Adaptive Internal Readout for Native Multimodal Models

Yunpeng Zhou
Nanjing University of Information Science and Technology
University of Reading
Nanjing, Jiangsu, China
kc804139@student.reading.ac.uk

Abstract

Native multimodal models form representations across visual, multimodal, and language streams, yet predictions typically rely on the final state. We study whether intermediate states can support task answers that remain less accessible through a final-state readout. We measure this availability–accessibility gap (AAG) using a small-budget internal readout reference and a learned final-only parity head under the same supervised answer space and loss. We introduce adaptive internal readout (AIR), which selects a few internal sites for each input while keeping the backbone frozen. Across six multimodal benchmarks and three backbones, AIR outperforms final-only, static-fusion, and dense-fusion readouts under controlled budgets. A pooled-final control using the same context-token pooling as internal multimodal sites improves the final-only baseline, explaining part of the original difference. AIR retains an advantage after this control, although the measured gap narrows on the stronger backbones studied. Routing diagnostics and functional site tests support the contribution of the selected representations. These findings characterize AAG as a protocol-dependent diagnostic in supervised structured-output evaluation.

1 Introduction

Native multimodal models process images, documents, diagrams, and long interleaved contexts within a single model stack. Recent models increasingly integrate visual and text representations during training and inference [Bai et al., 2025a,b, Farabet and Lacombe, 2026, Shukor et al., 2025]. This changes how we should interpret errors. When a model gives a wrong answer, did it fail because the relevant information was never represented, or because that information was not effectively accessed by the final readout?

Current evaluation protocols rarely separate these cases. Standard decoding relies on the final-layer representation. Probing, visual feature fusion, and early-exit methods show that intermediate representations can be useful [Alain and Bengio, 2017, Belinkov, 2022, Bolya et al., 2025, Yao et al., 2024, Cao et al., 2024, Li et al., 2026, Bajpai and Hanawal, 2025]. These approaches typically study intermediate states as useful features, earlier prediction points, or auxiliary evidence, leaving open whether task information is already available internally but less accessible through the final readout. This question is particularly relevant to native multimodal models, where evidence may be distributed across visual, multimodal, and language-side states.

We study this phenomenon as *readout mismatch* and formalize it through the *availability–accessibility gap* (AAG). We report a default gap relative to standard decoding, while the primary parity gap compares a small-budget internal readout reference with a learned final-only head under the same supervised answer space and loss. This comparison controls for the benefit of attaching a supervised

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

head, but the measured gap also depends on how site representations are constructed. We therefore include a pooled-final control that applies the same context-token pooling rule as internal multimodal sites. AAG is interpreted as a diagnostic under this specified readout protocol rather than an intrinsic measure of model knowledge.

To investigate this gap, we define a full-stack site space spanning visual, multimodal, and language states. We compare internal access against final-only readouts, including the pooled-final control, single-site readouts, static and dense fusion, visual-only readout, early-exit baselines, capacity-matched final-only controls, and matched random sites. We introduce *adaptive internal readout* (AIR), a lightweight policy that selects a small set of internal sites for each input and combines their projected representations through a shared readout head. The backbone remains frozen, and trainable budgets are matched against competing fusion baselines.

Our experiments use Gemma 4 E4B-it as the main analysis model, with additional evaluations on Qwen3-VL-8B-Instruct and Qwen3.5-27B. All learned readouts share the same answer space and loss, while default generative scores are reported separately. On Gemma 4, AIR reaches 78.9, compared with 74.4 for default decoding, 75.2 for the learned final-only head, and 76.0 for the pooled-final control. Pooling explains part of the original difference, while AIR retains a 2.9-point advantage. On Qwen3-VL and Qwen3.5, the corresponding advantages over pooled-final readout decrease to 0.6 and 0.5 points. The measured AAG also narrows on these stronger backbones, indicating that the magnitude of readout mismatch depends on the models examined.

The analysis further characterizes where and how internal information is accessed. Task-site heatmaps reveal different readout topologies: local perceptual tasks favor visual and middle sites, while structural and broad reasoning tasks rely more on multimodal and language-side states. Full-stack comparisons show that visual-only access is insufficient for several tasks requiring cross-modal or semantic integration. Routing diagnostics indicate that site selection becomes more concentrated during training while retaining input-dependent variation. Functional ablations show that removing selected sites hurts more than replacing them with neighboring or matched random sites, and rerouting only partially recovers the loss.

These results support the existence of readout mismatch under controlled supervised evaluation and show that sparse adaptive access can recover part of the measured gap without modifying the backbone. The gains vary substantially across models, and the current evidence concerns task-specific structured-output readouts rather than general autoregressive generation. AAG provides a protocol-dependent way to examine internal information accessibility, while AIR offers a lightweight mechanism for recovering information through selected internal states.

2 Related Work

Intermediate representations and adaptive layer use. Deep networks often expose useful information before the final layer. This has been studied through probing, logit lenses, mechanistic analysis, and visual representation analysis [Alain and Bengio, 2017, Belinkov, 2022, nostalgebraist, 2020, Geva et al., 2021, Olsson et al., 2022, Darcet et al., 2024]. Recent layer-wise analyses further show that intermediate representations can outperform final-layer states on downstream representation tasks [Skean et al., 2025]. Dynamic depth, mixture-of-depths, layer selection, and adaptive skipping methods also use internal layers adaptively, but primarily to allocate computation or reduce inference cost [Raposo et al., 2024, Glavas et al., 2024, Jain et al., 2025, He et al., 2025, Luo et al., 2025]. Our goal is different: we use adaptive selection to measure and recover accessibility gaps under a parity readout protocol. We include adaptive scalar mix as a direct dense layer-attention baseline, but AIR selects sparse input-dependent sites across visual, multimodal, and language-side streams rather than assigning dense weights over a single layer stack.

Multimodal fusion and native full-stack access. Recent multimodal work shows that intermediate visual features are often useful. Perception Encoder finds that strong image and video embeddings can reside in intermediate visual layers rather than the output layer [Bolya et al., 2025]. DenseConnector, MMFuser, and instruction-guided fusion use multi-layer visual features to improve LVLMs, with evidence that low-level, middle-level, and high-level visual features support different task types [Yao et al., 2024, Cao et al., 2024, Li et al., 2026]. Qwen3-VL also adopts DeepStack to inject multi-level ViT features into the language model [Bai et al., 2025b]. These works motivate reading beyond the final layer, but they primarily focus on visual-side aggregation. Native multimodal models instead

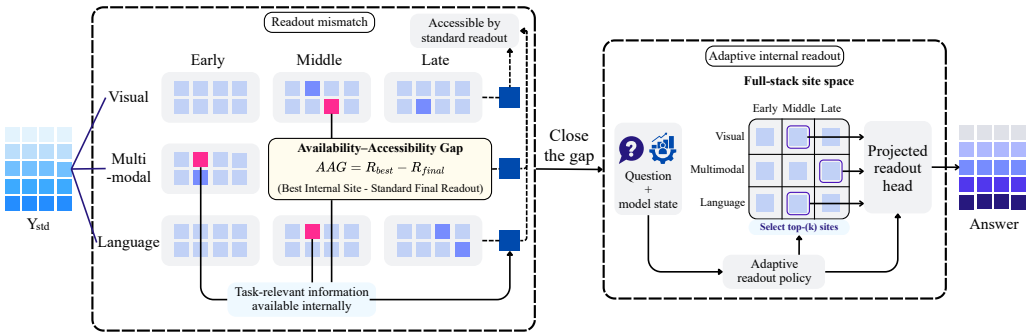


Figure 1: **Readout mismatch in native multimodal models. Top-left inset:** Standard decoding uses the final-layer representation to produce Y_{std}. **Left:** Task-relevant information may be accessible through internal visual, multimodal, or language sites even when the final readout does not recover it effectively. The availability–accessibility gap (AAG) measures this difference under a specified readout protocol. Its primary parity form compares a small-budget internal reference with a learned final-only head under the same answer space and loss. **Right:** Adaptive internal readout (AIR) selects a small set of internal sites for each input and combines their representations through a shared readout head, producing Y_{adapt}. The frozen backbone is unchanged; adaptation occurs in the selection and readout modules.

mix image and text tokens more deeply through unified token spaces, early fusion, long interleaved contexts, and joint multimodal training [Radford et al., 2021, Alayrac et al., 2022, Li et al., 2023, Dai et al., 2023, Liu et al., 2023, 2024, Bai et al., 2025a,b, Shukor et al., 2025, Farabet and Lacombe, 2026]. In this setting, relevant evidence can appear in visual, multimodal, or language-side states. We therefore define a full-stack site space and measure whether final-layer access leaves internally available information unused.

Interpretability, sparse features, and functional internal states. Probing and sparse autoencoders provide tools for identifying information in activations [Bricken et al., 2023, Huben et al., 2024, Templeton et al., 2024, Shu et al., 2025]. Recent benchmarks caution that unsupervised interpretability scores do not always predict practical usefulness [Karvonen et al., 2025]. In vision and vision-language models, SAEs have been used to discover interpretable visual concepts and test controlled interventions [Stevens et al., 2025, Pach et al., 2025]. These works distinguish representation from behavior. We build on the same distinction without requiring human-named features: we test whether an internal site can support the task answer, whether an adaptive readout can recover it under matched budgets, and whether selected sites are functionally important for the learned readout pathway.

Internal reasoning and deployable hidden information. Recent work increasingly treats reasoning as a process over hidden states rather than only emitted text. Latent reasoning methods introduce hidden thinking states, while Chain-of-Visual-Thought trains VLMs to reason through compact continuous visual tokens [Zelikman et al., 2024, Amos et al., 2026, Qin et al., 2025]. Work on in-context representation learning and KV binding shows a related failure mode: a model may encode a useful pattern internally without reliably deploying it through its output procedure [Lepori et al., 2026, Liu et al., 2026]. Early-exit and efficient VLM methods also access intermediate states, but mainly optimize latency or token cost rather than separating information availability from output accessibility [Schwartz et al., 2020, Xin et al., 2020, Chen et al., 2024, Yang et al., 2024, Vasu et al., 2025, Bajpai and Hanawal, 2025]. Our work formalizes this separation as the availability–accessibility gap and tests it against visual-only fusion, static fusion, dense all-site fusion, adaptive scalar mix, early exiting, and random matched-site controls.

3 Method

We study readout mismatch in a frozen native multimodal model. An internal state may support a task answer without making that information equally accessible through the final readout. We place all

learned readouts within a common supervised readout family, allowing final-state, internal-site, and adaptive readouts to be compared under the same answer space and controlled parameter budgets.

