MODALVERSE: MULTIMODAL AUTOREGRESSION IN PARALLEL INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal autoregressive models commonly serialize different modalities into a single sequence, coupling modality-local causal progression with cross-modal interaction. We introduce **ModalVerse**, a framework that decouples these two operations to enable parallel yet cooperative multimodal inference. ModalVerse organizes modality sequences into parallel lanes that share an autoregressive backbone while maintaining separate causal histories. Within each layer, modality-local updates proceed concurrently, followed by cross-lane attention that exchanges information across modalities. Moreover, to sustain interaction when output lengths differ, continuation tokens keep completed lanes participating in cross-modal communication after visible generation ends. The framework supports two complementary settings: parallel multimodal input processing, where distributed evidence is integrated into a shared prediction, and concurrent multimodal output generation, where heterogeneous outputs advance independently while remaining coordinated. On MUSIC-AVQA, balanced parallel input lanes achieve up to $1.51\times$ faster decoder prefill and reduce peak memory by up to 34.46% while maintaining comparable question-answering accuracy. On adapted A-OKVQA, ModalVerse improves joint text–speech accuracy from 29.07% with independent parallel generation to 38.26%, closely matching 38.68% with concatenated generation. These results show that multimodal autoregression need not serialize modalities into a single sequence: modality-specific computation can proceed in parallel while preserving effective cross-modal interaction.

1 INTRODUCTION

Building on the success of large language models (LLMs) Brown et al. (2020); Touvron et al. (2023), multimodal LLMs have made significant progress in both understanding and generation. Early works extended LLMs with visual inputs, enabling them to process images alongside text and generate textual responses Li et al. (2023); Liu et al. (2023). Subsequent models incorporated additional modalities, such as audio Chu et al. (2023) and video Zhang et al. (2023), further broadening the range of inputs. More recently, this progress has extended to the output side, with omni models (Cheng et al., 2026; Xu et al., 2025; OpenAI, 2024; Li et al., 2026) accepting multiple input modalities while generating multimodal outputs.

Despite this expansion in multimodal capabilities, these models commonly retain a *single-sequence computational paradigm* inherited from autoregressive language modeling. In this paradigm, tokens or encoded features from different modalities are concatenated or interleaved into a shared sequence, where attention performs both within-modality processing and cross-modality interaction. This organization couples two conceptually different dependencies: causal progression within each modality and information exchange across modalities, forcing both to follow the same serialized sequence order. *A single global causal order, however, is not necessary for many multimodal dependencies.* For example, audio and video of the same event, for instance, each follow their own temporal progression while providing complementary information to one another. More generally, multimodal computation contains two distinct operations: **modality-local sequence progression**, which follows each modality's own causal history, and **cross-modal interaction**, which exchanges information across modalities. If these two operations are decoupled, modality-local updates can proceed concurrently, followed by cross-modal interaction that informs subsequent computation. This exposes a natural form of modality-level parallelism without removing interaction across modalities.

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.

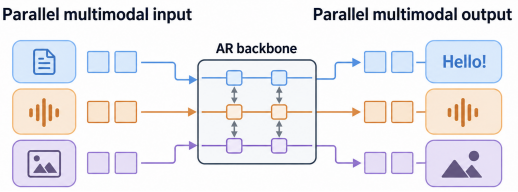

Figure 1: ModalVerse enables parallel multimodal input and output with cross-lane interaction.

Recent work on parallel LLM inference has explored how concurrent generations can cooperate on the same input to improve inference-time reasoning and efficiency. Hogwild! Inference (Rodionov et al., 2025) runs multiple LLM workers in parallel and allows them to access one another's partial progress through a shared key–value cache. Bridge (Dong et al., 2026) introduces step-wise attention across parallel generations for the same prompt, enabling them to exchange information throughout generation and improve the overall reasoning result. Instead, parallel multimodal inference introduces a different structure. Existing parallel LLM methods typically coordinate multiple generation streams operating over the same prompt and output space. In multimodal inference, however, parallel streams can carry different modalities, different contexts, and different computational roles. On the input side, distributed modality-specific evidence must be integrated into a shared prediction. On the output side, heterogeneous modalities must advance their own autoregressive sequences concurrently while remaining coordinated. The challenge is therefore not simply to run multiple streams in parallel, but to allow heterogeneous modality streams to progress independently while continuously exchanging information.

In this work, we propose **ModalVerse**, a framework that decouples modality-local autoregressive progression from cross-modal interaction (Figure 1). Each modality is organized into a parallel lane with its own causal history, while all lanes share the same autoregressive backbone. Each lane processes its own sequence independently, while cross-lane attention enables information exchange across modalities throughout computation. This design supports both parallel multimodal input processing, where distributed evidence is integrated into a shared prediction, and concurrent multimodal output generation, where different modalities generate their own sequences while remaining coordinated. Because output completion should stop visible emission rather than cross-modal participation, we further introduce *continuation tokens* that keep completed lanes active in latent communication until all outputs finish.

Experiments on MUSIC-AVQA (Li et al., 2022) and adapted A-OKVQA (Schwenk et al., 2022) demonstrate that ModalVerse achieves performance close to single-sequence baselines while enabling parallel multimodal inference. On MUSIC-AVQA, a balanced three-lane input layout delivers up to **1.51× faster decoder prefill** and reduces peak memory by up to **34.46%**, while maintaining comparable QA accuracy. On adapted A-OKVQA, ModalVerse generates text and speech concurrently with **38.26% joint accuracy**, closely matching **38.68%** for concatenated generation and substantially improving over **29.07%** for independent parallel generation. Ablations further establish the importance of cross-lane communication and show that mask tokens improve joint prediction by sustaining communication after shorter outputs finish. Together, these results demonstrate that modalities can be processed and generated in parallel while preserving effective cooperation.

