NEOPRUNE: NON-RECONSTRUCTIBLE ORTHOGONAL TOKEN PRUNING FOR OMNI MODAL MODELS

Anonymous authors
Paper under double-blind review

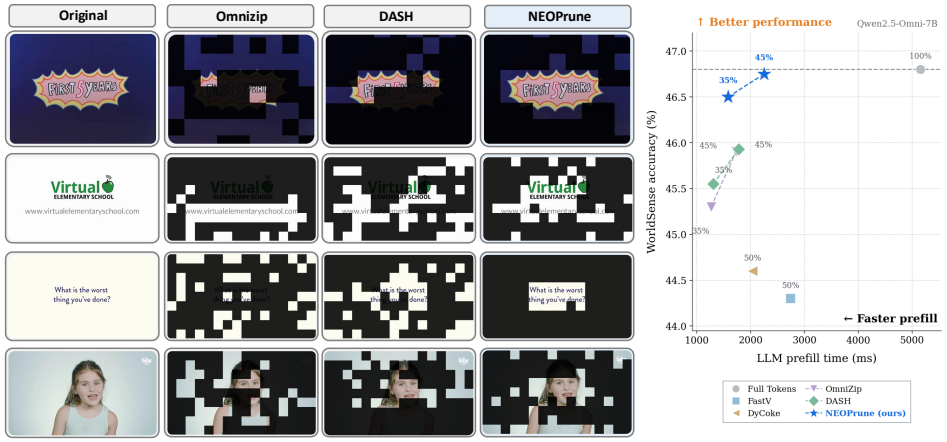

Figure 1: **Token selection and efficiency–performance trade-off.** NEOPrune selects informative tokens, achieving comparable WorldSense accuracy to full-token inference with lower LLM prefill latency (measured on the longest sample).

ABSTRACT

Omni-modal large language models (OmniLLMs) process large numbers of audio and video tokens, incurring substantial computational and memory costs. Existing token pruning methods primarily use token importance or pairwise similarity, leaving the collective replaceability of layer-wise token updates underexplored. We introduce **NEOPrune**, a training-free two-stage method that estimates replaceability by reconstructing orthogonalized layer-wise updates, focusing on information newly introduced at each layer. In the Pre-LLM stage, it combines within-modality reconstruction with encoder attention; in the Inner-LLM stage, it estimates video-update replaceability conditioned on audio and combines it with text-query attention. Across four audio–visual benchmarks and multiple OmniLLMs, NEOPrune generally achieves higher accuracy than evaluated pruning baselines at matched token-retention ratios, while retaining near-full-token performance with only 35–45% of audio–visual tokens.

1 INTRODUCTION

Multimodal large language models (MLLMs) extend the understanding and reasoning capabilities of language models to diverse input modalities. Following vision-language models (VLMs), which jointly process textual and visual information (Liu et al., 2023; Li et al., 2023; Wang et al., 2024; Dai et al., 2023; Chen et al., 2024b; Bai et al., 2025), recent advances have introduced omnimodal large language models (OmniLLMs) that jointly process audio, video, and text within a unified framework (Shu et al., 2025; Xu et al., 2025a;b). However, audio and video inputs generate long token sequences, resulting in substantial computational and memory costs. Their spatial and temporal continuity also causes similar information to be repeatedly represented across neighboring tokens, while audio and video corresponding to the same temporal segment may contain overlapping information.

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

Token compression that exploits such redundancy is therefore becoming increasingly important for efficient OmniLLM inference.

Training-free token compression for reducing such redundancy has primarily been studied for visual tokens in VLMs. Existing methods evaluate token importance based on the attention received by each token (Chen et al., 2024a; Zhang et al., 2025b), or reduce redundancy using feature similarity or distance (Bolya et al., 2023; Tao et al., 2025; Alvar et al., 2025). More recent methods use the updates added to tokens by attention and FFN modules to assess token importance (Li et al., 2026). Reconstruction-based approaches use the error of reconstructing each token from a small token set as an importance metric, aiming to preserve representations that are difficult to explain through token combinations under a limited token budget (Ma et al., 2026). These studies largely focus on visual token selection and do not explicitly consider relationships between audio and video tokens. Token compression research has recently extended to OmniLLMs, which process audio and video jointly.

Representative OmniLLMs comprise a Pre-LLM stage, in which modality-specific encoders process tokens independently, and an Inner-LLM stage, in which the LLM processes both modalities jointly (Xu et al., 2025a;b). OmniZip and DASH are training-free methods for OmniLLMs that use audio information to compress audio and video tokens jointly, selecting tokens based on encoder attention, pairwise token similarity, and representation density (Tao et al., 2026; Li & Huang, 2026). However, their compression criteria do not directly assess collective redundancy: whether the information carried by one token can be explained by a combination of other tokens. Both methods complete compression in the Pre-LLM stage and therefore do not exploit the audio–visual relationships that emerge during Inner-LLM processing. Their selection criteria also do not incorporate text-query relevance.

We incorporate both collective redundancy and text-query relevance into token selection. Accumulated token states can obscure changes introduced at individual layers, motivating us to extend reconstruction-based token selection to layer-wise updates. Specifically, we analyze the updates added to each token by the attention and FFN modules of Transformer blocks (Vaswani et al., 2017). Li et al. (2026) uses each token's own updates to assess its importance. We evaluate through reconstruction error whether other token updates can collectively explain each update. Our encoder analysis shows that the component orthogonal to the normalized module input generally accounts for most of the update magnitude. Reducing this component using the parallel-component norm as the reference yields larger observed mean accuracy drops than removing the parallel component entirely (Appendix A.2). Motivated by these observations, we reconstruct each token's orthogonal update from other token updates and use the resulting reconstruction error to guide token selection.

We propose **NEOPrune**, a training-free two-stage token pruning method that applies this update-level reconstruction criterion to both the Pre-LLM and Inner-LLM stages. In the **Pre-LLM stage**, we reconstruct video updates from spatial neighbors and audio updates from temporal neighbors, and select tokens by combining reconstruction error with encoder attention-based importance. In the **Inner-LLM stage**, we reconstruct each video token's orthogonal update from audio-token updates in the same temporal segment and combine the resulting error with text-query attention to determine which tokens to preserve, reflecting both cross-modal replaceability and query relevance.

Across four audio–visual understanding benchmarks and multiple OmniLLMs, NEOPrune generally achieves higher accuracy than the evaluated training-free token compression baselines at matched token budgets, while maintaining performance close to full-token inference with only 35–45% of audio–visual tokens. Figure 1 shows how Pre-LLM pruning retains salient tokens and summarizes the accuracy–efficiency trade-off. Compared with full-token inference, NEOPrune accelerates LLM prefill by up to $3.25\times$ and reduces peak GPU memory usage by up to 33%.

Our main contributions are summarized as follows:

- **Layer-wise update reconstruction.** We extend reconstruction-based token selection to layer-wise orthogonal updates, enabling token replaceability to be assessed through changes introduced at individual layers.
- **Two-stage reconstruction.** We apply this update-level criterion to local within-modality reconstruction before the LLM and to video-update reconstruction conditioned on temporally aligned audio updates inside the LLM, combining reconstruction error with encoder attention and text-query relevance, respectively.

- **Accuracy–efficiency trade-off.** Across four audio–visual benchmarks and multiple OmniLLMs, NEOPrune generally outperforms training-free baselines at matched token budgets, retains near-full-token accuracy with only 35–45% of the tokens, and delivers up to 3.25× faster prefill with 33% lower peak GPU memory than full-token inference.

2 Related Work

To jointly reduce audio and video tokens in OmniLLMs, recent training-free methods leverage audio information to guide multimodal compression. OmniZip compresses audio using encoder attention and cross-modal similarity, and adjusts video compression according to the retained audio within each temporal segment Tao et al. (2026). DASH constructs audio-driven temporal segments and selects audio tokens using boundary, uniqueness, and attention signals, followed by spatiotemporal video compression within each segment Li & Huang (2026).

However, these methods do not directly evaluate collective redundancy, where a token can be explained by a combination of other tokens Ma et al. (2026). They also complete compression in the Pre-LLM stage without incorporating text-query relevance during Inner-LLM processing. We instead evaluate replaceability using reconstruction errors of layer-wise orthogonal updates and combine this signal with text-query-based attention in the Inner-LLM.

3 Observations for Token Selection in OmniLLMs

In this section, we examine token states in OmniLLMs and the updates introduced by the attention and FFN modules. We first compare token states with layer-wise updates and investigate reconstruction-based token selection using the orthogonal component of the updates. We then examine within- and cross-modal reconstruction relationships that can be exploited in the Pre-LLM and Inner-LLM stages.

3.1 Accumulated Token States and Layer-wise Updates

In the Pre-LLM stage, modality-specific encoders process audio and video independently. For each encoder, we compare the cosine similarity between neighboring tokens using the token states at the Transformer block output and the block updates, defined as the difference between the block output and input. A block update represents the overall change introduced to the existing token state by the attention and FFN modules within the block.

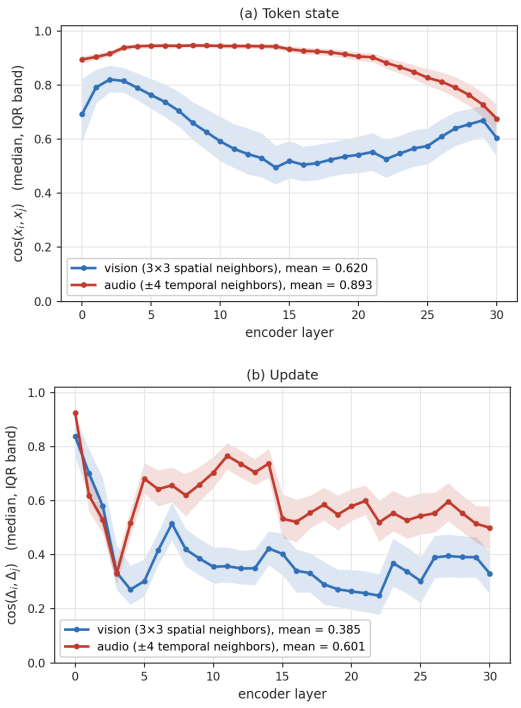

Figure 2: **Neighboring-token similarity in audio and vision encoders.** Cosine similarity of neighboring tokens for (a) token states and (b) block updates. Video uses same-frame 3×3 neighbors and audio uses ± 4 temporal neighbors, excluding the target.