3.1 Full-stack readout sites

Let f be a frozen native multimodal model and let $x = (v, q)$ denote a visual input and a question. During a forward pass, the model produces hidden states at multiple locations. We collect a finite candidate set

$$\mathcal{S} = \mathcal{S}_v \cup \mathcal{S}_m \cup \mathcal{S}_l,$$

where $\mathcal{S}_v$ contains visual sites, $\mathcal{S}_m$ contains multimodal sites, and $\mathcal{S}_l$ contains language-side sites. Each site $s \in \mathcal{S}$ returns a representation $h_s(x) \in \mathbb{R}^{d_s}$. Visual states use masked mean pooling over visual tokens, while multimodal states are pooled over eligible image-text context tokens, excluding answer-side tokens. Language-side sites use the hidden state at the answer position induced by the fixed prompt, without teacher-forcing ground-truth answer tokens. Exact site counts and tensor shapes are reported in Appendix A.

To compare sites with different hidden sizes, each representation is mapped into a shared readout dimension d_r:

$$z_s(x) = P_s h_s(x), \qquad P_s \in \mathbb{R}^{d_r \times d_s}.$$

For final-state comparisons, we separately use the final answer-side representation $z_{\text{final}}(x) = P_{\text{final}} h_{\text{final}}(x)$. All backbone parameters remain frozen. Only site projections, readout heads, fusion modules, and site-selection modules are trained.

For analysis, sites are grouped by stream and depth. The streams are visual, multimodal, and language, and depth is divided into early, middle, and late bins. These groups define the grids used in the figures; learned readouts operate on the original candidate sites rather than grouped averages.

3.2 A common readout family

Default generative decoding and a trained discriminative head use different output procedures. We therefore distinguish the model's default output from learned parity readouts.

For a dataset with answer set $\mathcal{Y}$, every learned readout uses the same scoring form

$$\ell_R(x, y) = w_y^\top z_R(x) + b_y, \qquad y \in \mathcal{Y},$$

where $z_R(x)$ is the representation supplied by method R. For multiple-choice tasks, $\mathcal{Y}$ contains canonical option labels, while the option texts are included in the input prompt processed by the frozen backbone. For short-answer tasks, $\mathcal{Y}$ is the normalized training-answer vocabulary. Default generative scores are reported separately from learned-readout results.

The learned final-only head is the primary parity control. It reads $z_{\text{final}}(x)$ under the same answer space and scoring rule as internal-site heads, controlling for the gain obtained by attaching a supervised readout to the final state.

To examine the effect of representation construction, we introduce a pooled-final parity control. It applies the same masked context-token pooling rule used for internal multimodal sites to the final-layer token representations, excluding answer-side tokens. The resulting representation is projected into the same readout dimension and evaluated with the same head architecture, loss, and training protocol. This control isolates the contribution of token pooling to the observed internal-versus-final difference. The original answer-side final-only head remains the reference for the formal AAG definition; improvements over pooled-final readout are reported separately.

All learned readouts are trained with cross entropy,

$$\mathcal{L}(R) = -\log \frac{\exp \ell_R(x, y)}{\sum_{y' \in \mathcal{Y}} \exp \ell_R(x, y')}.$$

We use the same training split, answer normalization, loss, and evaluation protocol across learned readouts. For short-answer tasks, the vocabulary is constructed only from normalized training answers. Held-out targets outside this vocabulary are counted as incorrect for all learned heads.

3.3 Availability–accessibility gap

We define availability within the common supervised readout family. For an internal site s, let r_s denote its validation score after training a lightweight head on $z_s(x)$. The best single-site score is

$$r_{\text{single}} = \max_{s \in \mathcal{S}} r_s.$$

This quantity is used to analyze which streams and depths support task predictions.

For gap-closure analysis, we use a small-budget internal oracle envelope. Let $\mathcal{R}_{\text{int}}^{(K)}$ denote the class of internal readouts using at most K sites under the same answer space and loss. We define

$$r_{\text{env}}^{(K)} = \max_{R \in \mathcal{R}_{\text{int}}^{(K)}} r_R.$$

The main analysis uses $K = 2$ as a compact diagnostic reference. Oracle site selection uses validation labels, whereas AIR does not receive oracle site labels at test time. We also examine the envelope across $K \in \{1, 2, 4, 8\}$ to assess its dependence on the site budget. The chosen envelope is a reference for small-site readouts and need not upper-bound methods operating under a different selection budget.

Let $r_{\text{final}}^{\text{def}}$ denote the default final-layer decoding score and $r_{\text{final}}^{\text{head}}$ the learned final-only head score. We define

$$\text{AAG}_{\text{def}}^{(K)} = r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{def}},$$

and

$$\text{AAG}_{\text{head}}^{(K)} = r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{head}}.$$

The first measures the gap relative to default decoding. The second is the primary parity metric, controlling for the use of a learned final-state head.

For any learned readout R, absolute gap closure is

$$\text{GapClosed}(R) = r_R - r_{\text{final}}^{\text{head}},$$

and normalized closure is

$$\text{ClosureRatio}^{(K)}(R) = \frac{r_R - r_{\text{final}}^{\text{head}}}{r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{head}}},$$

when the denominator is positive. The learned final-only head defines the zero point, and the internal envelope provides the diagnostic reference. These quantities depend on the candidate sites, representation construction, and readout protocol. The pooled-final control measures the contribution of pooling without changing the original gap definitions.

3.4 Adaptive internal readout

Adaptive internal readout selects a small subset of sites for each input. A site scorer assigns

$$a_s(x) = g_\theta(u(x), e_s),$$

where $u(x)$ summarizes the current input and model state, and e_s is a learned site embedding. The scorer is implemented as a two-layer MLP:

$$\tilde{u}(x) = \text{LN}(W_u u(x)), \qquad a_s(x) = w_2^\top \sigma(W_1[\tilde{u}(x); e_s] + b_1) + b_2,$$

where σ is GELU. The scorer width is reported in Appendix A.

In the main configuration, $u(x)$ is obtained from the final answer-side state and serves only as the selector query. The adaptive readout representation is constructed from selected internal sites rather than directly from this final-state query. Appendix D evaluates question-only, image-only, and shallow multimodal selector summaries, which remain functional with somewhat lower accuracy.

The selected site set is

$$\mathcal{T}_k(x) = \arg\operatorname*{topk}_{s \in \mathcal{S}} a_s(x),$$

with ties broken by a fixed site order. Scores are normalized over the selected sites:

$$\alpha_s(x) = \frac{\exp(a_s(x))}{\sum_{j \in \mathcal{T}_k(x)} \exp(a_j(x))}, \qquad z_{\text{adapt}}(x) = \sum_{s \in \mathcal{T}_k(x)} \alpha_s(x) z_s(x).$$

The shared readout head maps $z_{\text{adapt}}(x)$ to answer scores.

We use hard top-k selection during training and inference, without Gumbel-top-k, temperature relaxation, or straight-through estimation. Gradients flow through the selected site projections and their scorer values, while the backbone remains frozen. No explicit exploration noise is added. Input-dependent selector summaries and randomly initialized site embeddings provide variation in early routing, which we examine using training-time entropy, site-selection frequencies, and cross-seed stability diagnostics in Appendix C.

Although AIR aggregates only k selected site representations, the main selector query is computed from the final answer-side state and therefore requires a full backbone forward pass. AIR reduces the additional readout cost relative to dense internal fusion; it does not reduce computation by skipping backbone layers.

3.5 Matched baselines

We compare AIR with controls that separate the effects of supervision, representation construction, readout capacity, site access, and adaptive selection. Standard final readout uses the model's default output. The learned final-only head provides the primary supervised parity reference, while pooled-final readout uses the same context-token pooling rule as internal multimodal sites.

Two final-state capacity controls use only $z_{\text{final}}(x)$. The capacity-matched final-only MLP increases head capacity, while the final-only routed head constructs learned pseudo-sites from the same final state and applies a routing mechanism comparable to AIR. These controls test whether additional capacity or routing alone can account for improvements obtained through internal-site access. The best single-site, best visual-site, and best multimodal-site readouts provide separate internal availability references.

Static learned fusion combines candidate sites using learned weights. Dense full-stack fusion accesses all candidate sites, while adaptive scalar mix assigns input-dependent dense weights across the same site space. Instruction-guided visual fusion restricts access to visual sites and uses the question representation for aggregation. Early-exit readout uses an intermediate language-side state.

Random matched top-k readout selects the same number of sites as AIR, matching stream and depth where possible. To isolate input-dependent routing, frequency-fixed top-4 readout uses the four sites most frequently selected by a trained AIR selector on the training split of each benchmark. The subset remains fixed across inputs, and the readout is retrained with the same selected-site budget and head architecture. Site selection does not use validation or test labels. The oracle top-2 readout uses validation labels to choose a small internal-site reference and is not treated as a general upper bound on AIR.

All learned baselines follow the same answer-space, loss, and evaluation protocol. Trainable budgets for the main full-stack comparison methods are approximately matched by adjusting scorer or fusion widths. Appendix A reports candidate sites, projection dimensions, trainable parameters, FLOPs, memory overhead, and latency.

3.6 Functional site tests

Our ablations evaluate whether the selected representations are functionally used by the learned readout pathway. They do not establish that these sites are necessary for the backbone's original computation. After AIR selects its top-k sites, we compare counterfactual readout conditions with the same site budget.

For selected-site removal, the selected representations are replaced with zeros before the readout head. Neighboring, same-depth random, and global random controls replace the selected sites with alternative representations and apply the same readout head. In the rerouting condition, the original sites are excluded from the candidate set before the selector chooses a new subset.

We report score changes relative to the original AIR prediction. Larger drops under selected-site removal, together with comparisons against matched replacements, characterize the dependence of the learned readout on particular internal states. Rerouting measures how much performance can be recovered when the policy must use alternative sites.

4 Experimental Setup

Models, tasks, and metrics. We use Gemma 4 E4B-it as the main analysis model because it permits full internal sweeps while providing a rich multimodal representation space. We further evaluate Qwen3-VL-8B-Instruct and Qwen3.5-27B to examine how readout mismatch varies across stronger backbones. All backbones remain frozen, with training restricted to readout projections, fusion modules, selectors, and output heads. We evaluate six benchmarks: DocVQA for document understanding, ChartQA for chart reasoning, AI2D for diagram understanding, TallyQA for counting, NLVR2 for compositional visual reasoning, and MMMU-Pro for broad multimodal reasoning. We use accuracy or the official dataset metric and report the average across six tasks. Readout heads are trained separately on each benchmark's training split and evaluated on its held-out split. Short-answer readouts use normalized training-answer vocabularies, while default generative decoding is evaluated separately.

Readout sites and baselines. Candidate sites span visual, multimodal, and language-side states, grouped into early, middle, and late depth bins for analysis. AIR selects $k = 4$ original sites per input rather than operating on grouped representations. We compare standard decoding, learned final-only and pooled-final readouts, capacity-matched final-state controls, single-site readouts, static and dense fusion, adaptive scalar mix, instruction-guided visual fusion, early-exit readout, and random matched-site selection. The pooled-final control matches the context-token pooling used by internal multimodal sites while retaining the same supervised readout protocol. A frequency-fixed top-4 control preserves the most frequently selected training sites without input-dependent routing. Oracle top-2 provides the primary small-budget diagnostic reference, with additional envelope evaluations at $K \in \{1, 2, 4, 8\}$.