Our contributions can be summarized as follows:

- We formulate multimodal autoregression as alternating *modality-local progression* and *cross-modal interaction*, exposing modality-level parallelism beyond the conventional single-sequence formulation.
- We develop **ModalVerse**, which realizes this formulation with shared-backbone parallel lanes, step-wise cross-lane attention, and continuation tokens for sustained interaction across unequal-length outputs.
- We demonstrate parallel multimodal inference on both input and output, achieving improved prefill efficiency and memory usage while preserving multimodal generation quality.

2 Revisiting Multimodal Modeling

2.1 Single-Sequence Multimodal Modeling

The success of large language models (LLMs) Brown et al. (2020); Touvron et al. (2023) has pushed language modeling beyond text-only interaction toward multimodal understanding and generation.

Early vision-language models extended LLMs with visual inputs, allowing language models to reason over both text and images Li et al. (2023); Liu et al. (2023). This capability was later expanded to richer perceptual inputs, including audio Chu et al. (2023) and video Zhang et al. (2023). More recently, multimodal models have moved beyond understanding heterogeneous inputs to generating outputs in multiple modalities, such as speech and images (OpenAI, 2024). This progression has recently led to omni models, which unify multimodal perception and generation within a single model. For instance, Qwen2.5-Omni (Xu et al., 2025) supports text, image, audio, and video inputs while generating text and speech, whereas AR-Omni (Cheng et al., 2026) further explores unified autoregressive modeling over text, image, and speech tokens for both understanding and generation.

Single-sequence Computing Paradigm. Despite differences in supported modalities and model architectures, a common computational paradigm in LLM-based multimodal modeling is to organize different modalities along a single sequence. Tokens or encoded features from different input modalities are concatenated or interleaved into a shared context, where each attention operation reads both same-modality and cross-modal representations as permitted by its mask. We refer to this organization as *single-sequence computation*: within-modality processing and cross-modal information exchange are performed through attention over the same sequence.

Formally, let $H^{(m)} \in \mathbb{R}^{S_m \times d}$ denote the hidden representations of modality m entering an attention layer, where S_m is its sequence length and d is the hidden dimension. Let P_π specify a concatenated or interleaved ordering that preserves each modality's internal token order, and let B denote the attention mask. With S denoting the total sequence length, single-sequence computation takes the form

$$H = P_\pi[H^{(1)}; \ldots; H^{(M)}] \in \mathbb{R}^{S \times d}, \qquad O = \mathrm{Attention}(H; B). \tag{1}$$

2.2 Rethinking single-sequence multimodal computation

Single-sequence computation is inherited naturally from autoregressive language modeling (Vaswani et al., 2017), where tokens follow a causal order and each prediction conditions on a growing prefix. When extended to multimodal models, different modality sequences are serialized into a single global sequence, imposing a single global causal order across modalities. However, multimodal dependencies do not inherently follow such an order: different modalities can carry either shared or complementary information, while each retains its own sequence structure and dependency pattern. For example, a conversational avatar may jointly generate speech, facial motion, and body gestures from the same dialogue context, while movie generation may coordinate visual scenes, dialogue, background audio, and sound effects over a shared narrative. In both cases, *the modalities must remain coordinated, but they need not progress according to a single global causal order.*

Separating Two Types of Dependencies. Consequently, multimodal computation involves two distinct dependencies: ***within-modality dependencies***, which govern how each modality evolves along its own sequence, and ***cross-modality dependencies***, which govern how information is exchanged across modalities. Single-sequence computation binds both to the same global sequence organization. This motivates redesigning multimodal computation around two explicit operations: modality-specific sequence processing and cross-modality interaction.

Formally, consider M modality sequences, and let $H_{l-1}^{(m)}$ denote the hidden representations of the m modalities entering layer l. Each modality first updates its representation along its own sequence, after which the resulting modality representations interact through a cross-modality operation:

$$\widetilde{H}_l^{(m)} = \mathcal{F}_l\Big(H_{l-1}^{(m)}\Big) \qquad \textit{(within-modality processing)}, \tag{2}$$

$$H_l^{(m)} = \mathcal{G}_l^{(m)}\Big(\widetilde{H}_l^{(1)}, \ldots, \widetilde{H}_l^{(M)}\Big) \qquad \textit{(cross-modality interaction)}. \tag{3}$$

Here, F_ℓ models within-modality dependencies by operating along each modality sequence, while G_ℓ models cross-modality dependencies by exchanging information among the resulting modality representations. This decomposition exposes modality-level parallelism: the M modality-local updates can be computed concurrently before cross-modal interaction synchronizes their representations for subsequent computation.

Enabling Parallel Multimodal Inference. By separating within-modality processing from cross-modality interaction as shown in Equations (2) and (3), each modality can compute its local update independently within a layer, after which the resulting representations are exchanged to inform the next layer. This organization provides a foundation for *parallel inference*, enabling different input modalities to be processed in parallel and multiple output modalities to be generated concurrently. The detailed design is presented in Section 3.