As shown in Figure 2, both encoders exhibit lower similarity in block updates than in the token states of neighboring tokens. In addition, the magnitude of the accumulated token states is larger than that of the updates introduced by individual blocks (Appendix A.1). This indicates that, even when accumulated representations dominate and neighboring tokens have similar representations, the changes introduced by each block can differ across tokens. Therefore, examining layer-wise updates allows us to distinguish how tokens with similar token states undergo different changes at each layer.

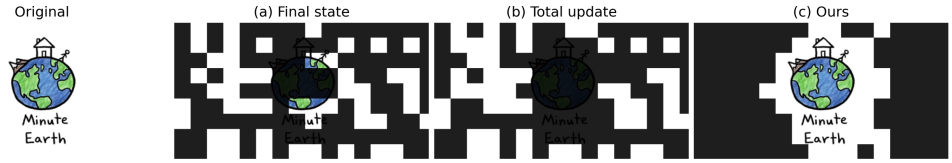

Figure 4: **Pre-LLM token selection with different reconstruction targets.** We compare reconstruction-based selection in the vision encoder using (a) the final token states, (b) the orthogonal component of the total update, and (c) layer-wise orthogonal updates, retaining the same number of tokens per frame based solely on reconstruction scores.

3.2 Orthogonal Updates and Reconstruction-Based Token Selection

To examine the direction of the changes introduced at each layer, we decompose the updates from the attention and FFN modules into components parallel and orthogonal to the normalized module input, referring to the latter as the *orthogonal update*. In the audio encoder, the average orthogonal energy ratio exceeds 88% for both the attention and FFN modules, as shown in Figure 3, and the vision encoder exhibits similarly high ratios. Moreover, in both the audio and vision encoders, reducing the orthogonal component using the computed parallel-component norm as the reference yields a larger mean accuracy drop than removing the parallel component. These observations motivate focusing on orthogonal updates for token selection. The detailed decomposition, vision-encoder results, and component-intervention analyses are provided in Appendix A.2.

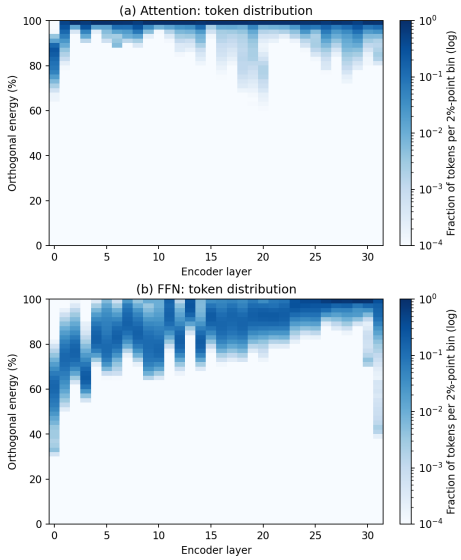

Figure 3: **Orthogonal update composition in the audio encoder.** Orthogonal energy ratios of the updates from the attention and FFN modules. The orthogonal component accounts for more than 88% of the update energy on average.

However, the magnitude of an orthogonal update alone does not indicate whether that change can be represented by a combination of neighboring tokens. To evaluate this, we adopt the reconstruction-based notion of replaceability (Ma et al., 2026) and reconstruct the orthogonal update of each target token using a linear combination of the orthogonal updates of neighboring tokens. The reconstruction residual measures the portion of the change that is not explained by the neighboring set. We accumulate the magnitudes of these residuals from the attention and FFN modules across layers and use them to prioritize token retention.

Let h_i^0 and h_i^L denote the initial and final encoder token states, respectively. We compare reconstruction of the final token state h_i^L and the *total update* $\Delta_i^{\text{total}} = h_i^L - h_i^0$ with our layer-wise orthogonal updates. The former two aggregate changes across encoder layers, whereas layer-wise reconstruction evaluates the change introduced at each layer separately. As shown in Figure 4, layer-wise orthogonal update reconstruction better preserves text and object regions under the same token budget, while reconstructing the orthogonal component of the total update also misses some of these regions. Further analysis of these reconstruction targets, including the motivation for layer-wise reconstruction and additional qualitative comparisons, is provided in Appendix A.3.

3.3 Reconstruction Relationships in Pre-LLM and Inner-LLM

In the Pre-LLM stage, audio and video are processed independently, and the cosine similarity between their modality-wise mean token states remains near zero across encoder lay-

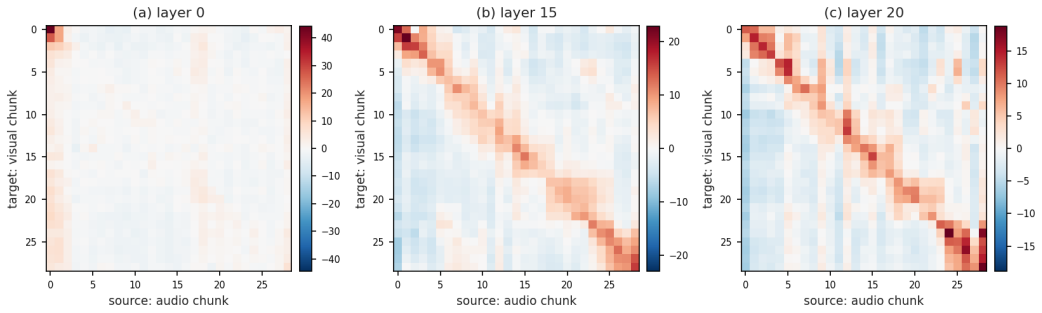

Figure 6: **Reconstruction of video orthogonal updates using audio.** Rows correspond to video chunks and columns to audio chunks. Colors indicate the difference in reconstruction explained ratio relative to the mean of the off-diagonal entries in each row. Diagonal entries correspond to audio and video chunks at the same temporal position.

ers. In contrast, their similarity generally increases with depth in the Inner-LLM stage, as shown in Figure 5; modality-wise token-state distributions are provided in Appendix A.4. This suggests that audio and video representations become increasingly aligned inside the LLM and may share information. We therefore use within-modality spatial or temporal neighbors for reconstruction in the Pre-LLM stage, while examining whether the increasing cross-modal alignment can be exploited for reconstruction in the Inner-LLM stage. To examine this, we reconstruct the attention orthogonal updates of each video chunk using those of each audio chunk and measure the reconstruction explained ratio. For each token, the explained ratio is computed from the reconstruction residual normalized by the magnitude of the original video orthogonal update and is then averaged within each chunk. As shown in Figure 6, the internal LLM layers exhibit relatively higher explained ratios along the diagonal, where audio and video chunks at the same temporal position are paired. This indicates that temporally corresponding audio chunks can explain a larger portion of the changes in video, providing a basis for constructing video reconstruction candidates from the corresponding audio chunks in the Inner-LLM stage.

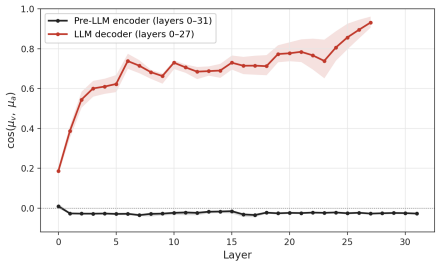

Figure 5: **Audio–video alignment across processing stages.** Cosine similarity between modality-wise mean token states in the Pre-LLM encoders and Inner-LLM layers.

4 METHOD

NEOPrune extracts the component of each Transformer module update that is orthogonal to the module input and evaluates token replaceability based on the reconstruction error obtained by reconstructing the orthogonal update using those of other tokens. In the **Pre-LLM stage**, the orthogonal attention and FFN updates of each modality are reconstructed using neighboring tokens from the same modality, and the reconstruction error is combined with the encoder attention score for token selection. In the **Inner-LLM stage**, the orthogonal attention update of each video token is reconstructed using audio tokens from the same temporal segment, and the reconstruction error is combined with the text-query-based attention score to determine which video tokens are retained.

4.1 ORTHOGONAL UPDATES AND RECONSTRUCTION ERROR

For token i, let h_i denote its token state and $x_i = \mathrm{Norm}(h_i) \in \mathbb{R}^d$ denote the normalized input to the corresponding attention or FFN module. We use the model's original RMSNorm (Zhang &

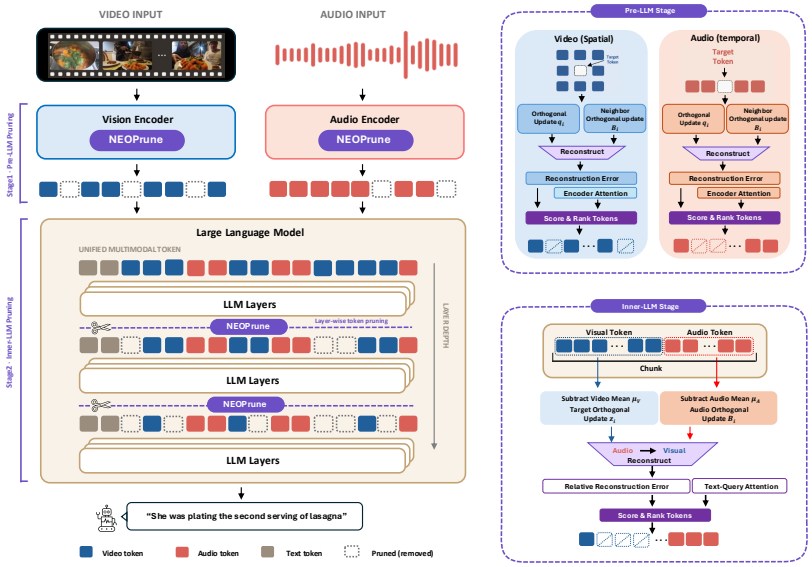

Figure 7: **Overview of NEOPrune.** NEOPrune performs two-stage training-free token pruning using orthogonal updates and attention-based importance, with within-modality reconstruction in Pre-LLM and cross-modal reconstruction in Inner-LLM.