Analysis protocol. We examine readout mismatch through task-by-site topology, AAG and gap closure, and functional tests of selected representations. The default-decoding and learned-head parity comparisons are reported separately, with pooled-final results used to assess the effect of representation construction. We evaluate robustness through paired readout-training seeds, selector entropy trajectories, and fixed-versus-adaptive routing comparisons. Additional analyses examine cross-task selector transfer and candidate reranking using outputs from the native decoder. Detailed site configurations, evaluation protocols, statistical procedures, and runtime measurements are provided in the appendices.

5 Results

5.1 Adaptive internal readout improves over matched baselines

Table 1 reports results on Gemma 4 E4B-it. Standard final-layer decoding reaches 74.4 across the six benchmarks, while the learned final-only parity head reaches 75.2. AIR improves the average score to 78.9. The best single visual and multimodal sites reach 75.8 and 76.3, respectively, showing that individual internal representations can support stronger predictions than the original final-only head. The capacity-matched final-only MLP and routed head reach 75.8 and 76.0, indicating that additional final-state readout capacity does not account for the full AIR gain.

AIR exceeds static learned fusion by 1.5 points, adaptive scalar mix by 1.2 points, and dense full-stack fusion by 1.0 point, while accessing only four selected sites rather than the entire candidate set. The advantage over dense fusion persists across three readout-training seeds, with robustness statistics reported in Appendix C. The oracle top-2 readout reaches 79.8, providing a small-budget reference close to AIR's 78.9. Since AIR uses four sites, this reference is a diagnostic envelope rather than a strict upper bound on adaptive readout.

Table 1: Main results on Gemma 4 E4B-it across document understanding, chart reasoning, diagram understanding, counting, compositional visual reasoning, and broad multimodal reasoning benchmarks.

Method	DocVQA	ChartQA	AI2D	TallyQA	NLVR2	MMMU-Pro	Avg.
Standard final-layer readout	83.9	72.4	78.6	87.9	70.8	52.6	74.4
Final-only learned head	84.6	73.1	79.4	88.7	71.6	53.7	75.2
Capacity-matched final-only MLP	85.1	74.1	80.0	89.3	72.2	54.0	75.8
Final-only routed head	85.2	74.5	79.9	89.2	72.8	54.2	76.0
Best single visual site	85.4	74.6	80.9	89.8	71.4	52.8	75.8
Best single multimodal site	84.7	75.8	80.1	89.1	73.9	54.1	76.3
Static learned fusion	85.9	77.1	81.4	90.4	74.6	55.0	77.4
Instruction-guided visual fusion baseline	85.6	76.4	80.7	89.6	72.7	53.7	76.5
Early-exit readout baseline	84.8	73.5	79.2	88.5	71.9	53.4	75.2
Adaptive scalar mix	86.1	77.5	81.8	90.7	74.9	55.1	77.7
Dense full-stack fusion	86.3	77.8	82.0	90.9	75.2	55.3	77.9
Random matched top-k sites	85.2	75.6	80.4	89.7	73.3	54.2	76.4
Adaptive internal readout (ours)	**87.1**	**79.0**	**83.1**	**91.8**	**76.1**	**56.2**	**78.9**
Oracle top-2 site readout	87.9	80.2	84.0	92.6	77.0	57.0	79.8

Table 2: Cross-backbone gap closure under the parity readout protocol. The internal reference is the oracle top-2 envelope $r^{(2)}_{\text{env}}$, used only as a diagnostic upper reference for measuring AIR's gap closure.

Model	Std. final	Final head	Oracle top-2	AIR	$\text{AAG}^{(2)}_{\text{head}}$	Closed	Closure_2
Gemma 4 E4B-it	74.4	75.2	79.8	78.9	4.6	3.7	80.4%
Qwen3-VL-8B-Instruct	83.1	83.6	85.0	84.5	1.4	0.9	64.3%
Qwen3.5-27B	87.3	87.7	89.0	88.5	1.3	0.8	61.5%

$\text{AAG}^{(2)}_{\text{head}}$ is Oracle top-2 minus Final head. Closed is AIR minus Final head. Closure_2 is Closed divided by $\text{AAG}^{(2)}_{\text{head}}$.

5.2 The effect replicates across stronger multimodal models

Table 2 reports cross-backbone gap closure under the original parity protocol. Relative to default decoding, AIR improves by 4.5 points on Gemma 4 E4B-it, 1.4 on Qwen3-VL-8B-Instruct, and 1.2 on Qwen3.5-27B. Relative to the learned final-only head, the gains are 3.7, 0.9, and 0.8 points. The corresponding parity AAG decreases from 4.6 to 1.4 and 1.3 points. AIR closes 80.4%, 64.3%, and 61.5% of these respective reference gaps. Both the measured gap and the absolute improvement decrease on the stronger backbones studied here.

Table 3 examines the contribution of representation construction. Applying the internal multimodal context-token pooling rule at the final layer improves the final-only head by 0.8 points on Gemma and 0.3 points on both larger backbones. AIR remains 2.9, 0.6, and 0.5 points above the pooled-final control, respectively. Pooling therefore explains part of the original difference, while the residual advantage is substantially smaller on Qwen3-VL and Qwen3.5. The paired-seed comparison assesses this residual using AIR-minus-pooled-final differences computed within matched training runs. The formal AAG values in Table 2 retain the original learned final-only reference.

We also vary the oracle-envelope budget over $K \in \{1, 2, 4, 8\}$. Increasing K from 2 to 8 raises the envelope by 0.3, 0.3, and 0.2 points across the three backbones. The measured AAG depends on the chosen budget, although the additional headroom beyond two sites is small in these experiments. The full sensitivity results are reported in Appendix C.

5.3 Internal topology and full-stack access

Figure 2 examines where task information is accessible within Gemma 4 E4B-it. The heatmap shows distinct task-dependent patterns: DocVQA, AI2D, and TallyQA favor visual or middle multimodal sites, while ChartQA, NLVR2, MMMU-Pro, and the controlled stress tasks draw more strongly on multimodal and language-side representations. The default-readout comparison in Figure 2(b)

Table 3: Effect of final-layer context-token pooling across backbones. Pooled-final uses the same context-token pooling rule as internal multimodal sites, excluding answer-side tokens. All learned readouts share the same answer space and loss. The final column reports paired 95% confidence intervals for AIR minus pooled-final across matched training seeds.

Backbone	Final head	Pooled final	AIR	AIR − pooled	95% paired CI
Gemma 4 E4B-it	75.2	76.0	78.9	**+2.9**	[+2.73, +3.07]
Qwen3-VL-8B-Instruct	83.6	83.9	84.5	**+0.6**	[+0.45, +0.75]
Qwen3.5-27B	87.7	88.0	88.5	**+0.5**	[+0.35, +0.65]

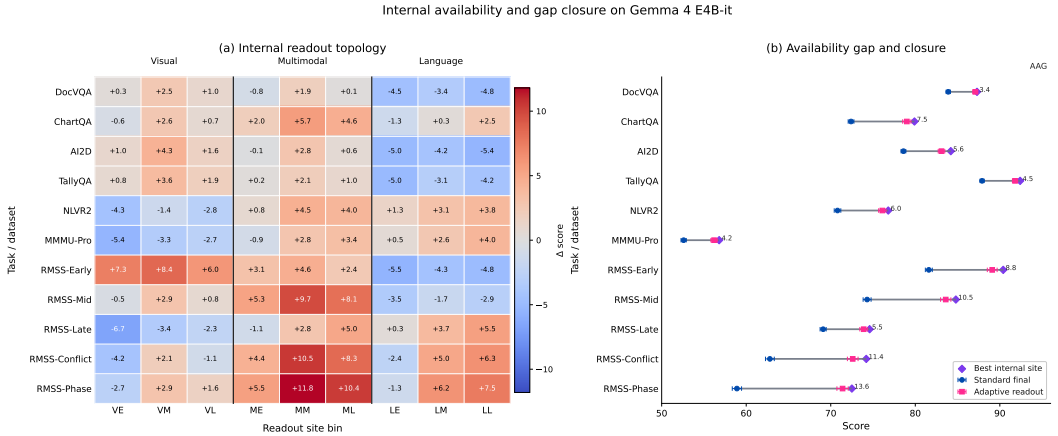


Figure 2: **Internal availability and gap closure on Gemma 4 E4B-it.** (a) Task-by-site score changes under forced internal readout relative to standard final-layer decoding. Positive values indicate stronger support than the default output. (b) Default-readout accessibility comparison using the best internal-site reference. Each interval spans standard final decoding and the internal reference, with the AIR result shown separately. The primary learned-head parity AAG is reported in Table 2. RMSS rows are controlled readout stress tests probing early, middle, late, conflicting, and phase-shifted evidence; they are excluded from the main average.

illustrates how internal access can recover information that is less accessible through standard decoding.

Figure 3(a) compares readout spaces across task families. Visual-only access is competitive on local perceptual tasks but weaker on structural and broad multimodal reasoning. Adding multimodal and language-side sites improves these harder task families, with full-stack AIR providing the strongest overall results.

We further examine whether hard top-k selection retains input dependence. Selector entropy decreases during training, reaching 1.07, 0.99, and 0.92 on the three backbones, while multiple sites remain active at convergence. A frequency-fixed top-4 control, using the most frequently selected training sites, reaches 78.0, 84.3, and 88.4, compared with AIR at 78.9, 84.5, and 88.5. Global site preference explains much of the performance on the stronger backbones, whereas input-dependent routing contributes a larger margin on Gemma. Training trajectories, routing diversity, and selector-transfer results are detailed in Appendices C and D.

5.4 Selected sites are functionally important

Figure 3(b) evaluates the functional role of selected representations in the learned readout pathway. Removing selected sites causes larger score drops than replacing them with neighboring, same-depth random, or global random sites. Rerouting recovers part of the lost performance but does not restore the original result. These controls indicate that AIR relies on particular internal representations rather than arbitrary site selections. This evidence concerns the learned readout pathway and does not establish that the same sites are necessary for the backbone's original computation.