3 MODALVERSE: COOPERATIVE MODALITY-PARALLELISM

To realize multimodal parallel inference, we introduce **ModalVerse**, which organizes modality sequences into parallel lanes with separate attention mechanisms for *within-modality processing* and *cross-modality interaction* (Figure 2). Each lane maintains its own sequence and causal history while sharing the same autoregressive backbone. At each layer, the decoder block processes the lanes independently, after which cross-lane attention exchanges their hidden states to inform subsequent computation.

3.1 WITHIN-MODALITY PROCESSING

We organize modality sequences into parallel lanes that share one autoregressive backbone while maintaining separate causal histories. Given M modality lanes, we arrange their sequences along the batch dimension and process them using the same decoder parameters. This organization allows all lanes to execute the backbone in parallel without introducing a separate set of decoder parameters for each modality.

Local-lane Autoregressive Computation. Each lane maintains its own sequence state, including its token sequence, causal attention mask, and KV cache. Self-attention within the decoder therefore reads only the positions permitted by that lane's causal mask and does not attend to tokens in other lanes. Let $H_{\ell-1}^{(m)}$ denote the hidden states entering decoder block ℓ for lane m. The lane-local update is

$$\widetilde{H}_{\ell}^{(m)} = D_{\theta_\ell}\left(H_{\ell-1}^{(m)}; \mathcal{M}^{(m)}\right), \qquad m = 1, \ldots, M, \tag{4}$$

where D_{θ_ℓ} denotes decoder block ℓ, comprising causal self-attention and a feed-forward network, with parameters θ_ℓ shared across all lanes. The mask $\mathcal{M}^{(m)}$ specifies the causal attention pattern within lane m. This update gives the concrete form of the within-modality processing operation $\mathcal{F}_\ell$ introduced in Equation (2).

3.2 CROSS-MODALITY INTERACTION

Following within-modality processing, ModalVerse exchanges the updated hidden states across parallel modality lanes. This interaction takes place after each decoder block, allowing the information obtained in each lane to inform the others before processing continues at the next layer.

Step-wise Cross-Lane Attention. After each decoder block, we exchange information across modality lanes through a cross-lane attention module. At computational position t, we stack the updated hidden states from the M lanes into $\widetilde{H}_{\ell,t} \in \mathbb{R}^{M \times d}$, where each row corresponds to one modality lane. Since the lanes carry different modalities and serve different input/output roles, the interaction should have access to these distinctions. We therefore apply RMSNorm (Zhang & Sennrich, 2019) to control the scale of each lane's hidden state and propose to add learned embeddings E_ℓ encoding each lane's modality and

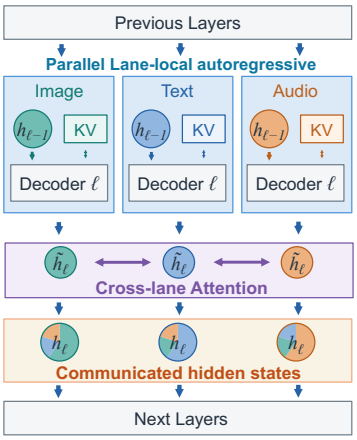

Figure 2: Overview of the ModalVerse architecture.

role:

$$Z_{\ell,t} = \mathrm{RMSNorm}_\ell\Big(\widetilde{H}_{\ell,t}\Big) + E_\ell. \tag{5}$$

For each attention head r, we multiply $Z_{\ell,t}$ by the module's learned projection matrices $W^Q_{\ell,r}$, $W^K_{\ell,r}$, and $W^V_{\ell,r}$ to obtain $Q_r, K_r, V_r \in \mathbb{R}^{M\times d_h}$, respectively. Each row of these matrices represents the query, key, or value of one modality lane. Inspired by Bridge (Dong et al., 2026), we perform cross-lane attention as

$$O_r = \mathrm{softmax}\left(\frac{Q_r K_r^\top}{\sqrt{d_h}} + B_t\right) V_r, \tag{6}$$

where d_h is the head dimension and B_t masks invalid key positions. The matrix $Q_r K_r^\top \in \mathbb{R}^{M\times M}$ contains attention scores between modality lanes, and the softmax normalizes each query's scores over the available lanes. Each modality thus aggregates information from the current value representations across lanes.

We concatenate the head outputs, project them back to the hidden dimension, and apply a residual update:

$$H_{\ell,t} = \widetilde{H}_{\ell,t} + \mathrm{Concat}_r(O_r) W^O_\ell. \tag{7}$$

The updated representations carry cross-modal information into the next decoder block, allowing each lane to continue processing its own sequence with information from the others. This update gives the concrete form of the cross-modality interaction operation $\mathcal{G}_\ell$ introduced in Equation (3).

Cross-Modality Interaction Beyond Output Completion. Step-wise cross-lane attention requires participating lanes to provide hidden states at each computation step. However, output sequences from different modalities typically have different lengths and therefore finish generating at different steps. For example, a text response may end while its speech output is still being generated. If the text lane stops computing immediately, the remaining lanes can retain information exchanged earlier but can no longer receive updated representations from it.

To address this, we introduce a **continuation token** g, an internal mask token that maintains continuous participation in cross-modality interaction after a lane's visible output ends. Once the lane processes its end token, the decoding procedure feeds g as its input at every subsequent step, keeping the lane active until all requested outputs finish from other lanes. Each occurrence of g occupies a new, valid sequence position. Through causal self-attention, this position reads the lane's existing history and produces new hidden states and KV cache entries. After each decoder block, its hidden state participates in cross-lane attention, allowing the completed lane to both receive information from other lanes and provide an updated representation of its own. Although the supplied token is fixed, its representation evolves with the growing history and cross-modality interaction. Lanes without a requested output receive g from the start of decoding and participate through the same mechanism.