Sennrich, 2019) or LayerNorm (Ba et al., 2016) for Norm, and denote by $\Delta_i \in \mathbb{R}^d$ the output of the attention or FFN module before it is added to the residual stream. For notational simplicity, we omit the layer and module indices, as well as a small constant used for numerical stability. To characterize the change orthogonal to the input direction and its magnitude relative to the input, we define

$$\Delta_i^{\perp} = \Delta_i - \frac{x_i^{\top}\Delta_i}{\|x_i\|_2^2}x_i, \qquad q_i = \frac{\Delta_i^{\perp}}{\|x_i\|_2}. \tag{1}$$

The vector q_i captures the direction of the update orthogonal to the module input and its magnitude relative to the input magnitude.

To evaluate how well a set of candidate tokens can jointly explain a target update, we reconstruct a target r_i using the candidate update matrix B_i, whose columns correspond to the candidate tokens, by solving the following ridge regression:

$$\begin{aligned} c_i^* &= \arg\min_c \left\{ \|r_i - B_i c\|_2^2 + \lambda_i \|c\|_2^2 \right\}, \\ e_i &= \|r_i - B_i c_i^*\|_2. \end{aligned} \tag{2}$$

Here, $\lambda_i > 0$ is the regularization coefficient. The reconstruction error e_i represents the magnitude of the target update that is not explained by the given candidate set. In the Pre-LLM stage, we use $r_i = q_i$, whereas in the Inner-LLM stage, we use a modality-wise mean-centered update as the reconstruction target. Detailed computation procedures and settings are provided in the Appendix A.5.

4.2 Pre-LLM Pruning with Local Reconstruction

In the Pre-LLM stage, we evaluate the replaceability of token updates using neighboring tokens within each modality encoder. To account for the changes introduced by both attention and FFN modules, we reconstruct their orthogonal updates separately. For video, we use spatial neighbors within the same frame, while for audio, we use temporal neighbors as the candidate set $\mathcal{N}_i$.

Using the ridge regression defined in Section 4.1, we set $r_i = q_i$ and construct the candidate matrix from the orthogonal updates of neighboring tokens:

$$r_i = q_i, \qquad B_i = [q_j : j \in \mathcal{N}_i], \qquad e_i = \|q_i - B_i c_i^*\|_2. \tag{3}$$

We collect and sum the reconstruction errors from the attention and FFN modules over the selected encoder layers, and aggregate them at the level of LLM input tokens to obtain the reconstruction score s_i. A larger s_i indicates that a larger portion of the token update remains unexplained by neighboring-token updates. To combine the reconstruction score s_i with the encoder attention score a_i, which may have different scales, we apply rank normalization within each modality and take a weighted sum:

$$S_i = wR(a_i) + (1 - w)R(s_i). \tag{4}$$

Here, R normalizes ranks within the same modality to $[0, 1]$, assigning larger values to larger original scores, and $w \in [0, 1]$ controls the contribution of the attention score. The reconstruction signals are collected and accumulated over the selected encoder layers without removing tokens during encoder processing. After the modality encoders have finished processing the inputs, we perform a single pruning step immediately before the LLM, retaining the tokens with the highest S_i within each temporal chunk according to its modality-specific token quota.

4.3 Inner-LLM Pruning with Audio-Guided Reconstruction

In the Inner-LLM stage, we exploit the cross-modal relationships formed through attention. Based on the original temporal positions of the tokens retained after the Pre-LLM stage, we use the audio tokens remaining in the same temporal chunk as each video token as reconstruction candidates for the attention orthogonal update.

To remove the modality-specific mean component, we apply mean-centering (Mu & Viswanath, 2018; An et al., 2025). Let μ_m denote the mean of q_i over all remaining tokens of modality m at the current layer. We define the mean-centered update of token i as

$$z_i = q_i - \mu_m. \tag{5}$$

Let $\mathcal{A}_i$ denote the set of audio tokens in the same temporal chunk as video token i. Using the ridge regression defined in Section 4.1, we set $r_i = z_i$ and reconstruct the mean-centered video update using the z_j of the candidate audio tokens. To account for the magnitude of the target update, we define the relative reconstruction error as

$$r_i = z_i, \qquad B_i = [z_j : j \in \mathcal{A}_i], \qquad \rho_i = \frac{\|z_i - B_i c_i^*\|_2}{\|z_i\|_2}. \tag{6}$$

A smaller relative reconstruction error ρ_i indicates that the mean-centered variation of the video token can be better reconstructed by audio tokens within the same chunk. Starting from the first selected LLM layer, we accumulate ρ_i up to the current layer to obtain the reconstruction score s_i^x, where a larger value indicates that more of the video-token variation remains unexplained by the audio tokens.

To additionally incorporate relevance to the question, we compute the attention score a_i by averaging, across attention heads, the attention assigned to video token i by the last prompt token at the current layer. We then combine it with the reconstruction score as

$$S_i = w_2 R(a_i) + (1 - w_2)R(s_i^x). \tag{7}$$

Here, R denotes rank normalization over the currently remaining video tokens, and $w_2 \in [0, 1]$ controls the contribution of the attention score. At each selected pruning layer, we update the accumulated reconstruction score, recompute S_i, and progressively remove video tokens with lower scores while retaining the audio tokens. Unlike the single pruning step in the Pre-LLM stage, this progressively selects video tokens across multiple LLM layers based on both the variation unexplained by temporally corresponding audio tokens and its relevance to the question. Detailed implementation settings for both the Pre-LLM and Inner-LLM stages are provided in Appendix A.6.

5 Experiments

5.1 Experimental Setup

Experimental setup. We evaluate NEOPrune on four audio–visual understanding benchmarks: WorldSense (Hong et al., 2026), AVUT (Yang et al., 2025), VideoMME (Fu et al., 2025), and

Table 1: Comparison of token pruning methods on AVUTBench, VideoMME, and ShortVid-Bench with Qwen2.5-Omni. The final Avg. denotes average performance retention relative to Full Tokens, following OmniZip (Tao et al., 2026).

Method	Settings	AVUTBench							VideoMME	ShortVid-Bench	Avg.
	Retained Ratio	EL	OR	OM	IE	CC	CM	Avg.	wo	Avg. Score	
Qwen2.5-Omni-7B											
Full Tokens	100%	38.2	67.8	59.6	85.6	44.1	66.7	64.5	66.0	70.5	100%
Random	55%	38.2	64.9	55.6	80.1	34.7	65.0	61.0	65.4	68.3	96.9%
FastV	50%	34.1	64.3	57.1	77.6	36.4	56.4	58.4	–	68.0	94.3%
DyCoke (V&A)	50%	**38.8**	67.2	58.2	81.9	39.0	62.4	62.0	65.5	68.5	97.5%
OmniZip	45%	38.4	67.2	56.9	**85.3**	42.4	66.0	63.0	66.3	69.9	99.1%
DASH	45%	36.5	**68.2**	56.6	84.4	40.7	**67.6**	63.5	**66.4**	69.9	99.4%
NEOPrune (Ours)	45%	38.2	**68.2**	**59.2**	85.0	**45.8**	65.5	**64.2**	**66.4**	**70.3**	**99.9%**
FastV	35%	24.1	60.7	54.3	81.6	40.7	58.3	57.8	–	67.9	93.8%
DyCoke (V&A)	35%	32.9	62.1	54.9	74.5	39.0	58.3	57.4	65.2	68.0	94.7%
OmniZip	35%	34.1	**67.5**	54.6	83.7	42.4	61.2	61.0	66.1	69.0	97.6%
DASH	35%	32.0	62.1	**58.9**	**85.5**	40.4	64.2	61.5	**66.7**	69.4	98.3%
NEOPrune (Ours)	35%	**37.1**	67.2	57.9	84.7	**44.1**	**64.7**	**63.3**	66.5	**69.7**	**99.3%**
Qwen2.5-Omni-3B											
Full Tokens	100%	32.9	65.3	58.4	85.0	44.1	62.6	62.2	62.6	69.4	100%
Random	55%	31.7	59.2	55.4	77.3	**44.9**	**62.1**	58.7	61.1	67.9	96.6%
FastV	50%	27.1	57.0	56.3	80.5	42.3	60.1	55.9	–	68.0	95.7%
DyCoke (V&A)	50%	31.9	64.3	57.3	82.2	40.7	61.3	60.7	61.6	67.4	97.7%
OmniZip	45%	32.4	65.0	**57.7**	84.9	41.5	61.4	**61.3**	62.8	68.5	99.2%
DASH	45%	**33.5**	63.3	57.1	85.0	44.1	61.2	61.2	**63.0**	68.6	99.3%
NEOPrune (Ours)	45%	31.8	**65.3**	56.4	**85.6**	44.1	60.9	**61.3**	**63.0**	**69.6**	**99.8%**
FastV	35%	24.2	60.8	54.3	81.6	40.7	58.3	57.7	–	67.7	96.9%
DyCoke (V&A)	35%	**32.9**	62.1	54.9	74.5	38.9	58.3	57.4	61.0	67.5	95.7%
OmniZip	35%	28.8	63.1	**58.2**	84.0	**42.4**	**60.4**	60.1	62.7	68.0	98.3%
DASH	35%	**32.9**	61.7	55.6	83.4	**42.4**	60.0	59.9	62.6	68.3	98.2%
NEOPrune (Ours)	35%	31.2	**65.0**	56.4	**85.6**	**42.4**	59.7	**60.8**	**63.7**	**69.2**	**99.7%**

ShortVid (Ge et al., 2025), using Qwen2.5-Omni-7B, Qwen2.5-Omni-3B, and Qwen3-Omni-30B-A3B. For VideoMME, we follow the evaluation protocol implemented in LMMs-Eval (Zhang et al., 2025a). All experiments are conducted using NVIDIA RTX PRO 6000 Blackwell GPUs.