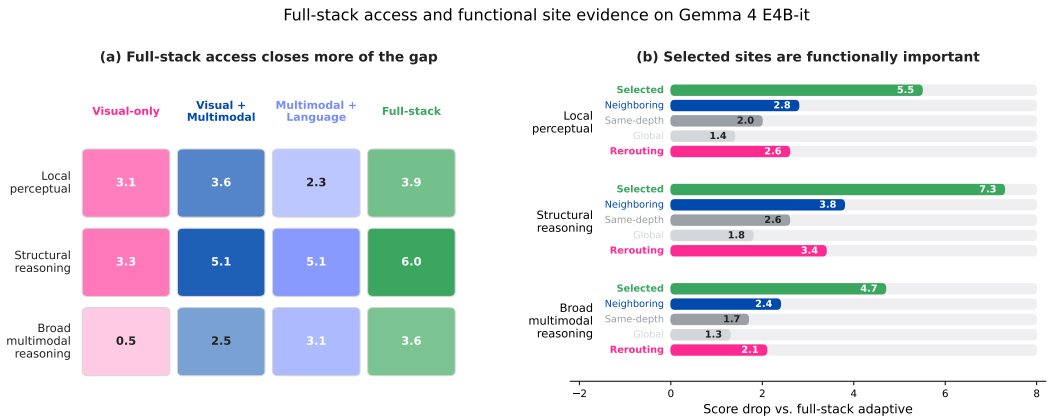


Figure 3: **Full-stack access and functional site evidence.** (a) Full-stack readout gives the largest gains over standard final-layer readout, especially on structural and broad reasoning tasks. (b) Removing selected sites hurts more than replacing them with neighboring or matched random control sites; rerouting only partially recovers the loss.

As a limited test beyond direct structured-output prediction, we also evaluate candidate reranking using outputs from the native decoder. On DocVQA, AIR reranking improves over final-state reranking from 85.2 to 85.9; on ChartQA, the corresponding scores increase from 74.8 to 76.2. Both methods rank the same five generated candidates. The results show a use of internal readout for candidate selection, while token-wise autoregressive routing remains outside the current evaluation. The reranking protocol is reported in Appendix D.

6 Discussion and Conclusion

Our results suggest that internal representations can support task answers more effectively than final-state readout under a common supervised protocol. The pooled-final control attributes part of this difference to representation construction, while AIR retains a smaller residual advantage on stronger backbones. AAG is therefore a protocol-dependent diagnostic whose magnitude varies across models. Task-site analyses reveal different preferences across visual, multimodal, and language streams. AIR accesses these representations through sparse, input-dependent selection, although frequency-fixed routing explains much of the gain on larger models. Functional ablations indicate that selected states contribute to the learned readout pathway. These findings support a distinction between internal information availability and final-state accessibility without implying new reasoning capabilities in the frozen backbone.

Candidate reranking provides a limited interface with native generation, improving selection among responses produced by the original decoder. Cross-task experiments show partial selector transfer, although the target readout still requires supervision. These results suggest practical uses for internal readout within the current evaluation scope.

Limitations. The measured AAG depends on candidate sites, pooling rules, the supervised readout family, and oracle budget. Our evaluation covers three open multimodal backbones and six structured-output benchmarks; shrinking gains on stronger models do not establish behavior at frontier scale. Task-specific answer vocabularies count out-of-vocabulary targets as incorrect. AIR's main selector uses the final answer-side state and thus requires a full backbone forward pass. Its efficiency advantage concerns readout overhead relative to dense fusion. Reranking is limited by the decoder's generated candidates, and AIR does not support token-wise routing during long-form generation.

Future work. AAG may help assess whether architectures and training objectives improve final-state accessibility. Further work should examine broader model families, stronger cross-task transfer, and adaptive readout over evolving states during multi-step generation.

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, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *ArXiv*, abs/2502.13923, 2025a. URL https://api.semanticscholar.org/CorpusID:276449796.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025b.

Clement Farabet and Olivier Lacombe. Gemma 4: Byte for byte, the most capable open models. https://blog.google/innovation-and-ai/technology/developers-tools/gemma-4/, 2026. Google Blog. Accessed: 2026-04-15.

Mustafa Shukor, Enrico Fini, Victor Guilherme Turrisi da Costa, Matthieu Cord, Joshua Susskind, and Alaaeldin El-Nouby. Scaling laws for native multimodal models, 2025. URL https://arxiv.org/abs/2504.07951.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2017. URL https://openreview.net/forum?id=ryF7rTqgl.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, March 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7/.

Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Abdul Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Shang-Wen Li, Piotr Dollar, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=INqBOmwIpG.

Huanjin Yao, Wenhao Wu, Taojiannan Yang, Yuxin Song, Mengxi Zhang, Haocheng Feng, Yifan Sun, Zhiheng Li, Wanli Ouyang, and Jingdong Wang. Dense connector for mllms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 33108–33140. Curran Associates, Inc., 2024. doi: 10.52202/079017-1043. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/3a10c46572628d58cb44fb705f25cbbf-Paper-Conference.pdf.

Yue Cao, Yangzhou Liu, Zhe Chen, Guangchen Shi, Wenhai Wang, Danhuai Zhao, and Tong Lu. Mmfuser: Multimodal multi-layer feature fuser for fine-grained vision-language understanding. *arXiv preprint arXiv:2410.11829*, 2024.

Xu Li, Yi Zheng, Haotian Chen, Xiaolei Chen, Yuxuan Liang, Chenghang Lai, Bin Li, and Xiangyang Xue. Instruction-guided fusion of multi-layer visual features in large vision-language models. *Pattern Recognition*, 170:111932, 2026. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2025.111932. URL https://www.sciencedirect.com/science/article/pii/S0031320325005928.

Divya Jyoti Bajpai and Manjesh Kumar Hanawal. Free: Fast and robust vision language models with early exits, 2025. URL https://arxiv.org/abs/2506.06884.

nostalgebraist. Interpreting gpt: The logit lens. https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens, 2020. Accessed: 2026-04-15.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=2dnO3LLiJ1.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Nikul Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=WGXb7UdvTX.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models, 2024. URL https://arxiv.org/abs/2404.02258.

Theodore Glavas, Joud Chataoui, Florence Regol, Wassim Jabbour, Antonios Valkanas, Boris N. Oreshkin, and Mark Coates. Dynamic layer selection in decoder-only transformers, 2024. URL https://arxiv.org/abs/2410.20022.

Akriti Jain, Saransh Sharma, Koyel Mukherjee, and Soumyabrata Pal. FiRST: Finetuning router-selective transformers for input-adaptive latency reduction. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 21957–21975, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1197. URL https://aclanthology.org/2025.findings-emnlp.1197/.

Zhuomin He, Yizhen Yao, Pengfei Zuo, Bin Gao, Qinya Li, Zhenzhe Zheng, and Fan Wu. Adaskip: Adaptive sublayer skipping for accelerating long-context llm inference, 2025. URL https://arxiv.org/abs/2501.02336.

Xuan Luo, Weizhi Wang, and Xifeng Yan. Adaptive layer-skipping in pre-trained LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Gu0XSax2YS.

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. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L. Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 23716–23736. Curran Associates, Inc., 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: towards general-purpose vision-language models with instruction tuning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34892–34916. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6dcf277ea32ce3288914faf369fe6de0-Paper-Conference.pdf.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Dong Shu, Xuansheng Wu, Haiyan Zhao, Daking Rai, Ziyu Yao, Ninghao Liu, and Mengnan Du. A survey on sparse autoencoders: Interpreting the internal mechanisms of large language models, 2025. URL https://arxiv.org/abs/2503.05613.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum McDougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability, 2025. URL https://arxiv.org/abs/2503.09532.

Samuel Stevens, Wei-Lun Chao, Tanya Berger-Wolf, and Yu Su. Interpretable and testable vision features via sparse autoencoders, 2025. URL https://arxiv.org/abs/2502.06755.

Mateusz Pach, Shyamgopal Karthik, Quentin Bouniot, Serge Belongie, and Zeynep Akata. Sparse autoencoders learn monosemantic features in vision-language models, 2025. URL https://arxiv.org/abs/2504.02821.

Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D. Goodman. Quiet-star: Language models can teach themselves to think before speaking, 2024. URL https://arxiv.org/abs/2403.09629.

Ido Amos, Avi Caciularu, Mor Geva, Amir Globerson, Jonathan Herzig, Lior Shani, and Idan Szpektor. Latent reasoning with supervised thinking states, 2026. URL https://arxiv.org/abs/2602.08332.

Yiming Qin, Bomin Wei, Jiaxin Ge, Konstantinos Kallidromitis, Stephanie Fu, Trevor Darrell, and XuDong Wang. Chain-of-visual-thought: Teaching vlms to see and think better with continuous visual tokens, 2025. URL https://arxiv.org/abs/2511.19418.

Michael A. Lepori, Tal Linzen, Ann Yuan, and Katja Filippova. Language models struggle to use representations learned in-context, 2026. URL https://arxiv.org/abs/2602.04212.

Junchen Liu, Sven Elflein, Or Litany, Zan Gojcic, and Ruilong Li. Test-time training with kv binding is secretly linear attention, 2026. URL https://arxiv.org/abs/2602.21204.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. The right tool for the job: Matching model and instance complexities. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6640–6651, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.593. URL https://aclanthology.org/2020.acl-main.593/.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.204. URL https://aclanthology.org/2020.acl-main.204/.

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

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. *arXiv preprint arXiv:2412.04467*, 2024.

Pavan Kumar Anasosalu Vasu, Fartash Faghri, Chun-Liang Li, Cem Koc, Nate True, Albert Antony, Gokul Santhanam, James Gabriel, Peter Grasch, Oncel Tuzel, and Hadi Pouransari. Fastvlm: Efficient vision encoding for vision language models, 2025. URL https://arxiv.org/abs/2412.13303.

A Implementation Details and Budget Matching

This appendix section provides the concrete implementation details that support the parity protocol in the main text. We first define the candidate site space across backbones, including stream and depth allocations. We then specify the pooling rules used to construct site representations, the selector summary used by adaptive internal readout (AIR), and the exact training protocol for hard top-k selection. Finally, we report budget matching, runtime, and cross-model scaling statistics.

Candidate sites across backbones. AIR operates over a finite candidate site set $\mathcal{S}$ drawn from the internal stack of each backbone. We partition sites into three streams: visual (V), multimodal (M), and language-side (L). For analysis, each stream is further grouped into early, middle, and late depth bins, yielding the common full-stack grid used in the main text. AIR itself operates on the original candidate sites rather than on the coarse analysis bins. We use Gemma 4 E4B-it as the main analysis model with a regular 18-site full-stack configuration, while Qwen3-VL-8B-Instruct allocates more sites to the visual stream to reflect its stronger public emphasis on multi-level visual fusion, and Qwen3.5-27B uses a slightly denser site set to reflect its greater depth and width. Table A1 summarizes the site configuration and AIR hyperparameters across backbones, and Table A2 gives the stream-depth allocation in each model.

Table A1: Site definitions and AIR configuration across backbones. Public scale and native-context anchors are included to contextualize the site-space design. Gemma 4 E4B-it uses the most regular 18-site full-stack configuration; Qwen3-VL-8B-Instruct allocates more visual candidate sites; Qwen3.5-27B increases the total site count while keeping $k = 4$ fixed.