Continuation tokens serve to sustain computation rather than represent output content. During training, we therefore exclude their positions from the prediction loss while allowing their hidden states to support predictions in other lanes through cross-lane attention. During inference, their insertion follows the lane's completion status rather than a model prediction, so the decoding procedure supplies them directly and omits them from the returned outputs.

4 Parallel Multimodal Inference

In this section, we describe how ModalVerse supports two complementary settings of parallel multimodal inference (Figure 3). In *N-to-1 parallel multimodal input processing*, modality lanes process distributed input contexts concurrently and integrate their information into a shared prediction. In *1-to-N parallel multimodal output generation*, modality lanes start from a shared context and generate modality-specific outputs concurrently while continuously exchanging information.

4.1 Parallel Multimodal Input

Processing Multimodal Inputs in Parallel. Given a question q and modality inputs $x^{(1)}, \ldots, x^{(M)}$, we construct K parallel lanes, each containing a modality-specific input and the shared question. With one lane per modality, $K = M$. We format each lane using the backbone's input template, left-pad the prompts to a common length, and process them in parallel.

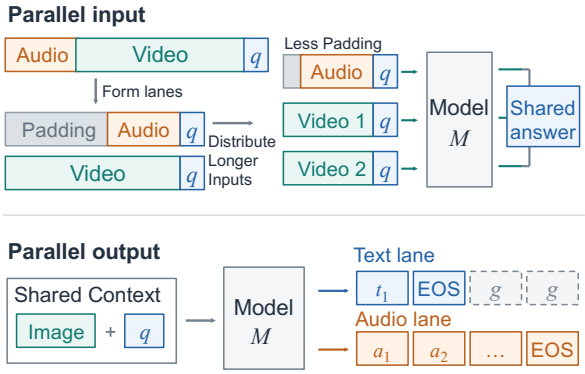

Figure 3: Overview of the two ModalVerse settings.

Shared-Answer Decoding and Training. Since all K lanes provide complementary evidence for the same question, we combine their predictions into a shared answer by averaging their output logits at each decoding step t:

$$\hat{y}_t = \arg\max_{w \in \mathcal{V}} \left[\frac{1}{K} \sum_{k=1}^{K} \mathbf{z}_t^{(k)} \right]_w , \tag{8}$$

where $\mathcal{V}$ is the output vocabulary, $\mathbf{z}_t^{(k)}$ is the output logit vector of lane k, and $[\cdot]_w$ denotes the entry corresponding to token w. The selected token is fed back to every lane, so all lanes share the same answer prefix while retaining their own input context.

During training, each lane predicts the same target answer $y = (y_1, \ldots, y_T)$ under teacher forcing. With $p_t^{(k)} = \mathrm{softmax}(\mathbf{z}_t^{(k)})$, we average the prediction loss across lanes and target positions:

$$\mathcal{L}_{\text{in}} = -\frac{1}{KT} \sum_{k=1}^{K} \sum_{t=1}^{T} \log p_t^{(k)}(y_t). \tag{9}$$

Balancing Computation Across Lanes. Modality inputs can differ substantially in sequence length. When their lanes are batched together, the longest sequence determines the padded length, introducing unnecessary computation at padding positions in shorter lanes. To reduce this imbalance, we distribute longer modality inputs across additional lanes. This shortens the longest sequence and reduces the padding required for batched processing, while cross-lane attention allows information to be exchanged across the resulting segments. In our audio–video setting, we split the sampled video into two consecutive segments at a temporal patch boundary, yielding two video lanes and one audio lane while retaining all sampled frames. Each lane receives the shared question and follows the same shared-answer prediction rule. With K lanes serving M modalities, decoding and training average over the K lanes.

Inference Efficiency. A key benefit of parallel multimodal input processing is improved inference efficiency through shorter attention sequences and concurrent computation across modality lanes. For S input tokens evenly distributed across K lanes, with hidden dimension d and excluding duplicated prompt tokens, the per-layer prefill attention cost decreases from $O(S^2 d)$ to $O(S^2 d/K + SKd)$, where the first term accounts for self-attention within the K shorter lanes and the second for cross-lane attention at each sequence position. This parallelism is available even with a request batch size of one, making it particularly useful for personal-device inference where opportunities for request-level batching are limited.

4.2 Parallel Multimodal Output

Generating Multiple Modalities in Parallel. ModalVerse initializes the output lanes with a shared context and modality-specific control tokens. Unlike parallel multimodal input processing, which feeds a shared token back to every lane, each output lane selects its own next token and appends it to its own prefix. A single decoding step therefore advances all unfinished modality outputs, even when their tokenizations and sequence lengths differ. Each lane stops emitting upon reaching its end token and then receives continuation tokens until the remaining outputs finish.