5.2 Main Results

Performance under matched token budgets. Tables 1 and 3 compare NEOPrune with existing token-compression methods at matched retained-token ratios. Across the evaluated model sizes and benchmarks, NEOPrune preserves strong performance while substantially reducing the number of audio–visual tokens. For example, with Qwen2.5-Omni-7B at a 45% retained-token ratio, NEOPrune improves over DASH from 63.5 to **64.2** on AVUT and from 69.9 to **70.3** on ShortVid. On WorldSense, NEOPrune achieves **46.7** and **46.5** at 45% and 35% retention, respectively. These results exceed the corresponding OmniZip scores of 45.9 and 45.3 and the DASH scores of 45.9 and 45.6. Overall, NEOPrune generally achieves higher accuracy than the evaluated pruning baselines at matched token budgets while remaining close to full-token performance. Further analyses of correct-answer log-probability and results on Qwen3-Omni are provided in Appendices A.7 and A.8, respectively.

Table 2: **WorldSense results on Qwen2.5-Omni-7B.** Efficiency is measured on the longest sample; accuracy is reported over the full benchmark.

Method	Retained	GPU Mem. ↓ (GiB)	LLM prefill ↓ (ms)	Acc. ↑ (%)	Latency ↓ (s)
Full (BASE)	100%	43.80	5,154	46.8	8.31
FastV	50%	33.38	2,743	44.3	5.69
DyCoke (V&A)	50%	32.82	2,036	44.6	4.96
OmniZip	35%	29.34	1,267	45.3	4.40
DASH	35%	34.60	1,306	45.55	4.56
NEOPrune (ours)	35%	29.41	1,584	**46.50**	5.02

Efficiency–performance trade-off. Table 2 reports the efficiency of NEOPrune on WorldSense with Qwen2.5-Omni-7B, evaluated using the longest sample. At a 35% retained-token ratio, NEOPrune achieves 46.50% accuracy, compared with 45.30% for OmniZip and 45.55% for DASH, while remaining close to the full-token accuracy of 46.80%. Its memory usage is comparable to OmniZip, and its prefill time and end-to-end latency remain substantially lower than those of the full-token

Table 3: Comparison on WorldSense with Qwen2.5-Omni-7B and Qwen2.5-Omni-3B. Results for Full Tokens and competing baselines are reported from OmniZip, except for DASH, whose results are obtained using the DASH baseline, while NEOPrune results are obtained from our experiments. Best and second-best results under the same retained ratio are shown in **bold** and underlined, respectively.

Method	Retained Ratio	Tech & Science	Culture & Politics	Daily Life	Film & TV	Performance	Games	Sports	Music	Avg.
Qwen2.5-Omni-7B										
Full Tokens	100%	52.4	50.1	48.5	44.6	43.8	41.6	41.6	47.3	46.8
Random	55%	47.1	47.0	44.4	41.2	40.0	40.1	40.1	46.3	43.6
FastV	50%	48.8	47.4	44.2	44.1	41.2	38.3	40.0	46.6	44.3
DyCoke (V&A)	50%	48.4	49.9	46.7	41.4	39.9	40.8	40.2	46.5	44.6
OmniZip	45%	49.4	51.1	45.6	43.9	40.1	40.8	41.9	46.7	45.9
DASH	45%	49.8	50.5	45.9	42.2	43.8	38.6	**45.8**	47.0	45.9
NEOPrune (Ours)	45%	**50.4**	**51.5**	**47.1**	**44.3**	**44.6**	**42.9**	43.7	**47.3**	**46.7**
OmniZip	35%	48.3	49.5	**47.6**	42.5	40.1	40.2	42.3	46.3	45.3
DASH	35%	50.2	**50.8**	45.9	43.5	41.9	38.2	43.5	46.1	45.6
NEOPrune (Ours)	35%	**50.8**	49.8	46.4	**44.3**	**43.4**	**41.2**	**45.6**	**47.0**	**46.5**
OmniZip	30%	50.2	51.1	**47.1**	42.2	**43.1**	39.9	41.2	46.3	45.6
DASH	30%	50.2	**52.4**	46.0	42.2	39.7	36.9	43.7	46.3	45.4
NEOPrune (Ours)	30%	**50.6**	49.8	45.7	**43.8**	**43.1**	**42.5**	**44.2**	**47.8**	**46.2**
Qwen2.5-Omni-3B										
Full Tokens	100%	51.5	50.8	45.0	45.4	43.8	42.5	44.2	46.1	46.4
Random	55%	48.2	46.3	40.7	41.4	38.6	40.0	41.8	43.4	42.8
FastV	50%	50.0	50.5	44.1	43.0	40.5	41.6	41.8	42.1	44.4
DyCoke (V&A)	50%	48.1	48.5	42.3	43.3	39.7	43.4	42.1	43.0	44.0
OmniZip	45%	50.1	50.5	43.9	**45.6**	40.5	40.8	43.7	43.1	45.2
DASH	45%	50.4	50.5	45.6	**45.6**	41.6	40.8	45.1	43.8	45.8
NEOPrune (Ours)	45%	**51.0**	**50.8**	**46.0**	45.1	**43.1**	**42.1**	**45.3**	**45.8**	**46.5**

model. Overall, NEOPrune provides a favorable accuracy–efficiency trade-off by preserving higher task performance with comparable resource reduction. Dataset-level average results and results for the 3B model are provided in Appendix A.9.

Ablation study. Table 4 compares alternative reconstruction signals for Pre-LLM token selection under the same target token budget. We evaluate the final token state, the total orthogonal update, and layer-wise orthogonal updates from attention and FFN. We also include a variant that uses only encoder attention for Pre-LLM token ranking. Across these variants, the Inner-LLM configuration and the chunk-budget allocation rule are kept fixed. Layer-wise reconstruction using both attention and FFN orthogonal updates achieves the highest accuracy among the reconstruction variants. In contrast, using either module alone, reconstructing the final token state, or reconstructing the total orthogonal update yields lower accuracy.

Table 4: **Pre-LLM reconstruction ablation on AVUT.** Results use Qwen2.5-Omni-7B with 35% token retention.

Reconstruction signal	Acc.	Δ vs. Ours
Final token state (L31)	61.30	-1.96
Encoder Attention only	62.46	-0.80
Total orthogonal update	62.75	-0.51
Layer-wise orthogonal update (Attn.)	63.03	-0.23
Layer-wise orthogonal update (FFN)	62.40	-0.86
Layer-wise orthogonal update (Attn.+FFN)	**63.26**	–

6 Conclusion

We introduced **NEOPrune**, a training-free two-stage token pruning method for OmniLLMs based on the reconstruction of layer-wise orthogonal updates. NEOPrune exploits within-modality redundancy in the Pre-LLM stage and cross-modal relationships in the Inner-LLM stage to identify replaceable audio–visual tokens. Across multiple OmniLLMs and audio–visual benchmarks, NEOPrune consistently preserves strong task performance under reduced token budgets while improving inference efficiency.

AI USE STATEMENT

Generative AI tools were employed to enhance the manuscript's grammatical accuracy, clarity, and readability. Their use focused on polishing sentence structure and word choice and strengthening transitions across sentences and paragraphs.

REFERENCES

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9392–9401. IEEE, 2025.

Na Min An, Eunki Kim, James Thorne, and Hyunjung Shim. I0t: Embedding standardization method towards zero modality gap. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27182–27199, 2025.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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, 2025. URL https://arxiv.org/abs/2502.13923.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JroZRaRw7Eu.

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. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2024a.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24185–24198, 2024b.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24108–24118. IEEE, 2025.

Yuying Ge, Yixiao Ge, Chen Li, Teng Wang, Junfu Pu, Yizhuo Li, Lu Qiu, Jin Ma, Lisheng Duan, Xinyu Zuo, et al. Arc-hunyuan-video-7b: Structured video comprehension of real-world shorts. *arXiv preprint arXiv:2507.20939*, 2025.

Jack Hong, Shilin Yan, Jiayin Cai, Xiaolong Jiang, Yao Hu, and Weidi Xie. Worldsense: Evaluating real-world omnimodal understanding for multimodal llms. In *International Conference on Learning Representations*, volume 2026, pp. 52423–52443, 2026.

Ao Li, Yuxiang Duan, Jinghui Zhang, Congbo Ma, Yutong Xie, Gustavo Carneiro, Mohammad Yaqub, and Hu Wang. Transprune: Token transition pruning for efficient large vision-language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 39529–39538, June 2026.

Bingzhou Li and Tao Huang. Dash: Dynamic audio-driven semantic chunking for efficient omnimodal token compression. In *European Conference on Computer Vision*. Springer, 2026.

