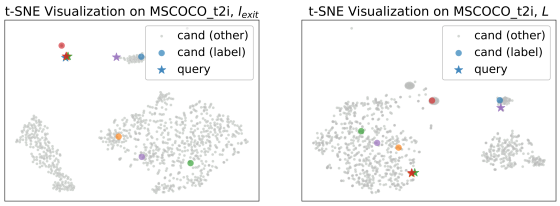


(a) Before training

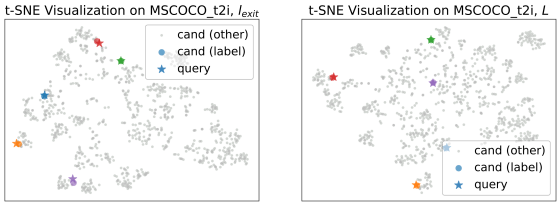


(b) After training

Figure 10: t-SNE visualizations on VisDial

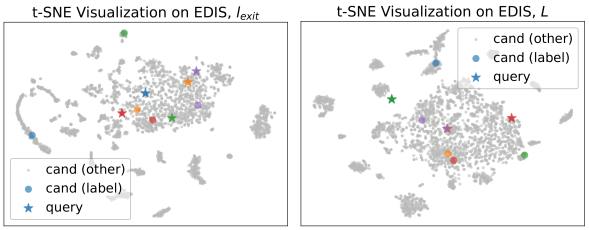


(a) Before training

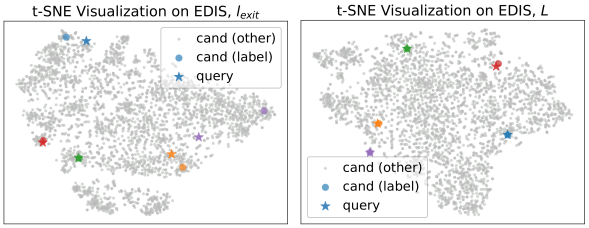


(b) After training

Figure 11: t-SNE visualizations on MSCOCO_t2i

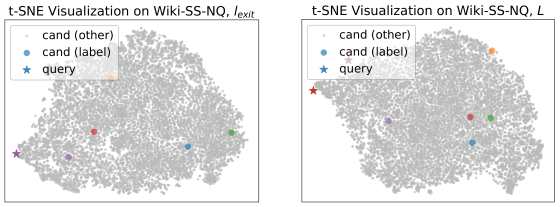


(a) Before training

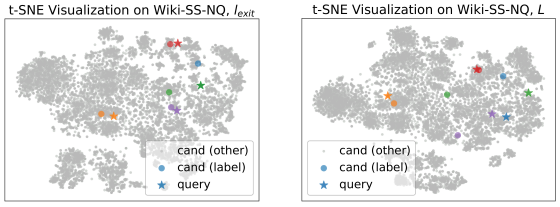


(b) After training

Figure 12: t-SNE visualizations on EDIS



(a) Before training



(b) After training

Figure 13: t-SNE visualizations on Wiki-SS-NQ