Model	Public scale anchor	Hidden size	Layers	Native context	$\lvert\mathcal{S}\rvert$	Stream split (V/M/L)	k	d_r	Selector width
Gemma 4 E4B-it	4.5B eff. / 8B total	3584 (est.)	42	128K	18	6 / 6 / 6	4	256	192
Qwen3-VL-8B-Instruct	9B	4096 (pub.)	32	256K	20	8 / 6 / 6	4	256	192
Qwen3.5-27B	27B	5120 (pub.)	64	256K	24	8 / 8 / 8	4	256	256

Hidden size is reported from public model documentation when available; for Gemma 4 E4B-it we use an implementation-side estimate sufficient for configuration reporting.

Table A2: Stream-depth allocation of candidate sites. Counts are reported in the order early / middle / late for each stream. Gemma uses the most regular full-stack layout, while Qwen3-VL and Qwen3.5 allocate more capacity to visual early/middle sites and mid/late multimodal-language regions, respectively.

Model	Visual (E / M / L)	Multimodal (E / M / L)	Language (E / M / L)
Gemma 4 E4B-it	2 / 2 / 2	2 / 2 / 2	2 / 2 / 2
Qwen3-VL-8B-Instruct	3 / 3 / 2	2 / 2 / 2	1 / 2 / 3
Qwen3.5-27B	3 / 3 / 2	2 / 3 / 3	1 / 3 / 4

Pooling rules and selector summary. To compare heterogeneous internal states under a common readout family, each candidate site is first reduced to a fixed-size site representation before projection into the shared readout dimension d_r. For visual sites, we apply masked mean pooling over visual tokens. For multimodal sites, we apply masked mean pooling over multimodal context tokens, excluding answer-side tokens so that multimodal summaries remain distinct from language-side summaries. For language-side sites, we use the hidden state at the answer position induced by the fixed prompt template, without teacher-forcing ground-truth answer tokens or introducing lookahead leakage. This choice keeps the language-side representation aligned with the model's standard output pathway while preserving a clear distinction between internal-site access and final-layer accessibility.

The selector summary $u(x)$ is taken from the final answer-side state. Importantly, $u(x)$ is used only as a compact query for site selection and is not itself used as the AIR readout representation. The information recovered by AIR still comes from the selected internal sites after stream-specific pooling and projection. The final-only learned head remains the main parity control for separating site selection from final-state accessibility.

Table A3: Answer-space construction for short-answer learned heads. Vocabularies are built only from training answers after dataset-specific normalization. Held-out OOV answers are counted as incorrect.

Dataset	Train vocab size	Held-out coverage	OOV handling
DocVQA	38,742	91.6%	Incorrect
ChartQA	6,214	87.3%	Incorrect
TallyQA	27	99.4%	Incorrect

Answer-space construction. For short-answer datasets, the learned answer vocabulary is built only from the training split after dataset-specific normalization. Held-out examples whose normalized gold answer is outside this vocabulary are counted as incorrect under learned-head evaluation. Table A3 reports the resulting vocabulary size and held-out answer coverage.

Training details for AIR. AIR uses hard top-k site selection during both training and inference. We do not use Gumbel-top-k, straight-through estimation, or temperature relaxation. Gradients flow only through the selected site projections and through the scorer values associated with the selected sites, while the backbone remains frozen throughout. All learned baselines use the same answer space, loss, training split, and evaluation protocol as described in the main text. The shared readout dimension is fixed to $d_r = 256$ in all reported AIR experiments, and the selector hidden width is tuned only to support approximate budget matching against the strongest fusion baselines.

Budget matching and runtime on Gemma 4 E4B-it. All runtime and memory measurements are obtained on a node with two NVIDIA A100 80GB GPUs using CUDA 12.1, PyTorch 2.5.1, Transformers 4.51.3, and bf16 inference. We use the same two-GPU inference configuration for all compared methods. Unless otherwise stated, latency is measured with batch size 1, benchmark-specific inputs after the standard model processor, 5 warmup runs, and 50 timed runs. We report relative overhead over the standard final-layer readout under the same input batch, precision, and serving configuration; peak activation memory is measured as the maximum per-GPU allocated memory during inference.

Table A4 reports budget matching and runtime for the main Gemma 4 E4B-it analysis model. The most important comparison group is the full-stack family of learned baselines: static learned fusion, adaptive scalar mix, dense full-stack fusion, random matched top-k, and AIR. These methods are kept within a narrow trainable-parameter band, which makes it unlikely that AIR's gains can be explained by a trivial capacity advantage. Final-only and early-exit readouts are intentionally lighter because they serve as parity and control baselines rather than full-stack competitors.

The runtime results are also important for interpretation. AIR reads only $k = 4$ sites at inference, while dense full-stack fusion reads all 18 candidate sites. As a result, AIR has nearly matched trainable parameters to dense fusion, but lower extra FLOPs and lower latency overhead. This supports the main claim that AIR improves over dense fusion because it performs sparse adaptive access to useful internal states, not because it simply reads more layers.

Cross-model AIR scaling. Table A5 reports AIR-only scaling statistics across backbones. As model size and candidate site count increase from Gemma to Qwen3-VL and then to Qwen3.5-27B, trainable parameters and measured runtime overhead rise smoothly rather than abruptly. This pattern is consistent with the larger hidden sizes and denser site spaces used in the stronger models. The scaling behavior is important for interpreting the cross-model results in the main text: the site-space and AIR module grow moderately with backbone scale, while the adaptive readout remains a lightweight module relative to the frozen backbone.

Existing assets and licenses. See table A6.

Layer-index selection. Candidate sites are selected deterministically before training. We divide each stream into early, middle, and late depth bins and choose evenly spaced interior block indices within each bin. All indices below are 0-based block indices in the corresponding stream and are fixed across all runs.

Table A4: Budget matching and runtime on Gemma 4 E4B-it. FLOPs, latency overhead, and peak activation memory overhead are measured in our implementation. Final-only and early-exit readouts are parity/control baselines; the main budget-matched comparison group is the full-stack family of learned baselines. For the final-only routed head, "candidate sites" refers to one backbone state with multiple learned final-state pseudo-experts; it matches AIR's routing capacity but never reads internal sites.

Method	Cand. sites	Sites read @ infer.	d_r	Params (M)	Extra FLOPs/ req. (GF)	Latency overhead	Peak activ. mem. ovhd.
Standard final-layer readout	0	0	–	0.00	0.00	+0.0%	+0.0%
Final-only learned head	1	1	256	1.34	0.42	+0.8%	+0.4%
Early-exit readout	1	1	256	1.37	0.44	+0.9%	+0.4%
Instruction-guided fusion	6	6	256	12.1	2.7	+4.2%	+3.6%
Static learned fusion	18	18	256	18.2	4.9	+6.8%	+5.8%
Adaptive scalar mix	18	18	256	18.5	6.1	+8.8%	+7.0%
Dense full-stack fusion	18	18	256	18.7	7.7	+12.4%	+9.5%
Random matched top-k	18	4	256	18.4	4.7	+6.8%	+5.9%
Capacity-matched final-only MLP	1	1	256	18.4	4.1	+5.0%	+1.3%
Final-only routed head	1	1	256	18.4	4.4	+5.6%	+1.6%
AIR (Ours)	18	4	256	18.4	4.8	+6.9%	+6.0%

Table A5: AIR-only scaling summary across backbones. FLOPs, latency overhead, and peak activation memory overhead are measured in our implementation. Site count grows moderately with model scale, while k is fixed to 4 in all reported experiments.

Model	$\lvert\mathcal{S}\rvert$	k	d_r	AIR params (M)	Extra FLOPs / req. (GF)	Latency overhead	Peak activ. mem. ovhd.
Gemma 4 E4B-it	18	4	256	18.4	4.8	+6.9%	+6.0%
Qwen3-VL-8B-Instruct	20	4	256	22.2	5.6	+7.7%	+6.6%
Qwen3.5-27B	24	4	256	33.4	8.3	+9.8%	+8.2%

Summary. Together, Tables A1–A5 clarify three implementation points that are central to the fairness of the AIR comparison. First, AIR operates over a concrete, explicitly enumerated candidate site space that is comparable across backbones but not artificially identical. Second, all learned baselines are evaluated under the same parity readout family and answer-space protocol. Third, the main full-stack competitors are approximately budget-matched, while AIR remains more efficient than dense fusion at inference because it reads only a small selected subset of internal sites.

B Additional Formalization and Metric Properties

This appendix section collects several simple formal properties of the parity readout protocol used in the main text. The goal is not to introduce a stronger theoretical claim than the paper supports, but to make explicit the objects being measured and the meaning of the resulting gap and closure metrics. We distinguish between the best *single-site* readout used for topology analysis and the small-budget *oracle internal envelope* used as the reference point in the main gap-closure tables.

Common readout family. Let $\mathcal{S} = \mathcal{S}_v \cup \mathcal{S}_m \cup \mathcal{S}_l$ denote the candidate site set, and let each site $s \in \mathcal{S}$ produce a pooled hidden state $h_s(x) \in \mathbb{R}^{d_s}$. As in the main text, every site is mapped into the shared readout dimension d_r by

$$z_s(x) = P_s h_s(x), \qquad P_s \in \mathbb{R}^{d_r \times d_s}.$$

The final model state is treated as an additional site,

$$z_{\text{final}}(x) = P_{\text{final}} h_{\text{final}}(x).$$

For a dataset with answer space $\mathcal{Y}$, every learned readout method R uses the same scoring form

$$\ell_R(x, y) = w_y^\top z_R(x) + b_y, \qquad y \in \mathcal{Y},$$

Table A6: Existing models and datasets used in this work. We use all assets for research evaluation and follow the licenses or terms specified by their original providers or dataset/model cards.

Asset	Type	License / terms used in our study
Gemma 4 E4B-it	Model	Apache-2.0 / provider model terms
Qwen3-VL-8B-Instruct	Model	Apache-2.0 model card
Qwen3.5-27B	Model	Apache-2.0 model card
DocVQA	Dataset	Official RRC challenge terms
ChartQA	Dataset	GPL-3.0 dataset card
AI2D	Dataset	Dataset-card license for the mirror used in our evaluation
TallyQA	Dataset	Apache-2.0 dataset card, using the VLMEvalKit mirror
NLVR2	Dataset	CC-BY-4.0 annotations; images follow original-owner terms
MMMU-Pro	Dataset	Apache-2.0 dataset card

Table A7: Exact candidate-layer indices used for each backbone. Indices are 0-based, fixed before training, and selected by deterministic spacing within stream-depth bins.