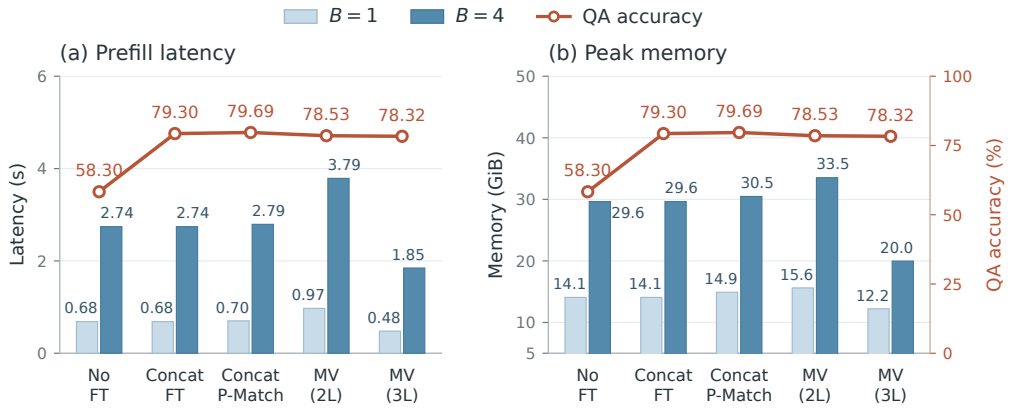

Figure 4: Question-answering accuracy vs. prefill latency vs. peak memory on the MUSIC-AVQA Audio-Visual test split across methods. MV (2L) marks for ModalVerse (2 lanes). MV (3L) marks for ModalVerse (3 lanes). B marks for Batch size.

Training for Parallel Generation. Each output lane is supervised with its own modality-specific target sequence under teacher forcing, while cross-modality interaction remains active throughout training. Because target lengths differ across modalities, we average the token loss within each sequence before averaging across output lanes, preventing longer sequences from receiving greater weight solely because they contain more tokens. Let $\mathcal{O}$ denote the set of output lanes, T_m the target length of lane m, and $p_t^{(m)}$ its next-token distribution. The training objective is

$$\mathcal{L}_{\text{out}} = -\frac{1}{|\mathcal{O}|} \sum_{m \in \mathcal{O}} \frac{1}{T_m} \sum_{t=1}^{T_m} \log p_t^{(m)} \left(y_t^{(m)} \right), \tag{10}$$

where $y_t^{(m)}$ is the target token. Continuation-token positions are excluded from the loss while remaining active in cross-modality interaction.

Toward Streaming Multimodal Generation. A key benefit of parallel multimodal output is its potential to support streaming generation across modalities. By generating multiple modalities concurrently as the response unfolds, ModalVerse avoids waiting for one modality to complete before producing another, making the design suitable for responsive multimodal applications.

5 Experiments

We evaluate ModalVerse in the two settings introduced in Section 4: audio–visual question answering with distributed inputs and concurrent text–speech generation from a shared context. The experiments examine answer quality, decoder-prefill efficiency, and the contributions of cross-lane communication and mask-token continuation.

5.1 Parallel Multimodal Input

Experimental Setup. We evaluate the Qwen2.5-Omni-3B Thinker (Xu et al., 2025) on the Audio-Visual subset of MUSIC-AVQA (Li et al., 2022), using the official splits of 17,815 training, 2,549 validation, and 5,096 test questions. Each example pairs audio and video from the same clip with a question and a single text answer. We compare two ModalVerse layouts: an audio lane plus a video lane, and an audio lane plus two consecutive video segments. The single-sequence baselines are a parameter-count-matched two-layer MLP adapter (Concat P-Match) and full fine-tuning (Concat FT).

All trained models use five epochs and a global batch size of 128 on eight H200 NVL GPUs. We sample video at one frame per second, retaining the same frames across layouts. Checkpoints are selected by validation answer-token negative log-likelihood. Evaluation uses the PyTorch SDPA math

Table 1: Inference-time cross-lane attention ablation on MUSIC-AVQA. Both settings use the same trained two-lane ModalVerse checkpoint and the same shared-answer logit-averaging rule.

Cross-lane attention	Accuracy (%)	Correct answers	Invalid responses
Enabled	78.53	4,002	0
Disabled	58.32	2,972	308

backend (Paszke et al., 2019), and greedy decoding with a maximum of eight generated tokens and the same answer normalization for all models.

Main Results. Figure 4 summarizes question-answering accuracy, decoder-only prefill latency, and peak memory on the MUSIC-AVQA Audio-Visual test split. ModalVerse with one audio lane and one video lane, denoted as MV (2L), achieves 78.53% accuracy, within 1.2 percentage points of the two single-sequence baselines, Concat FT and Concat P-Match. Further splitting the video across two lanes, denoted as MV (3L), retains similar answer accuracy (78.32%) while shortening the per-lane contexts.

We measure decoder-only prefill using precomputed audio and video features with the PyTorch SDPA math backend. The resulting prefill latency and peak memory are shown in Figure 4(a) and (b), respectively. We use the first test example for $B = 1$ and the first four for $B = 4$. With one lane per modality, the longer video sequence determines the padded width, leaving unused positions in the audio lane. At $B = 4$, splitting the video across two lanes reduces the padded width by 49.45%, and the padding ratio is reduced by approximately 62.5%. Relative to the two-lane layout, this reduces prefill latency by 51.26% and peak memory by 40.42%.

The three-lane layout also improves prefill efficiency over Concat P-Match. At $B = 1$, MV (3L) achieves a $1.46\times$ prefill speedup and an 18.04% reduction in peak memory. At $B = 4$, these improvements increase to a $1.51\times$ speedup and a 34.46% reduction in peak memory. The additional video lane thus shortens the contexts used to process the same sampled inputs, delivering both runtime and memory savings while retaining similar answer accuracy.