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 *International conference on machine learning*, pp. 19730–19742. PmLR, 2023.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Qiankun Ma, Ziyao Zhang, Haofei Wang, Zhen Song, Jie Chen, and Hairong Zheng. Apet: Approximation-error guided token compression for efficient vlms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26306–26316, 2026.

Jiaqi Mu and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HkuGJ3kCb.

Fangxun Shu, Lei Zhang, Hao Jiang, and Cihang Xie. Audio-visual llm for video understanding. In *2025 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 4305–4314. IEEE, 2025.

Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. Dycoke: Dynamic compression of tokens for fast video large language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18992–19001. IEEE, 2025.

Keda Tao, Kele Shao, Bohan Yu, Weiqiang Wang, Jian Liu, and Huan Wang. Omnizip: Audio-guided dynamic token compression for fast omnimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17682–17692, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. Qwen2.5-omni technical report, 2025a. URL https://arxiv.org/abs/2503.20215.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, et al. Qwen3-omni technical report. *arXiv preprint arXiv:2509.17765*, 2025b.

Yudong Yang, Jimin Zhuang, Guangzhi Sun, Changli Tang, Yixuan Li, Peihan Li, Yifan Jiang, Wei Li, Zejun Ma, and Chao Zhang. Audio-centric video understanding benchmark without text shortcut. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 6580–6598, 2025.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. LMMs-eval: Reality check on the evaluation of large multimodal models. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 881–916, Albuquerque, New Mexico, April 2025a. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.51. URL https://aclanthology.org/2025.findings-naacl.51/.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. SparseVLM: Visual token sparsification for efficient vision-language model inference. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=80faIPZ67S.

A Appendix

A.1 Comparison of Token States and Updates

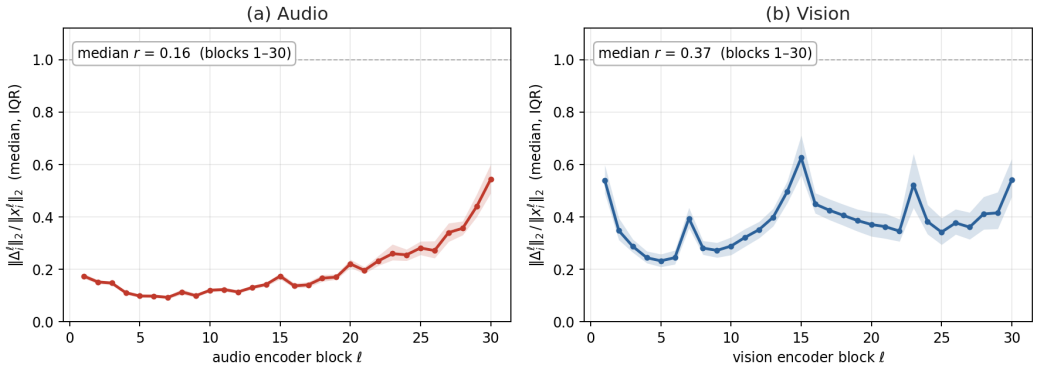

Figure 8: **Relative magnitude of block updates in audio and vision encoders.** The block-update magnitude is normalized by the magnitude of the corresponding input token state.

Figure 2 compares the cosine similarity of the same neighboring token pairs across encoder layers 0–30. For video, we use spatial neighbors within a 3×3 region of the same frame, excluding the target token, while for audio, we use the four preceding and four following temporal neighbors. Averaged across layers, the cosine similarity is 0.620 for token states and 0.385 for block updates in the vision encoder, and 0.893 and 0.601, respectively, in the audio encoder.

Figure 8 shows the magnitude of each block update relative to that of the corresponding input token state over encoder blocks 1–30. In both the audio and vision encoders, token states maintain larger magnitudes than the updates introduced by individual blocks. When the ratio is summarized by first taking the median over tokens for each video and layer and then taking the median across videos and layers, the resulting values are 0.16 for audio and 0.37 for video.

A.2 Validation of Orthogonal Updates and the Reconstruction Criterion

Detailed decomposition. For each attention or FFN module, let $h_i \in \mathbb{R}^d$ denote the token state of token i, and let $x_i = \mathrm{Norm}(h_i)$ denote the normalized module input. We denote the update introduced by the module as $\Delta_i \in \mathbb{R}^d$. We decompose Δ_i into components parallel and orthogonal to the direction of x_i.

By the orthogonal projection theorem, the component of Δ_i in the one-dimensional subspace spanned by x_i is

$$\Delta_i^{\parallel} = \mathrm{proj}_{x_i}(\Delta_i) = \frac{x_i^\top \Delta_i}{\|x_i\|_2^2} x_i. \tag{8}$$

The remaining component is defined as

$$\Delta_i^{\perp} = \Delta_i - \Delta_i^{\parallel}. \tag{9}$$

By construction,

$$x_i^\top \Delta_i^{\perp} = x_i^\top \Delta_i - \frac{x_i^\top \Delta_i}{\|x_i\|_2^2} x_i^\top x_i = 0, \tag{10}$$

and therefore $\Delta_i^{\perp}$ is orthogonal to x_i. Since $\Delta_i^{\parallel}$ is parallel to x_i, the two components are also mutually orthogonal:

$$\left(\Delta_i^{\parallel}\right)^\top \Delta_i^{\perp} = 0. \tag{11}$$

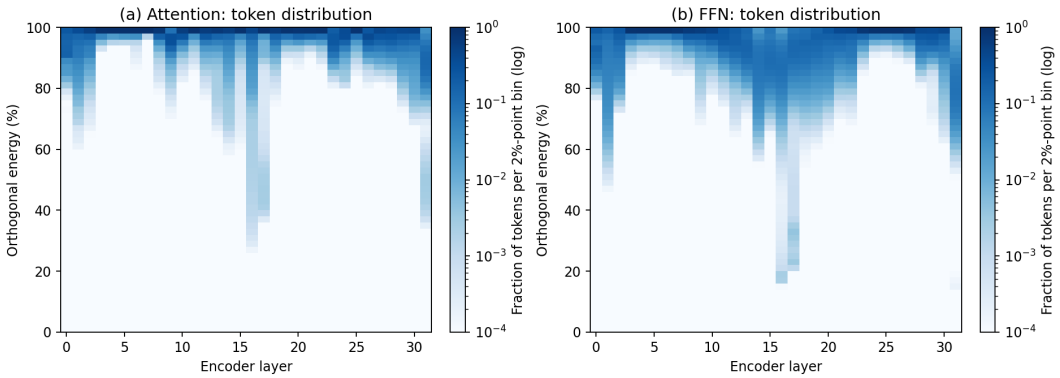

Figure 9: **Orthogonal energy distribution in the vision encoder.** Token distributions of the orthogonal energy ratio $\|\Delta^{\perp}\|_2^2/\|\Delta\|_2^2$ across encoder layers for (a) attention and (b) FFN. Colors indicate the fraction of tokens within each 2-percentage-point bin on a logarithmic scale.

Thus, the update can be written as the orthogonal decomposition

$$\Delta_i = \Delta_i^{\parallel} + \Delta_i^{\perp}, \tag{12}$$

and the Pythagorean identity gives

$$\|\Delta_i\|_2^2 = \|\Delta_i^{\parallel}\|_2^2 + \|\Delta_i^{\perp}\|_2^2. \tag{13}$$

To compare orthogonal changes across tokens relative to their input magnitudes, we further define the normalized orthogonal update as

$$q_i = \frac{\Delta_i^{\perp}}{\|x_i\|_2}. \tag{14}$$

Since this normalization only rescales $\Delta_i^{\perp}$, q_i preserves its direction and remains orthogonal to x_i.

Vision-encoder results. We repeat the analysis on the vision encoder by computing the orthogonal energy ratio $\eta_i = \|\Delta_i^{\perp}\|_2^2/\|\Delta_i\|_2^2$ for each token across encoder layers. As shown in Figure 9, the distributions of both attention and FFN updates are concentrated toward high orthogonal energy ratios. Averaged over tokens and layers, the orthogonal component accounts for 97.3% of the attention update energy and 94.6% of the FFN update energy. This is consistent with the audio encoder, where the corresponding ratios are 98.7% and 88.2%, respectively, showing that most of the update energy in both modality encoders lies in the component orthogonal to the normalized module input.

Component-scaling analysis. To compare the effects of the parallel and orthogonal update components on the final representation, we selectively scale one component while keeping the other unchanged. When scaling the parallel component, we use

$$\widetilde{\Delta}_i^{(\parallel)}(\alpha) = \alpha\Delta_i^{\parallel} + \Delta_i^{\perp}, \tag{15}$$

whereas scaling the orthogonal component gives

$$\widetilde{\Delta}_i^{(\perp)}(\alpha) = \Delta_i^{\parallel} + \alpha\Delta_i^{\perp}, \qquad \alpha \in [0, 1]. \tag{16}$$

Here, $\alpha = 1$ corresponds to the original update, while $\alpha = 0$ completely removes the selected component. Let z denote the final representation produced by the original model and z' the representation after component scaling. We measure the resulting directional change as

$$D(z, z') = 1 - \cos(z, z'). \tag{17}$$

Figures 10 and 11 show the results for the audio and vision encoders, respectively. In both modalities, reducing the orthogonal component produces a larger directional change in the final representation than reducing the parallel component by the same factor. This pattern is observed for both

attention and FFN modules, and the difference becomes more pronounced as the retained strength α decreases.