Model	Visual sites	Multimodal sites	Language-side sites
Gemma 4 E4B-it	`[4, 9, 18, 23, 32, 37]`	`[4, 9, 18, 23, 32, 37]`	`[4, 9, 18, 23, 32, 37]`
Qwen3-VL-8B-Instruct	`[2, 5, 8, 13, 16, 19, 25, 28]`	`[3, 7, 14, 18, 25, 29]`	`[6, 14, 18, 24, 27, 30]`
Qwen3.5-27B	`[5, 10, 15, 26, 32, 37, 50, 57]`	`[7, 14, 26, 32, 38, 48, 53, 59]`	`[10, 26, 32, 38, 47, 51, 56, 60]`

and is trained under the same empirical cross-entropy objective

$$\mathcal{L}(R) = \frac{1}{|\mathcal{D}_{\mathrm{tr}}|} \sum_{(x,y)\in\mathcal{D}_{\mathrm{tr}}} -\log \frac{\exp \ell_R(x,y)}{\sum_{y'\in\mathcal{Y}} \exp \ell_R(x,y')}.$$

This makes all learned readouts directly comparable under a shared answer space and loss.

AIR as sparse adaptive aggregation. Adaptive internal readout defines a scorer

$$a_s(x) = g_\theta(u(x), e_s),$$

and a hard selected set of site indices

$$\mathcal{T}_k(x) = \arg\operatorname*{topk}_{s\in\mathcal{S}} a_s(x),$$

with ties broken by a fixed site order. The selected scores are normalized over the selected sites,

$$\alpha_s(x) = \frac{\exp(a_s(x))}{\sum_{j\in\mathcal{T}_k(x)} \exp(a_j(x))}, \qquad s \in \mathcal{T}_k(x).$$

The AIR representation is then

$$z_{\mathrm{adapt}}(x) = \sum_{s\in\mathcal{T}_k(x)} \alpha_s(x)\, z_s(x).$$

Thus AIR remains inside the same readout family: it changes *which* internal states are exposed to the readout, not the scoring function itself.

Single-site availability and oracle internal envelopes. For a single internal site s, let r_s be the validation score of the learned head attached to site s. We define the best single-site score as

$$r_{\mathrm{single}} = \max_{s\in\mathcal{S}} r_s.$$

This quantity is used for single-site topology analysis and for identifying which individual streams or depths contain useful information.

For the main gap-closure tables, we use a small-budget internal oracle envelope because AIR itself reads from multiple internal sites. Let $\mathcal{R}_{\text{int}}^{(K)}$ denote the class of oracle internal readouts that may use at most K internal sites under the same answer space and loss. We define

$$r_{\text{env}}^{(K)} = \max_{R \in \mathcal{R}_{\text{int}}^{(K)}} r_R.$$

In the main cross-backbone gap-closure table, we use $K = 2$, corresponding to the oracle top-2 site readout. This envelope is a diagnostic reference selected using validation labels; AIR does not receive oracle site labels at test time.

Gap definitions. Let $r_{\text{final}}^{\text{def}}$ be the score of the model's default final-layer decoding, and let $r_{\text{final}}^{\text{head}}$ be the score of the learned final-only head. We define the default-decoding and parity gaps against the oracle internal envelope as

$$\text{AAG}_{\text{def}}^{(K)} = r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{def}}, \qquad \text{AAG}_{\text{head}}^{(K)} = r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{head}}.$$

For any learned readout R,

$$\text{GapClosed}(R) = r_R - r_{\text{final}}^{\text{head}},$$

and, when $r_{\text{env}}^{(K)} > r_{\text{final}}^{\text{head}}$,

$$\text{ClosureRatio}^{(K)}(R) = \frac{r_R - r_{\text{final}}^{\text{head}}}{r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{head}}}.$$

This makes the learned final-only head the zero point and the oracle internal envelope the reference point for learned-readout comparisons. When we discuss single-site availability, we explicitly refer to r_{single} rather than $r_{\text{env}}^{(K)}$.

Proposition 1 (default-vs-parity gap decomposition). For any fixed oracle-envelope budget K, the default-decoding gap and the parity gap differ only by the performance gap between the learned final-only head and the default final decoding:

$$\text{AAG}_{\text{def}}^{(K)} = \text{AAG}_{\text{head}}^{(K)} + \left(r_{\text{final}}^{\text{head}} - r_{\text{final}}^{\text{def}}\right).$$

Proof. By definition,

$$\text{AAG}_{\text{def}}^{(K)} - \text{AAG}_{\text{head}}^{(K)} = \left(r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{def}}\right) - \left(r_{\text{env}}^{(K)} - r_{\text{final}}^{\text{head}}\right) = r_{\text{final}}^{\text{head}} - r_{\text{final}}^{\text{def}}.$$

Rearranging gives the result.

Interpretation. Proposition 1 explains why $\text{AAG}_{\text{head}}^{(K)}$ is the stricter quantity. It subtracts out the part of the gain that can be obtained simply by replacing default decoding with a learned final-state head.

Proposition 2 (zero point of the parity protocol). Let R_{final} denote the learned final-only head. Then

$$\text{GapClosed}(R_{\text{final}}) = 0.$$

Moreover, whenever $r_{\text{env}}^{(K)} > r_{\text{final}}^{\text{head}}$,

$$\text{ClosureRatio}^{(K)}(R_{\text{final}}) = 0.$$

Proof. Since $r_{R_{\text{final}}} = r_{\text{final}}^{\text{head}}$,

$$\text{GapClosed}(R_{\text{final}}) = r_{\text{final}}^{\text{head}} - r_{\text{final}}^{\text{head}} = 0.$$

Substituting into the definition of $\text{ClosureRatio}^{(K)}$ yields 0 as well.

Interpretation. The learned final-only head is therefore the natural zero point for learned-readout comparisons. Any positive $\text{GapClosed}(R)$ means that method R improves over a parity head restricted to the final state alone.

Proposition 3 (oracle-envelope normalization). Let $R_{\mathrm{env}}^{(K)}$ denote an oracle internal readout in $\mathcal{R}_{\mathrm{int}}^{(K)}$ such that

$$r_{R_{\mathrm{env}}^{(K)}} = r_{\mathrm{env}}^{(K)}.$$

Then, whenever $r_{\mathrm{env}}^{(K)} > r_{\mathrm{final}}^{\mathrm{head}}$,

$$\mathrm{GapClosed}(R_{\mathrm{env}}^{(K)}) = \mathrm{AAG}_{\mathrm{head}}^{(K)}, \qquad \mathrm{ClosureRatio}^{(K)}(R_{\mathrm{env}}^{(K)}) = 1.$$

Proof. The first identity follows directly:

$$\mathrm{GapClosed}(R_{\mathrm{env}}^{(K)}) = r_{\mathrm{env}}^{(K)} - r_{\mathrm{final}}^{\mathrm{head}} = \mathrm{AAG}_{\mathrm{head}}^{(K)}.$$

Substituting this into the definition of $\mathrm{ClosureRatio}^{(K)}$ gives 1.

Interpretation. This gives the normalized meaning of closure in the main gap-closure tables: the oracle internal envelope defines the reference point, and the learned final-only head defines zero closure. In the main experiments, this reference is the oracle top-2 readout.

Proposition 4 (ranking preservation under fixed reference). Fix a model, task, candidate site set, and oracle-envelope budget K such that

$$r_{\mathrm{env}}^{(K)} > r_{\mathrm{final}}^{\mathrm{head}}.$$

For any two learned readout methods R_1 and R_2, if

$$r_{R_1} \geq r_{R_2},$$

then

$$\mathrm{GapClosed}(R_1) \geq \mathrm{GapClosed}(R_2) \quad \text{and} \quad \mathrm{ClosureRatio}^{(K)}(R_1) \geq \mathrm{ClosureRatio}^{(K)}(R_2).$$

Proof. The denominator of $\mathrm{ClosureRatio}^{(K)}$ is fixed and positive under the stated condition. Both $\mathrm{GapClosed}(R)$ and $\mathrm{ClosureRatio}^{(K)}(R)$ are therefore monotone increasing affine transformations of r_R.

Interpretation. Under a fixed model-task-site configuration and fixed reference envelope, ranking learned readouts by raw score, absolute gap closed, or normalized closure yields the same ordering. The metrics change interpretation, not which method is better.

Remark 1 (bounded closure under the oracle envelope). If a learned readout R satisfies

$$r_{\mathrm{final}}^{\mathrm{head}} \leq r_R \leq r_{\mathrm{env}}^{(K)},$$

then, whenever $r_{\mathrm{env}}^{(K)} > r_{\mathrm{final}}^{\mathrm{head}}$,

$$0 \leq \mathrm{ClosureRatio}^{(K)}(R) \leq 1.$$

This bounded regime is the intended interpretation of the normalized closure metric: values near 0 behave like the final-only parity head, and values near 1 recover most of the oracle-envelope reference. If a learned method exceeds the chosen reference envelope, the ratio can exceed 1, which simply indicates that the reference class was not an upper bound for that method.

Remark 2 (single-site AAG versus oracle-envelope closure). The best single-site score r_{single} and the oracle-envelope score $r_{\mathrm{env}}^{(K)}$ serve different purposes. The former measures whether individual internal sites contain task-supporting information and is used for topology analysis. The latter provides a small-budget reference for closure analysis when the learned method itself may aggregate multiple internal sites. This distinction avoids conflating single-site availability with multi-site recoverability.

Table A8: Main robustness summary on Gemma 4 E4B-it. We report mean score and standard deviation over 3 readout-training seeds for the main learned-readout baselines.

Method	Avg. score	Std. over 3 seeds	Mean $\pm$ std range	Gain vs. final-only	Gain vs. dense
Standard final-layer	74.4	0.00	[74.4, 74.4]	—	—
Final-only head	75.2	0.26	[74.7, 75.6]	+0.0	—
Static learned fusion	77.4	0.25	[76.9, 77.8]	+2.2	-0.5
Adaptive scalar mix	77.7	0.29	[77.2, 78.2]	+2.5	-0.2
Dense full-stack	77.9	0.26	[77.5, 78.4]	+2.7	+0.0
Capacity-matched final-only MLP	75.8	0.24	[75.2, 76.4]	+0.6	-2.1
Final-only routed head	76.0	0.22	[75.5, 76.5]	+0.8	-1.9
AIR (Ours)	**78.9**	**0.20**	[78.5, 79.3]	**+3.7**	**+1.0**

Table A9: Cross-model robustness summary. We report mean $\pm$ std over 3 seeds and Mean $\pm$ stds for AIR gains relative to the final-only learned head and dense full-stack fusion.