Integrating Evidence Across Lanes. We evaluate the same trained two-lane ModalVerse checkpoint with cross-lane attention enabled or disabled at inference (Table 1). Both settings retain shared-answer decoding through logit averaging. Disabling communication reduces accuracy from 78.53% to 58.32%, a drop of 20.21 percentage points corresponding to 1,030 fewer correct answers. It also produces 308 invalid responses, compared with none when communication is enabled. This drop identifies hidden-state exchange during computation as a key contributor to the model's integration of distributed audio–visual evidence.

5.2 Parallel Multimodal Output

Experimental Setup. We evaluate output-side parallelism using AR-Omni (Cheng et al., 2026) on an adapted A-OKVQA (Schwenk et al., 2022) task. Given an image, a question, and four answer options, the model generates the correct option label as text and the corresponding option content as speech. We use all valid A-OKVQA questions, comprising 17,020 training questions and 1,137 validation questions. For each question, we construct four cyclic option orderings, placing the correct answer once in each position. We randomly sample one ordering per question per training epoch and evaluate all four orderings for each validation question, yielding 4,548 question–ordering instances. We synthesize training speech from the correct option text using CosyVoice-2-0.5B (Du et al., 2024) at a sampling rate of 24 kHz. The synthesized waveforms are then encoded into target audio tokens using WavTokenizer (Ji et al., 2025).

We compare **Native Concat**, which generates text and speech in a single autoregressive sequence; **Independent Parallel**, which uses separate lanes without cross-lane communication or mask-token continuation; and **ModalVerse**. All methods are initialized from AR-Omni-Chat-v0.1. Native Concat converges after five epochs, while both parallel methods use a ten-epoch training budget, with early stopping selecting epoch 6. We fully fine-tune the parallel models with a learning rate of 10^{-5} and a global batch size of 32, using 320 warmup steps for ModalVerse and 160 for Independent Parallel, followed by linear decay.

We report text accuracy, audio accuracy based on normalized speech-recognition transcripts, and joint accuracy, which requires both outputs to identify the ground-truth answer. Audio evaluation details are provided in Appendix B.1.

Main Results. Present in Table 2, relative to Independent Parallel, ModalVerse raises audio and joint accuracy by 4.90 and 9.19 percentage points, corresponding to 418 additional question–ordering instances with both outputs correct. Its audio and joint accuracies are within 0.37 and 0.42 percentage points of Native Concat, respectively. Although Independent Parallel achieves the highest text accuracy, it yields the lowest joint accuracy, highlighting a key limitation of independent generation: strong performance in one modality does not ensure jointly correct outputs. In contrast, cross-lane cooperation in ModalVerse improves joint success by enabling the parallel outputs to work toward the same answer decision.

Table 2: Text and speech generation on adapted A-OKVQA, evaluated over 4,548 question–ordering instances from 1,137 validation questions. Joint accuracy requires both outputs to identify the correct answer. All accuracies are reported in percent.

Method	Text	Audio	Joint
Native Concat	55.56	38.72	38.68
Independent Parallel	56.24	33.44	29.07
ModalVerse	54.71	38.35	38.26

Sharing Computation Across Modalities. The improvement in joint accuracy includes greater overlap between correct text and speech predictions. For Independent Parallel, 199 of its 1,521 audio-correct predictions have an incorrect text answer. ModalVerse reduces this to 4 out of 1,744 audio-correct predictions. The proportion of correct audio outputs accompanied by a correct text answer therefore rises from 86.92% to 99.77%. ModalVerse produces more correct speech outputs and pairs a greater share of them with correct text decisions.

Table 3: Inference-time cross-lane attention ablation for output generation. Both settings use the same ModalVerse checkpoint, with continuation tokens disabled. All accuracies are reported in percent.

Cross-lane attention	Text	Audio	Joint
Enabled	54.71	37.42	37.29
Disabled	55.04	21.55	15.96

We further examine cross-lane communication by disabling it at inference while keeping the ModalVerse checkpoint fixed. Mask-token continuation is disabled in both conditions (Table 3). Removing communication reduces audio accuracy and joint accuracy by 15.87 and 21.33 percentage points. Compared with text accuracy, the much larger change in speech accuracy shows that speech generation draws strongly on cross-lane information, contributing to the correctness of the joint output.

Cooperation Beyond Visible Output. We evaluate mask-token continuation at epochs 2, 4, and 6 while keeping cross-lane attention enabled. Each mask-on/off pair uses the same checkpoint (Table 4). Continuation improves both audio and joint accuracy at all three checkpoints. Joint accuracy increases by 3.25 percentage points at epoch 2 and 2.75 percentage points at epoch 4. At epoch 6, audio and joint accuracies increase by 0.92 and 0.97 percentage points, respectively.

Table 4: Inference-time mask-token continuation at epochs 2, 4, and 6. Cross-lane attention remains enabled, and each mask-on/off pair uses the same checkpoint. All accuracies are reported in percent.

	Audio accuracy		Joint accuracy	
Epoch	Mask off	Mask on	Mask off	Mask on
2	23.59	26.91	23.59	26.85
4	32.04	34.85	31.95	34.70
6	37.42	38.35	37.29	38.26

Mask tokens keep completed and non-emitting lanes updating their states and exchanging information with the continuing speech lane. The consistent improvement shows that keeping these lanes active improves the longer speech output and, in turn, joint prediction.