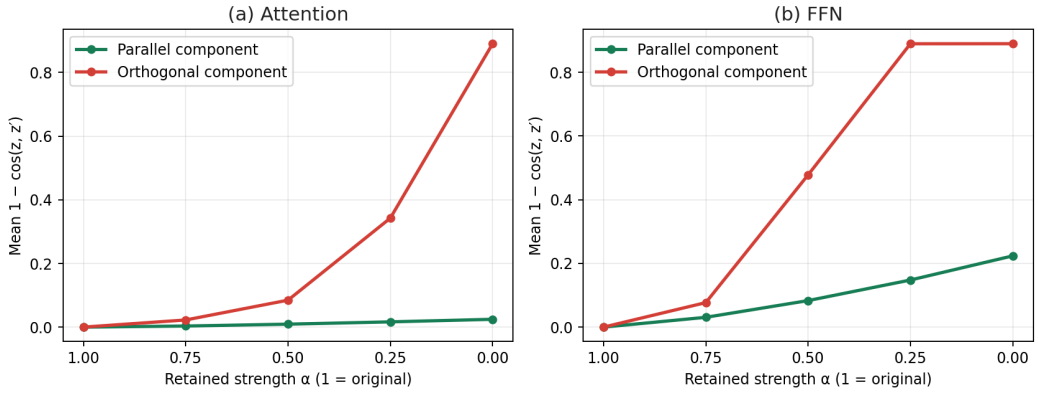

Figure 10: **Final-representation change under update-component scaling in the audio encoder.** For (a) attention and (b) FFN, either the parallel or orthogonal component is scaled by retained strength α. $\alpha = 1$ corresponds to the original update and $\alpha = 0$ to complete removal of the selected component. The directional change is measured as $1 - \cos(z, z')$.

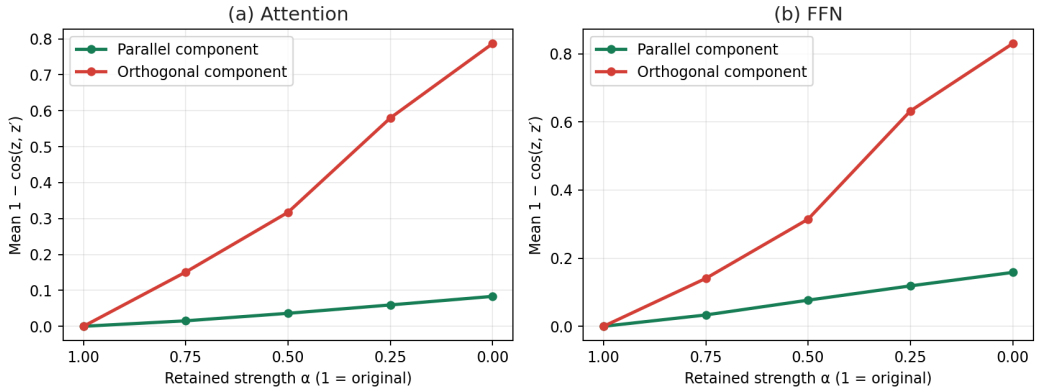

Figure 11: **Final-representation change under update-component scaling in the vision encoder.** The experimental setting follows Figure 10.

Parallel-norm-based component intervention. Because the orthogonal component generally accounts for most of the update magnitude, applying the same scaling factor α to both components removes a larger absolute amount from the orthogonal component. To examine whether the observed difference persists under an alternative reduction rule, we conduct an additional intervention in which the norm of the computed parallel component is used as the reference for reducing the orthogonal component.

Both the vision and audio encoders contain 32 blocks, indexed from 0 to 31. At every block, we decompose the module update Δ_i, produced by either the attention or FFN module for input token x_i, as

$$\Delta_i = \Delta_i^{\parallel} + \Delta_i^{\perp}. \tag{18}$$

For the parallel-removal intervention, we remove the entire parallel component:

$$h_i' = h_i + \Delta_i^{\perp}. \tag{19}$$

For the orthogonal-reduction intervention, we use $\|\Delta_i^{\parallel}\|_2$ as the reference reduction magnitude while constraining the scaling coefficient to be nonnegative:

$$h_i' = h_i + \Delta_i^{\parallel} + s_i^{\perp}\Delta_i^{\perp}, \qquad s_i^{\perp} = \begin{cases} \max\left(0, 1 - \dfrac{\|\Delta_i^{\parallel}\|_2}{\|\Delta_i^{\perp}\|_2}\right), & \|\Delta_i^{\perp}\|_2 > 0, \\ 0, & \|\Delta_i^{\perp}\|_2 = 0. \end{cases} \tag{20}$$

Thus, the computed parallel-component norm determines the target reduction of the orthogonal component. When the parallel-component norm exceeds the orthogonal-component norm, the orthogonal component is reduced to zero rather than being scaled by a negative coefficient.

After the intervention, we run the model without any additional adaptation and evaluate it on 102 questions selected from AVUT, comprising 17 questions from each of six tasks. Accuracy changes are reported relative to the unmodified Full model, which achieves 50.0% accuracy. The attention and FFN interventions are performed separately. Each Audio or Vision bar in Figure 12 averages the corresponding attention and FFN conditions, while All averages the four conditions formed by audio/vision encoder and attention/FFN module interventions.

As shown in Figure 12, a larger mean accuracy drop is observed for orthogonal-component reduction in all three reported aggregations. For Audio, the observed mean accuracy drops are 2.9 percentage points under parallel-component removal and 5.9 points under orthogonal-component reduction. The corresponding mean accuracy drops are 0.5 versus 1.5 points for Vision and 1.7 versus 3.7 points for All.

Across the three reported aggregations, reducing the orthogonal component using the parallel-component norm as the reference results in a larger observed mean accuracy drop than removing the parallel component. Thus, the observed asymmetry between the two component interventions remains under this alternative reduction rule, providing additional evidence that the orthogonal component is important for preserving task performance.

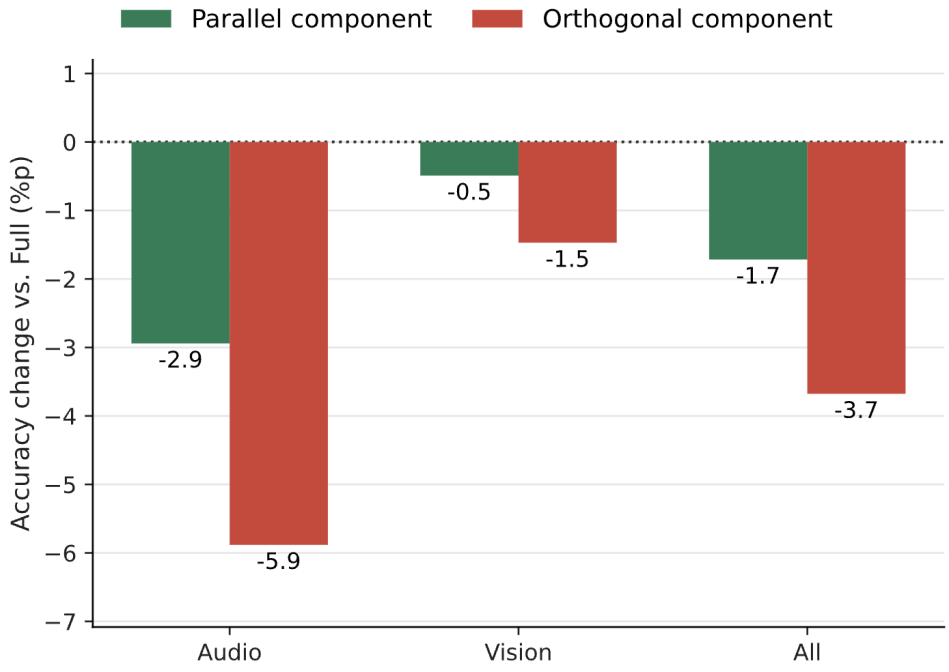

Figure 12: **Accuracy change under parallel-norm-based component intervention.** Accuracy change relative to Full after removing the parallel component or reducing the orthogonal component using the computed parallel-component norm as the reference at each token. Audio and Vision average the attention and FFN interventions, while All averages all four conditions. A larger mean accuracy drop is observed under orthogonal-component reduction in all three aggregations.

A.3 Analysis of Reconstruction-Based Token Selection

We further analyze the reconstruction targets compared in Figure 4. Let h_i^0 and h_i^L denote the initial and final encoder states of token i, respectively. For the *final token state*, we directly reconstruct h_i^L using the final states of neighboring tokens. For the *total orthogonal update*, we first define the accumulated change across the encoder as

$$\Delta_i^{\text{total}} = h_i^L - h_i^0, \tag{21}$$

and compute its component orthogonal to the initial state:

$$q_i^{\text{total}} = \frac{\Delta_i^{\text{total}} - \frac{(h_i^0)^\top \Delta_i^{\text{total}}}{\|h_i^0\|_2^2} h_i^0}{\|h_i^0\|_2}. \tag{22}$$

This represents the change accumulated across the entire encoder as a single update. In contrast, our layer-wise criterion computes the normalized orthogonal update $q_i^{\ell,m}$ separately for each selected layer ℓ and module $m \in \{\text{Attn.}, \text{FFN}\}$, reconstructs each update from neighboring tokens, and accumulates the resulting reconstruction errors. Across these variants, we use the same candidate neighborhoods and ridge-regression procedure and change only the reconstruction target.

Why layer-wise reconstruction? The final token state and total update collapse a sequence of intermediate changes into a single endpoint representation. For intuition, let Δ_i^ℓ denote the net residual update introduced to token i by encoder block ℓ. Then,

$$h_i^L - h_i^0 = \sum_{\ell=1}^{L} \Delta_i^\ell. \tag{23}$$

Updates introduced at different layers can partially cancel in this sum. For example, if $\Delta_i^1 = v$ and $\Delta_i^2 = -v$, the total update becomes zero even though the token undergoes nonzero changes at both layers. Therefore, different layer-wise update trajectories can produce the same endpoint change.