Model	Final-only head	Dense fusion	**AIR**	AIR $-$ Final Head		AIR $-$ Dense Fusion	
				Gain	Mean $\pm$ std	Gain	Mean $\pm$ std
Gemma 4 E4B-it	75.2 ± 0.26	77.9 ± 0.26	$\mathbf{78.9 \pm 0.20}$	$+3.7 \pm 0.17$	[3.2, 4.2]	$+1.0 \pm 0.14$	[0.6, 1.4]
Qwen3-VL-8B-Instruct	83.6 ± 0.14	84.2 ± 0.16	$\mathbf{84.5 \pm 0.18}$	$+0.9 \pm 0.10$	[0.6, 1.2]	$+0.3 \pm 0.10$	[0.0, 0.6]
Qwen3.5-27B	87.7 ± 0.11	88.2 ± 0.14	$\mathbf{88.5 \pm 0.15}$	$+0.8 \pm 0.08$	[0.5, 1.0]	$+0.3 \pm 0.07$	[0.1, 0.5]

Remark 3 (protocol dependence of AAG). AAG is not an intrinsic property of the frozen backbone alone. It depends on three design choices: the candidate site set $\mathcal{S}$, the pooling rules used to construct $h_s(x)$, and the common readout family used to define r_s, $r_{\text{env}}^{(K)}$, and $r_{\text{final}}^{\text{head}}$. The metric is therefore best interpreted as a controlled diagnostic under a specified protocol, rather than as an absolute statement about everything represented in the model.

Remark 4 (what hard top-k changes). Because AIR uses hard top-k selection, the selected set $\mathcal{T}_k(x)$ is discrete and instance-dependent. This affects the *pathway of access* to internal information, but it does not change the parity readout family itself. AIR therefore remains a readout-location intervention rather than a backbone adaptation method.

C Robustness and Sensitivity Analyses

This appendix section evaluates whether the main AIR results depend on a narrow choice of seed or configuration. We focus on three questions raised by the main text and by reviewer concerns: whether the gains are stable across random seeds, whether AIR is sensitive to the number of selected sites, and whether the conclusions depend strongly on the choice of readout space. We also report selector-stability diagnostics for hard top-k training.

Variance across seeds. Table A8 reports the main robustness summary on Gemma 4 E4B-it. AIR remains the strongest non-oracle method under repeated runs, reaching 78.9 ± 0.20 average score over three readout-training seeds, compared with 77.9 ± 0.26 for dense full-stack fusion and 75.2 ± 0.26 for the final-only learned head. The same pattern appears at the cross-model level in Table A9: AIR improves over the final-only learned head by $+3.7 \pm 0.17$ on Gemma 4 E4B-it, $+0.9 \pm 0.10$ on Qwen3-VL-8B-Instruct, and $+0.8 \pm 0.08$ on Qwen3.5-27B. These gains shrink on stronger models but remain positive across repeated runs, consistent with the interpretation that stronger backbones reduce the availability–accessibility gap without fully removing it.

Readout-Mismatch Stress Suite (RMSS). RMSS is a set of controlled readout stress tests used only for topology visualization. The five variants are designed to probe whether the readout favors evidence localized to early states, middle states, late states, conflicting sites, or phase-shifted evidence

Table A10: Selector stability diagnostics for AIR. We report the mean selector entropy, mean top-k overlap across seeds, the fraction of selection mass assigned to the top-8 most frequently selected sites on 500 validation examples, and whether selector collapse was observed.

Model	Mean selector entropy	Mean top-k overlap	Top-8 site mass	Collapse observed
Gemma 4 E4B-it	1.07	0.58	71%	No
Qwen3-VL-8B-Instruct	0.99	0.63	76%	No
Qwen3.5-27B	0.92	0.67	79%	No

Table A11: Readout-Mismatch Stress Suite (RMSS) variants used for topology analysis. All variants use balanced binary labels and accuracy as the metric. RMSS rows are excluded from the main benchmark average and cross-model score tables.

Variant	#Examples	Controlled evidence pattern	Used for
RMSS-Early	500	Evidence is most recoverable from early sites	Topology only
RMSS-Mid	500	Evidence is most recoverable from middle sites	Topology only
RMSS-Late	500	Evidence is most recoverable from late sites	Topology only
RMSS-Conflict	500	Competing cues are placed across site groups	Topology only
RMSS-Phase	500	Relevant evidence shifts across depth phases	Topology only

across depth. RMSS rows are not included in the six-benchmark average or in the main cross-model score tables. Table A11 summarizes the construction.

Sensitivity to the number of selected sites. Figure A1(a) shows AIR performance as a function of the selected site budget k. Across all three backbones, performance rises quickly from $k = 1$ to $k = 4$, then saturates or slightly declines. On Gemma 4 E4B-it, AIR improves from 77.4 at $k = 1$ to 78.5 at $k = 2$ and 78.9 at $k = 4$, then stays nearly flat at 78.8 for $k = 8$ and decreases slightly to 78.6 for $k = 16$. Qwen3-VL-8B-Instruct and Qwen3.5-27B show the same qualitative trend, but with flatter curves, again matching the smaller parity gaps observed in the main text. The main implication is that AIR does not benefit from simply reading more sites. A small selected set is already sufficient to recover most of the available gain, and the default choice $k = 4$ lies near the best point for all reported backbones.

The task-level results on Gemma 4 further show that the sensitivity to k is not uniform. The average sensitivity curve shows that performance improves up to a moderate site budget and saturates beyond that point. MMMU-Pro improves more gradually but does not continue to benefit from larger k once the selected-site budget exceeds the main setting. These trends are consistent with the main-text claim that structurally harder multimodal tasks often require broader internal access than local perceptual tasks, but that the useful access pattern remains sparse.

Sensitivity to readout space. Figure A1(b) reports task-level sensitivity to the readout space on Gemma 4 E4B-it. The overall ranking is consistent with the main text: full-stack AIR gives the strongest and most consistent gains, but the margin depends on the task. Visual-only access is already competitive on DocVQA, AI2D, and TallyQA, while NLVR2 and MMMU-Pro depend much more strongly on multimodal-language integration. For example, NLVR2 improves by only $+1.8$ points under visual-only readout, but by $+4.6$ under multimodal+language access and by $+5.3$ under full-stack AIR. MMMU-Pro shows a similar pattern, rising from $+0.5$ under visual-only access to $+3.6$ under full-stack access. These results reinforce the interpretation that AIR is not merely a stronger visual fusion module. The best readout space depends on the reasoning pattern required by the task, and broad multimodal reasoning often requires access to later multimodal or language-side states.

The same ordering also holds in the cross-model family-level summaries. On Gemma 4 E4B-it, the average gain rises from $+2.3$ under visual-only access to $+4.5$ under full-stack AIR. On Qwen3-VL-8B-Instruct and Qwen3.5-27B, the absolute margins are smaller, but the same progression remains visible: full-stack access is consistently stronger than visual-only access and slightly stronger than the partial two-stream alternatives. This supports the claim that the usefulness of full-stack access is structural rather than model-specific.

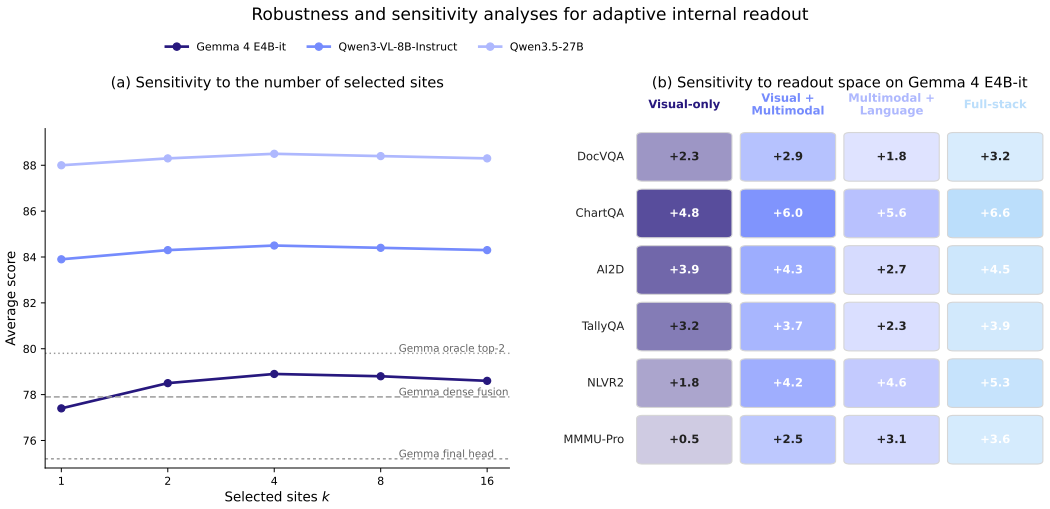


Figure A1: **Robustness and sensitivity analyses for adaptive internal readout. Panel (b) uses standard final-layer decoding as the baseline; Appendix D reports companion parity-head comparisons.**

Selector stability. Table A10 reports additional diagnostics for the AIR selector. Across all three backbones, the selector entropy remains moderate and the mean top-k overlap across seeds is well above chance, indicating that the hard top-k policy does not collapse to a degenerate selection pattern. The concentration of probability mass on a small shared subset of sites is also stronger in the larger models, which is consistent with their flatter k-sensitivity curves. Importantly, we do not observe evidence of selector collapse in the reported runs. Together with the variance and sensitivity results above, these diagnostics suggest that the AIR gains are not driven by fragile optimization behavior, but by a stable preference for a small number of task-relevant internal sites.

D Additional Evaluations and Ablations

This section reports additional evaluations that further test the scope and boundary conditions of adaptive internal readout. We focus on three questions. First, how sensitive is AIR to the choice of selector summary used to score candidate sites? Second, how should the parity-head protocol be interpreted relative to standard generative decoding on short-answer benchmarks? Third, do the task-by-site topology patterns observed in the main text persist on stronger backbones? The results reported here support the same overall picture as the main text: AIR benefits from selective access to useful internal states, and this effect is not tied to a single summary choice or a single backbone.

Alternative selector summaries. Table A12 compares several choices for the selector summary $u(x)$ on Gemma 4 E4B-it. The main implementation, which uses the final answer-side state as the selector query, gives the strongest overall result with an average score of 78.9. Question-only, image-only, and shallow multimodal summaries all remain competitive, but perform worse on average, especially on ChartQA, NLVR2, and MMMU-Pro. This pattern is consistent with the role of AIR in the main method: the selector does not read out the final state directly, but it still benefits from a compact query that reflects the model's current answer-side context. At the same time, the alternative summaries remain functional, which suggests that the AIR effect is not tied to a single fragile implementation choice.

Parity-head versus generative evaluation. Table A13 compares standard generative decoding, the final-only learned head, and AIR on selected short-answer datasets. The learned parity head gives a modest improvement over default decoding, while AIR produces a much larger additional gain. For example, on ChartQA the score rises from 72.4 under standard generative decoding to 73.1 under the final-only learned head, and then to 79.0 under AIR. A similar pattern appears on DocVQA and MMMU-Pro. This comparison clarifies the interpretation of the main protocol. The parity-head setup

Table A12: Alternative selector summaries for AIR on Gemma 4 E4B-it. The main implementation uses the final answer-side state as the selector query.