6 Conclusion

We present ModalVerse, a framework that separates intra-modal autoregression from cross-modal communication to support parallel input processing and concurrent output generation. Cross-lane attention enables cooperation across separate sequences, while mask-token continuation keeps completed and non-emitting lanes participating in ongoing computation. Experiments show that balanced input lanes improve prefill efficiency while maintaining comparable answer accuracy, and that cooperative text–speech generation achieves joint accuracy close to concatenated generation. These results demonstrate that unified multimodal modeling can be organized around cooperation across parallel modality sequences rather than a single shared sequence.

Reproducibility Statement.

We provide an anonymous implementation of ModalVerse in the supplementary material, including end-to-end README and scripts for environmental setup, model training, and model evaluation. Additional details, such as hyperparameters and experimental settings, are described in the main text and appendix to ensure full reproducibility.

AI use statement

In accordance with the ICLR 2027 policy on large language model (LLM) usage, we disclose that LLM-based tools (e.g., ChatGPT) were used solely to aid in polishing the writing and improving the clarity of exposition. In addition, pretrained generative audio models, including CosyVoice-2-0.5B and WavTokenizer, were used as part of the experimental pipeline for text-to-speech synthesis and audio tokenization, respectively, as described in Sec 5.2. These tools were not used for research ideation, experimental design, data analysis, or drawing scientific conclusions. All scientific content, results, and conclusions are the responsibility of the authors.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Gabriele Cesa, Thomas Hehn, Aleix Torres-Camps, Àlex Batlle Casellas, Jordi Ros-Giralt, Arash Behboodi, and Tribhuvanesh Orekondy. LaneRoPE: Positional encoding for collaborative parallel reasoning and generation. In *Workshop on Multi-Agent Learning and Its Opportunities in the Era of Generative AI*, 2026. URL https://openreview.net/forum?id=6WAuvwZjmw.

Dongjie Cheng, Ruifeng Yuan, Yongqi Li, Runyang You, Wenjie Wang, Liqiang Nie, Lei Zhang, and Wenjie Li. AR-Omni: A unified autoregressive model for any-to-any generation, 2026. URL https://arxiv.org/abs/2601.17761.

Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models, 2023. URL https://arxiv.org/abs/2311.07919.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue, 2024. URL https://arxiv.org/abs/2410.00037.

Harry Dong, David Brandfonbrener, Eryk Helenowski, Yun He, Mrinal Kumar, Han Fang, Yuejie Chi, and Karthik Abinav Sankararaman. Generalized parallel scaling with interdependent generations. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 43631–43646, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/48050f08c8f654e473a02ddae936dfb5-Paper-Conference.pdf.

Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao, Yexin Yang, Changfeng Gao, Hui Wang, Fan Yu, Huadai Liu, Zhengyan Sheng, Yue Gu, Chong Deng, Wen Wang, Shiliang Zhang, Zhijie Yan, and Jingren Zhou. Cosyvoice 2: Scalable streaming speech synthesis with large language models, 2024. URL https://arxiv.org/abs/2412.10117.

Chan-Jan Hsu, Davide Buffelli, Jamie McGowan, Feng-Ting Liao, Yi-Chang Chen, Sattar Vakili, and Da-shan Shiu. Group Think: Multiple concurrent reasoning agents collaborating at token level granularity, 2025. URL https://arxiv.org/abs/2505.11107.

Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, zehan wang, Ruiqi Li, Ziang Zhang, Xiaoda Yang, Rongjie Huang, YIDI JIANG, Qian Chen, Siqi Zheng, and Zhou Zhao. Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 93809–93826, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ea1f5f0878d43ff4fb8bf64ef4a2326c-Paper-Conference.pdf.

Guangyao Li, Yake Wei, Yapeng Tian, Chenliang Xu, Ji-Rong Wen, and Di Hu. Learning to answer questions in dynamic audio-visual scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19108–19118, June 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, 2023. URL https://arxiv.org/abs/2301.12597.

Lijiang Li, Zuwei Long, Yunhang Shen, Heting Gao, Haoyu Cao, Xing Sun, Caifeng Shan, Ran He, and Chaoyou Fu. Omni-Diffusion: Unified multimodal understanding and generation with masked discrete diffusion. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=EulXRTtFCd.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. URL https://arxiv.org/abs/2304.08485.

OpenAI. GPT-4o system card. Technical report, OpenAI, August 2024. URL https://openai.com/index/gpt-4o-system-card/.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28492–28518. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/radford23a.html.

Gleb Rodionov, Roman Garipov, Alina Shutova, George Yakushev, Erik Schultheis, Vage Egiazarian, Anton Sinitsin, Denis Kuznedelev, and Dan Alistarh. Hogwild! Inference: Parallel LLM generation via concurrent attention. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 46592–46633. Curran Associates, Inc., 2025. doi: 10.52202/085713-1551. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/4293727fc48c60b4adfe7a8e44f6a446-Paper-Conference.pdf.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-okvqa: A benchmark for visual question answering using world knowledge. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 146–162, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-20074-8.

Guinan Su, Yanwu Yang, Xueyan Li, and Jonas Geiping. Multi-Stream LLMs: Unblocking language models with parallel streams of thoughts, inputs and outputs, 2026. URL https://arxiv.org/abs/2605.12460.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

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, 2025. URL https://arxiv.org/abs/2503.20215.