A similar effect can occur for reconstruction residuals. Let

$$\epsilon_i^{\ell,m} = q_i^{\ell,m} - \hat{q}_i^{\ell,m} \tag{24}$$

denote the portion of the orthogonal update that is not reconstructed from neighboring tokens at layer ℓ and module m. Our reconstruction score accumulates these residuals separately:

$$s_i = \sum_{\ell \in \mathcal{L}} \sum_{m \in \{\text{Attn.}, \text{FFN}\}} \|\epsilon_i^{\ell,m}\|_2. \tag{25}$$

In a simplified setting where the residuals are aggregated before their magnitude is measured, the triangle inequality gives

$$\left\| \sum_\ell \epsilon_i^\ell \right\|_2 \le \sum_\ell \|\epsilon_i^\ell\|_2. \tag{26}$$

The inequality becomes strict when residuals point in different directions, and complete cancellation occurs, for example, when $\epsilon_i^1 = v$ and $\epsilon_i^2 = -v$. Thus, evaluating only an aggregated change can mask variations that appear at individual layers.

Moreover, the reference direction for the orthogonal component also changes across layers. Our criterion evaluates each update relative to the corresponding module input $x_i^{\ell,m}$, whereas the total-update variant projects the accumulated change with respect to the initial state. In general,

$$P_{h_i^0}^\perp \left(\sum_\ell \Delta_i^\ell \right) \neq \sum_\ell P_{x_i^\ell}^\perp \left(\Delta_i^\ell \right). \tag{27}$$

Layer-wise reconstruction therefore preserves information about both the intermediate changes and the representation direction at which each change is introduced.

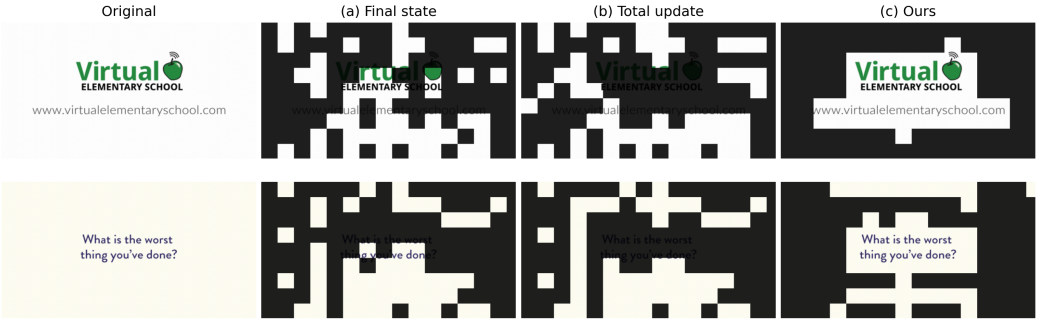

Figure 13: **Additional comparison of reconstruction targets.** We compare token selection using (a) the final token state, (b) the orthogonal component of the total update, and (c) our layer-wise orthogonal updates. All methods are assigned exactly the same token budget for each frame. White regions indicate retained tokens and black regions indicate removed tokens.

Qualitative comparison. Figure 13 provides additional examples comparing the three reconstruction targets. For a controlled comparison, all methods are assigned exactly the same token budget for each frame, with only the reconstruction target changed. Layer-wise orthogonal update reconstruction retains more tokens around text and localized object regions in these examples, whereas final-state and total-update reconstruction remove portions of these regions. This behavior is consistent with the layer-wise criterion retaining intermediate changes that can be obscured when updates are aggregated into a single endpoint representation.

A.4 Audio–Visual Alignment Across Processing Stages

Figure 14 visualizes the token-state distributions of audio and video across the Pre-LLM and Inner-LLM stages. In the Pre-LLM stage, audio and video are processed independently by their respective encoders and do not directly interact with each other. Accordingly, their modality-wise mean token states remain nearly orthogonal throughout the encoder layers, with cosine similarities close to zero.

After entering the LLM, audio and video tokens are processed jointly and can repeatedly interact through attention. Their representations are still relatively separated near the LLM input, but become increasingly aligned from intermediate to deeper layers. As shown in Figure 14, the cosine similarity between the modality-wise mean representations increases from 0.20 at the LLM input to 0.74 and 0.87 at layers 15 and 25, respectively. This increasing alignment suggests that audio and video representations may contain overlapping information after repeated cross-modal interaction, motivating the use of temporally corresponding audio tokens to reconstruct video-token updates in the Inner-LLM stage.

A.5 Orthogonal Update and Reconstruction Details

For each attention or FFN module, we obtain x_i from the normalized module input and Δ_i from the module output immediately before it is added to the residual stream. We cast the collected module inputs and outputs to FP32 before computing the orthogonal updates and reconstruction quantities. The normalized orthogonal update is computed as

$$q_i = \frac{\Delta_i - \frac{x_i^\top \Delta_i}{\|x_i\|_2^2+\epsilon} x_i}{\|x_i\|_2 + \epsilon}, \tag{28}$$

where $\epsilon = 10^{-6}$ is used for numerical stability. The model's original normalization layer is used without modification.

Given a reconstruction target r_i and a candidate matrix B_i, we solve the ridge regression

$$c_i^* = \arg\min_c \left(\|r_i - B_i c\|_2^2 + \lambda_i \|c\|_2^2 \right). \tag{29}$$

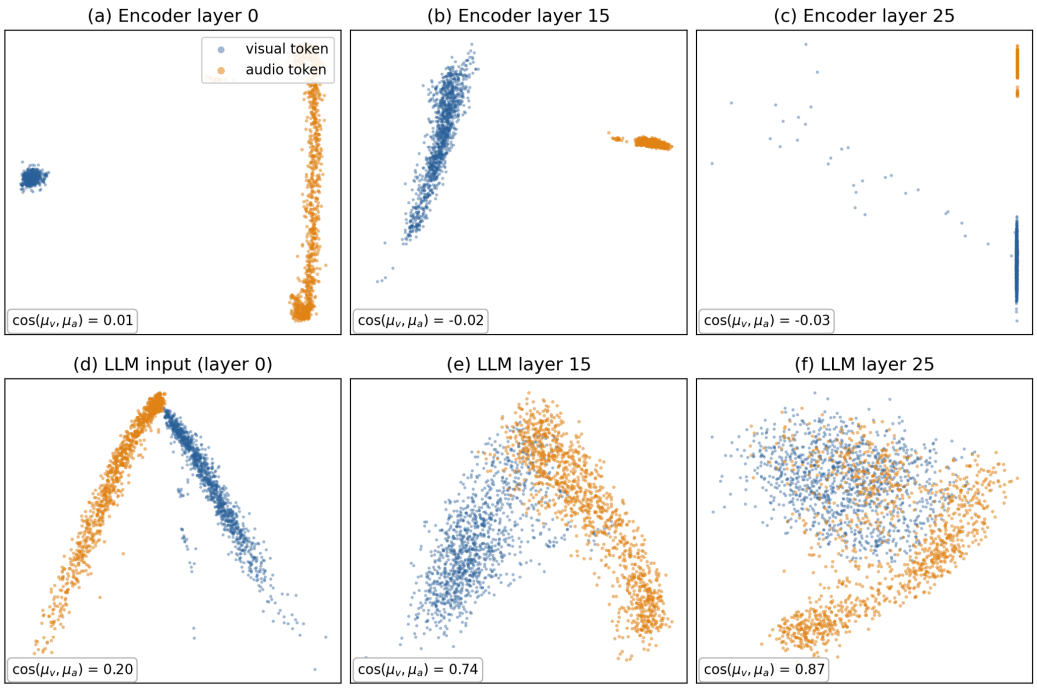

Figure 14: **Audio–visual token-state distributions across processing stages.** Audio and video representations remain separated in the independently processed encoder layers, whereas they become increasingly aligned inside the LLM. The value in each panel denotes the cosine similarity between the modality-wise mean token states.

Let

$$G_i = B_i^\top B_i. \tag{30}$$

The closed-form solution is

$$c_i^* = (G_i + \lambda_i I)^{-1} B_i^\top r_i, \tag{31}$$

and the reconstruction error is

$$e_i = \|r_i - B_i c_i^*\|_2. \tag{32}$$

In implementation, we solve the corresponding regularized linear system without explicitly forming the matrix inverse.

We use a base ridge coefficient $\alpha = 0.01$ in both stages and scale the effective regularization coefficient by the average squared norm of the candidate updates. The numerical floors are stage-specific, as specified below. In the Pre-LLM stage, the candidate neighborhood contains eight slots, including zero padding at spatial or temporal boundaries. We therefore use

$$\lambda_i = \alpha \max\left(\frac{\mathrm{tr}(G_i)}{8}, 10^{-12}\right). \tag{33}$$

In the Inner-LLM stage, let K_i denote the actual number of remaining audio candidates for video token i. The effective regularization is

$$\lambda_i = \max\left(\alpha \frac{\mathrm{tr}(G_i)}{K_i}, 10^{-12}\right). \tag{34}$$

Pre-LLM reconstructs each token's orthogonal update using neighboring-token orthogonal updates within the same modality, whereas Inner-LLM reconstructs mean-centered video orthogonal updates using temporally corresponding mean-centered audio orthogonal updates.

A.6 Implementation Details

Pre-LLM. We accumulate the reconstruction errors from both the attention and FFN modules over encoder layers 3–14. For video tokens, the candidate set consists of the spatial 3×3 neighborhood within the same frame, excluding the target token itself, while for audio tokens, we use the ± 4 temporal neighbors. Because the encoder-token resolution differs from that of the LLM input tokens, we aggregate the reconstruction scores accordingly. For video, we average the scores within each 2×2 merge group, whereas for audio, we pool the encoder-frame scores over the temporal interval corresponding to each LLM input token. This yields the reconstruction score s_i at the LLM-input-token level. We use $w = 0.5$ to combine the reconstruction score with the encoder attention score. No tokens are removed during encoder processing; pruning is performed only once, immediately before the LLM.