Selector summary source	DocVQA	ChartQA	AI2D	TallyQA	NLVR2	MMMU-Pro	Avg.
Final answer-side	**87.1**	**79.0**	**83.1**	**91.8**	**76.1**	**56.2**	**78.9**
Question-only	86.6	78.1	82.4	91.2	74.8	55.4	78.1
Image-only	86.4	77.5	82.1	91.0	73.9	54.7	77.6
Shallow multimodal	86.9	78.5	82.8	91.5	75.4	55.8	78.5

does not replace standard generation as the default usage mode of the model. Instead, it provides a controlled way to separate gains that come from attaching a supervised final-state head from gains that come from changing where the model reads from internally.

Table A13: Standard generative decoding versus parity-head evaluation on selected datasets. AIR is compared against both the model's default output and the learned final-only parity head.

Dataset	Standard generative	Final-only head	AIR	AIR − generative	AIR − final head
DocVQA	83.9	84.6	87.1	+3.2	+2.5
ChartQA	72.4	73.1	79.0	+6.6	+5.9
MMMU-Pro	52.6	53.7	56.2	+3.6	+2.5

Additional topology results across stronger backbones. Figure A2 reports companion task-by-site topology results on Qwen3-VL-8B-Instruct and Qwen3.5-27B, using score deltas relative to the final-only learned head. The same qualitative pattern observed in the main text remains visible. Local perceptual tasks such as DocVQA, AI2D, and TallyQA tend to favor earlier visual or multimodal sites, while NLVR2 and MMMU-Pro depend more strongly on later multimodal and language-side states. The absolute gains are smaller than on Gemma 4 E4B-it, which is consistent with the smaller parity gaps reported in the main text and Appendix C. However, the structure of the topology remains stable across models. This supports the view that AIR is capturing a general readout-mismatch phenomenon rather than overfitting to a single backbone.

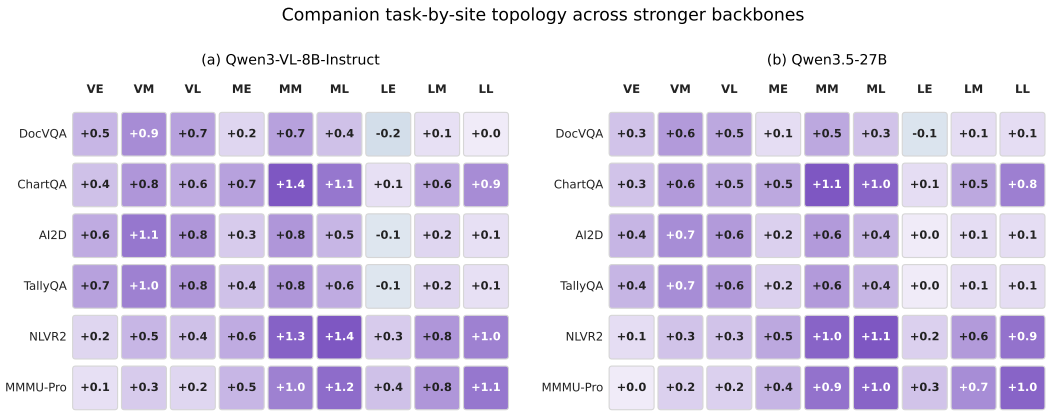


Figure A2: **Companion topology results across stronger backbones.** Task-by-site score deltas relative to the final-only learned head on Qwen3-VL-8B-Instruct and Qwen3.5-27B. The same qualitative pattern as in the main text remains visible: local perceptual tasks favor earlier visual or multimodal sites, while structural and broad multimodal reasoning depend more on later multimodal and language-side states.

Summary. Taken together, these additional evaluations reinforce three points. First, the AIR effect is robust to reasonable changes in the selector summary, though the final answer-side summary

remains the strongest choice in our implementation. Second, the parity-head protocol is useful precisely because it factors out the smaller gain obtained by attaching a supervised head to the final state alone. Third, the task-dependent internal topology described in the main text is not unique to Gemma 4 E4B-it, but persists on stronger native multimodal backbones with smaller overall gaps.

E Extended Discussion and Broader Context

This appendix section discusses the scope of the availability–accessibility gap (AAG) and clarifies how AIR should be interpreted relative to neighboring directions. The main text establishes that useful task information can be internally available before it becomes fully accessible to the model's standard final-layer readout. Appendix A and Appendix C further show that this conclusion is robust to the concrete site design, matched-budget controls, and sensitivity analyses. Appendix D shows that the effect is not tied to a single selector summary or to a single backbone. Here we discuss what these results do and do not imply.

Scope of AAG as a diagnostic. AAG is best interpreted as a controlled diagnostic under a specified protocol rather than as an intrinsic scalar property of a backbone in isolation. In our setup, the measured gap depends on the candidate site set $\mathcal{S}$, the pooling rules used to construct $h_s(x)$, and the common supervised readout family used to define r_s and $r_{\text{final}}^{\text{head}}$. A larger or denser site set may expose more recoverable information, while a coarser or more weakly aligned site set may miss useful internal states. For this reason, the main claim of the paper is not that AAG provides an absolute upper bound on what the model "really knows," but that it offers a principled way to separate internal availability from final-state accessibility under matched conditions.

AIR as a supervised readout adaptor. AIR should also be interpreted carefully. It is not a zero-shot decoding method and it is not a replacement for standard generative inference in all regimes. The method trains task-specific readout modules under a fixed answer space, then uses a sparse adaptive policy to decide where to read from. The resulting gains therefore show that some existing model capability is not fully exposed by the default output pathway. They do not show that AIR creates new reasoning ability inside the frozen backbone. This distinction is important when interpreting both the main-text gains and the parity comparisons in Appendix D. The parity-head protocol is useful precisely because it factors out the smaller gain obtained by attaching a supervised head to the final state alone.

Practical deployment and efficiency. From a systems perspective, AIR occupies a middle ground between simple final-state readout and dense internal fusion. As shown in Appendix A, AIR requires access to internal activations and therefore incurs nonzero serving complexity. However, it reads only a small selected subset of sites at inference, rather than aggregating the full internal stack. This makes it substantially lighter than dense full-stack fusion in measured FLOPs, latency overhead, and peak activation memory overhead. In that sense, AIR is not only a diagnostic but also a practical sparse adaptor for frozen backbones. At the same time, production deployment would still require engineering support for surfacing intermediate activations and maintaining alignment between internal-site definitions and model versions. The method is therefore easiest to apply in research or controlled serving environments, and less natural in black-box settings where internal states are inaccessible.

Relation to neighboring directions. AIR is related in spirit to several neighboring lines of work, but it is motivated by a different objective. Dynamic depth, layer routing, and adaptive computation methods typically aim to allocate computation more efficiently or to skip unnecessary layers. AIR instead uses adaptive selection to measure and recover accessibility gaps under a parity readout protocol. Likewise, visual fusion and learned layer aggregation methods show that combining intermediate features can improve downstream behavior, but they usually treat those features as useful auxiliary evidence rather than as evidence of a mismatch between internal availability and standard output accessibility. The distinction matters because our primary object is the gap itself, not merely a stronger fusion head.

AIR is also complementary to training-time approaches that try to reduce modality imbalance, preserve visual specificity, or strengthen multimodal semantic retention. If such objectives are successful,

one would expect the measured AAG to shrink, since more of the useful internal information should become accessible at the model's default output state. In this sense, AAG can serve not only as a post hoc diagnostic, but also as an evaluation tool for future training objectives and architectural choices.

Harder reasoning settings. Our experiments focus on short-answer and multiple-choice style evaluation, where a common supervised readout family can be defined cleanly. This gives a controlled setting for separating final-state accessibility from internal availability, but it also limits the scope of the present results. Long-form generation, multi-sentence responses, and multi-step chain-of-thought style reasoning may involve a more distributed notion of accessibility that is not fully captured by the current parity-head setup. Similarly, tasks that externalize reasoning through code, charts, tools, or structured intermediate representations may expose different kinds of mismatch between internal information and final answers. We therefore view the present paper as establishing the diagnostic and empirical phenomenon in a controlled regime, rather than exhausting all possible forms of multimodal readout mismatch.

Future directions. Several directions follow naturally from the current results. One is transfer: whether a selector or readout policy trained on one task family can generalize to related tasks without retraining. Another is training-time reduction of AAG: whether architectural changes or new objectives can make useful multimodal information more consistently accessible at the final readout location. A third is extension to richer generative settings, where accessibility may need to be measured over trajectories or multi-step outputs rather than over a fixed answer space. More broadly, the distinction between information availability and output accessibility may provide a useful framework for thinking about multimodal model failures in a way that is more precise than simply asking whether a model has or lacks a capability.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's main contributions, scope, and evaluation setting, and these claims are matched by the empirical results in Secs. 1, 4, and 5.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in the discussion section and Appendix E, including dependence on the candidate site set, the supervised parity-head protocol, the focus on short-answer and multiple-choice evaluation, and the limited coverage of long-form generation settings.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present formal theoretical results such as theorems with substantive proof obligations. It provides definitions, metric properties, and simple formal clarifications of the parity-readout protocol in the appendix, but the contribution is primarily empirical and diagnostic rather than theoretical.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper specifies the backbones, datasets, evaluation protocol, parity-readout setting, candidate site construction, readout definitions, adaptive selection mechanism, baseline design, and implementation details needed to reproduce the main results. Additional details on site definitions, budget matching, and sensitivity analyses are provided in the appendix.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper uses public datasets and publicly available backbone models, and provides detailed experimental and implementation information in the appendix, but anonymized code and full reproduction scripts are not provided with the submission.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the datasets, task settings, candidate site construction, readout family, adaptive selection configuration, matched baselines, and analysis protocol in the main text, with additional implementation and budget details in Appendix A.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The appendix reports robustness statistics, including means and standard deviations over multiple readout-training seeds for the main learned-readout comparisons.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix A reports parameter counts, FLOPs, latency, activation-memory overhead, and the hardware/software configuration used for runtime measurements.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: To the best of our knowledge, the research conforms to the NeurIPS Code of Ethics. The paper studies frozen publicly available multimodal backbones and public benchmarks, and does not involve human subjects, private user data, or unsafe deployment.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper and appendix discuss both positive and negative implications, including the potential value of diagnosing readout failures in multimodal systems and the risks of over-interpreting internal accessibility gains or applying such techniques in settings with limited transparency or accountability.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new high-risk generative model, scraped dataset, or safety-sensitive asset. It studies existing publicly available backbones and public evaluation datasets.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Appendix A includes an asset table listing all backbone models and evaluation datasets, together with the licenses or provider terms followed in this work.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce and release a new dataset, model, or benchmark as a standalone asset.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The core methodology explicitly operates on frozen native multimodal language models, including Gemma 4 E4B-it, Qwen3-VL-8B-Instruct, and Qwen3.5-27B. Their role in the method and experiments is described in the experimental setup, method, and appendix.