Xinyu Yang, Yuwei An, Hongyi Liu, Tianqi Chen, and Beidi Chen. Multiverse: Your language models secretly decide how to parallelize and merge generation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 100773–100804. Curran Associates, Inc., 2025. doi: 10.52202/085713-3371. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/91b1966914203338495a76d13b23599c-Paper-Conference.pdf.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/1e8a19426224ca89e83cef47f1e7f53b-Paper.pdf.

Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding, 2023. URL https://arxiv.org/abs/2306.02858.

Appendix

A Related Work

Unified Multimodal Modeling. Unified multimodal models use different computational structures for understanding and generation. AR-Omni (Cheng et al., 2026) models text, images, and speech within a single autoregressive token stream. Qwen2.5-Omni (Xu et al., 2025) interleaves audio and video on the input side and couples text generation with streaming speech through its Thinker–Talker architecture. Moshi (Défossez et al., 2024) models user and system speech as parallel streams, together with time-aligned text, for full-duplex dialogue. Omni-Diffusion (Li et al., 2026) uses masked discrete diffusion to predict multimodal tokens in parallel within a shared sequence. ModalVerse separates lane-local autoregression from cross-modal communication, allowing modality sequences to progress in parallel through a shared backbone while exchanging hidden states.

Parallel and Cooperative LLM Inference. Parallel LLM inference explores task decomposition, communication between ongoing generations, and concurrent streams with different roles. Multiverse (Yang et al., 2025) decomposes a task into parallel subtasks and merges their results. Hogwild! Inference (Rodionov et al., 2025) allows concurrent workers to share a key–value cache, while Group Think (Hsu et al., 2025) enables reasoning trajectories to interact through token-level access to one another's partial generations. Bridge (Dong et al., 2026) exchanges the current hidden states of parallel responses to the same prompt through step-wise attention for test-time scaling. LaneRoPE (Cesa et al., 2026) combines inter-sequence attention masking with positional encoding that captures relative token positions within and across sequences. Multi-Stream LLMs (Su et al., 2026) allow reading, thinking, and responding to proceed concurrently through streams with cross-stream causal attention. In ModalVerse, cooperation is organized across a task's modalities: input lanes contribute distributed evidence to a shared prediction, and output lanes generate modality-specific sequences while exchanging hidden states. Mask-token continuation keeps completed and non-emitting lanes participating as the other outputs continue. In contrast, ModalVerse introduces a unified paradigm for parallel and cooperative multimodal modeling, encompassing both multimodal understanding and generation.

B Detailed Experimental Setup

B.1 Detailed setup for parallel multimodal output

Audio Evaluation. For Native Concat, Independent Parallel and ModalVerse, we decode generated audio tokens with WavTokenizer and transcribe the resulting waveforms using Whisper (Radford et al., 2023). Audio accuracy uses English and VQA text normalization followed by exact matching against the correct option content.

C Scaling Prefill with Context Length and Batch Size

We examine prefill scaling through a controlled synthetic-token sweep using the PyTorch SDPA math backend (Paszke et al., 2019) in Table 5. Each comparison fixes the request batch size B and total input length L per request. Concat uses Concat P-Match, while the two- and three-lane layouts split L evenly. The context-length sweep fixes $B = 1$ and uses $L \in \{1{,}536, 3{,}072, 6{,}144, 12{,}288, 24{,}576\}$; the batch sweep fixes $L = 6{,}144$ and varies $B \in \{1, 2, 4, 8\}$. We measure median CUDA-event latency for decoder and adapter prefill, together with peak allocated GPU memory, on a single H200 NVL.

At $B = 1$, the three-lane speedup over Concat P-Match increases from $1.64\times$ at $L = 6{,}144$ to $2.53\times$ at $L = 24{,}576$. At the latter length, prefill latency falls from 9.40 to 3.72 s and peak memory from 95.08 to 39.08 GiB, a 58.90% reduction. With $L = 6{,}144$, the speedup remains between $1.64\times$ and $1.67\times$ as B increases from 1 to 8, while the memory reduction grows from 22.41% to 49.61%.

The sweep also demonstrates the benefit of distributing a fixed context across more lanes. At $B = 1$ and $L = 24{,}576$, moving from two to three lanes reduces prefill latency by 34.49% and peak memory by 26.11%. Shorter per-lane contexts yield larger savings as the total context grows.

Table 5: Synthetic-token prefill sweep on a single H200 NVL using the PyTorch SDPA math backend. B is the request batch size and L is the total input length per request. Concat uses Concat P-Match; the two- and three-lane layouts split L evenly. Latency is the median CUDA-event measurement for decoder and adapter prefill; memory is peak allocated GPU memory.

		Prefill latency (ms)			Peak memory (GiB)		
B	L	Concat	2 lanes	3 lanes	Concat	2 lanes	3 lanes
Context-length sweep ($B = 1$)							
1	1,536	82.39	97.09	89.70	10.10	9.94	9.88
1	3,072	216.85	217.64	161.66	11.22	10.59	10.37
1	6,144	672.76	540.36	409.83	15.39	12.81	11.94
1	12,288	2,588.77	1,592.38	1,140.48	31.58	21.03	17.58
1	24,576	9,396.82	5,674.20	3,716.90	95.08	52.89	39.08
Batch sweep ($L = 6{,}144$; $B = 1$ reported above)							
2	6,144	1,327.48	1,068.84	802.36	21.03	15.85	14.15
4	6,144	2,626.57	2,115.76	1,578.56	32.22	21.99	18.60
8	6,144	5,257.62	4,251.25	3,156.75	54.66	34.36	27.54