Inner-LLM. We apply Inner-LLM pruning at layers 15–25 for Qwen2.5-Omni-7B and layers 19–32 for Qwen2.5-Omni-3B, using only the orthogonal updates from the attention modules. For each video token, we accumulate the relative reconstruction error ρ_i obtained using audio tokens within the same 2-second temporal chunk and combine it with the attention from the last prompt token at the current layer using $w_2 = 0.7$. We set the nominal Stage-2 video-token retention ratio relative to the video tokens retained after Stage 1 to $X = 0.756$. Given n pruning layers, we use a constant per-layer video-token pruning ratio

$$p = 1 - X^{1/n}, \tag{35}$$

so that the progressive schedule targets a cumulative video-token retention ratio of X. Audio tokens are retained throughout the Inner-LLM stage. For temporal chunks containing no audio candidate tokens, we set $\rho_i = 1$.

Temporal Chunks and Token Budgets. Token selection is performed independently within 2-second temporal chunks. In the Pre-LLM stage, we retain approximately 85% of the audio tokens and use a fixed video pruning ratio π_v for each model and benchmark to target the nominal final audio–visual retention ratio after both stages. For both Qwen2.5-Omni models, the Pre-LLM video pruning ratios are:

Benchmark	Final AV 35%	Final AV 45%
AVUT	0.651	0.496
WorldSense	0.651	0.492
ShortVid	0.657	0.501
VideoMME	0.646	0.492

For Qwen3-Omni-30B-A3B on AVUT, we use $\pi_v = 0.6076181$ and $\pi_v = 0.4687267$ for the nominal 35% and 45% settings, respectively. The realized retention ratio of an individual sample may vary with its audio–visual token composition and integer token quotas.

To allocate the resulting video-token budget adaptively across temporal chunks, we compute act_c, the mean reconstruction score of the audio tokens in chunk c, and define

$$\beta_c = 1 + \gamma \left(R(\mathrm{act}_c) - 0.5 \right), \qquad \gamma = 0.3. \tag{36}$$

The chunk-wise video quotas are then adjusted according to β_c and normalized such that the total video-token budget determined by π_v is preserved.

A.7 Analysis of Correct-Answer Log-Probability

We compare NEOPrune, OmniZip, and DASH on 60 AVUT questions using Qwen2.5-Omni-7B at a nominal 35% audio–visual token retention ratio. NEOPrune uses the complete two-stage method, with both Pre-LLM and Inner-LLM pruning enabled. For each question, we measure the log-probability assigned to the correct option token in the full-vocabulary distribution at the first answer-generation step.

Figure 15(a) reports the mean log-probability across the evaluated questions. Panel (b) reports its change relative to the unpruned Full model:

$$\Delta\overline{\log p} = \overline{\log p}_{\mathrm{method}} - \overline{\log p}_{\mathrm{Full}}, \tag{37}$$

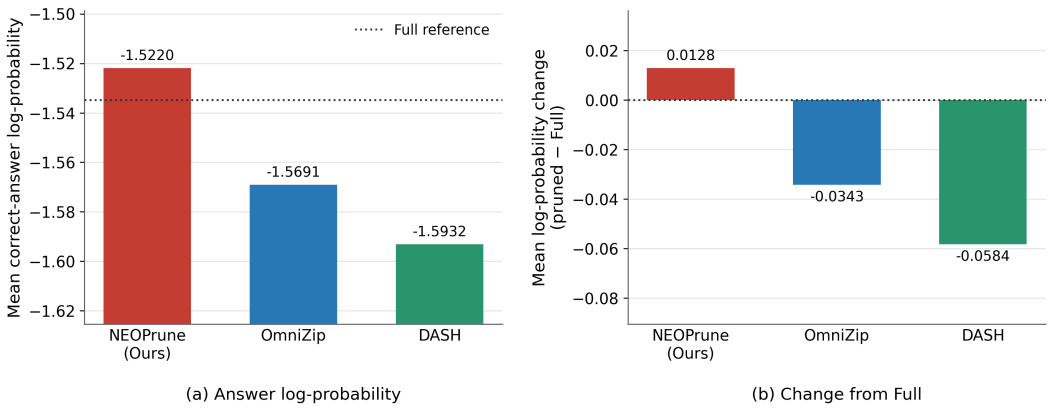

Figure 15: **Correct-answer log-probability under token pruning.** Results on 60 AVUT questions using Qwen2.5-Omni-7B at a nominal 35% audio–visual token retention ratio. NEOPrune applies both Pre-LLM and Inner-LLM pruning. (a) Mean correct-answer token log-probability. (b) Mean change relative to the unpruned Full model; positive values indicate an increase.

Table 5: Comparison of token pruning methods on AVUT with Qwen3-Omni-30B-A3B-Instruct. The Avg. score denotes the QA-weighted overall accuracy over all 1,734 samples. Best and second-best results under the same retained ratio are shown in **bold** and underlined, respectively.

Method	Retained Ratio	EL	OR	OM	IE	CC	CM	Avg.
OmniZip	45%	81.2	**79.4**	67.1	89.6	55.1	73.6	75.7
OmniZip	35%	76.5	77.5	65.6	**89.9**	55.9	73.4	74.6
DASH	45%	80.0	77.8	67.9	89.0	52.5	73.9	75.2
DASH	35%	77.1	77.2	65.3	88.3	51.7	**74.3**	74.2
NEOPrune (Ours)	45%	**82.9**	79.1	**68.6**	**89.9**	**56.8**	**74.3**	**76.5**
NEOPrune (Ours)	35%	**83.5**	**79.1**	**67.3**	89.6	**56.8**	72.7	**75.8**

where the overbar denotes the mean over the 60 questions.

On the 60 AVUT questions, the mean changes in correct-answer log-probability relative to Full are $+0.0128$ for NEOPrune, -0.0343 for OmniZip, and -0.0584 for DASH.

A.8 Results on Qwen3-Omni

Table 5 further evaluates NEOPrune on the larger Qwen3-Omni-30B-A3B-Instruct model. NEOPrune consistently achieves the highest overall accuracy under both retained-token ratios, demonstrating that its effectiveness generalizes beyond Qwen2.5-Omni. At 45% retention, NEOPrune achieves an average accuracy of **76.5**, outperforming OmniZip and DASH by 0.8 and 1.3 percentage points, respectively. It also achieves the best performance in five of the six AVUT categories while remaining competitive on OR. At the more aggressive 35% retention ratio, NEOPrune maintains the highest average accuracy of **75.8**, exceeding OmniZip by 1.2 percentage points and DASH by 1.6 percentage points. It achieves the best performance in four of the six categories while remaining competitive in the remaining categories. These results show that NEOPrune provides consistently strong performance across token budgets and generalizes effectively to a substantially larger Omni model.

A.9 Efficiency

We further evaluate the efficiency of NEOPrune using Qwen2.5-Omni-7B and Qwen2.5-Omni-3B. Tables 6 and 7 report efficiency metrics averaged over the first 30 WorldSense videos, alongside

Table 6: Average efficiency results on the first 30 WorldSense videos using Qwen2.5-Omni-7B. Accuracy is evaluated on the full WorldSense benchmark. The best accuracy among the pruning methods at each retained-token ratio is shown in bold.

Method	Retained	Prefill ↓ (ms)	Speedup ↑	Latency ↓ (s)	Acc. ↑ (%)
Full (BASE)	100%	1,022	1.00×	1.859	46.80
OmniZip	35%	297	3.44×	1.142	45.30
DASH	35%	296	3.45×	1.122	45.55
NEOPrune (ours)	35%	375	2.73×	1.375	**46.50**
OmniZip	45%	385	2.65×	1.259	45.90
DASH	45%	388	2.63×	1.244	45.93
NEOPrune (ours)	45%	505	2.02×	1.509	**46.75**

Table 7: Average efficiency results on the first 30 WorldSense videos using Qwen2.5-Omni-3B at a 45% retained-token ratio. Accuracy is evaluated on the full WorldSense benchmark. The best accuracy among the pruning methods is shown in bold.

Method	Retained	Prefill ↓ (ms)	Speedup ↑	Latency ↓ (s)	Acc. ↑ (%)
Full (BASE)	100%	555	1.00×	1.316	46.47
OmniZip	45%	200	2.78×	1.002	45.20
DASH	45%	203	2.73×	0.991	45.84
NEOPrune (ours)	45%	330	1.68×	1.255	**46.50**

accuracy evaluated on the full benchmark. The 30-video efficiency subset has an average input length of 16,499 tokens. Timing measurements exclude one warm-up run.

Across both model sizes, NEOPrune preserves task accuracy effectively under token reduction. With the 7B model, NEOPrune achieves 46.50% accuracy at a 35% retained-token ratio, remaining within 0.30 percentage points of the full-token result while outperforming OmniZip and DASH. At a 45% ratio, it reaches 46.75%, remaining within 0.05 percentage points of the full-token accuracy and providing the best result among the evaluated pruning methods.

With the 3B model at a 45% retained-token ratio, NEOPrune achieves 46.50% accuracy, slightly exceeding the full-token accuracy of 46.47% and outperforming OmniZip and DASH at the same token budget.

NEOPrune requires some additional computation compared with the lightweight pruning baselines, but its end-to-end latency remains below full-token inference in every reported setting, with prefill speedups ranging from $1.68\times$ to $2.73\times$. Overall, these results demonstrate that NEOPrune operates within a comparable runtime regime while providing a favorable accuracy–efficiency trade-off, particularly when preserving task performance is important.