DISTMOE: Private-data Rehearsal-free Routing in Mixture-of-Experts for Distributed Instruction Tuning

Mainak Singha[1] **Niccolò Biondi**[1] **Elisa Ricci**[1,2] **Subhankar Roy**[3]

[1]University of Trento, Italy [2]Fondazione Bruno Kessler, Italy
[3]University of Bergamo, Italy

{mainak.singha, niccolo.biondi, e.ricci}@unitn.it, subhankar.roy@unibg.it

Website Code

Abstract

Multimodal Large Language Models (MLLMs) have shown strong multimodal instruction-following ability, but adapting them to diverse visual-language domains typically assumes centralized data access and costly joint training. This is restrictive when data is distributed across private, domain-specific, or permission-limited clients. To this end, we propose DISTMOE, a mixture-of-experts (MoE) approach for distributed visual instruction tuning. In each layer of the language decoder it augments the public feedforward network (FFN) with a client-specific private FFN expert, with the goal to acquire domain-specific knowledge. However, independent expert training causes the private FFNs to learn representation of different scales and magnitudes, making merging the experts difficult. To reduce client-specific drift, we introduce a public-anchored expert composition stage that updates only routers and lightweight private projection adapters on a mix of *local* client data and public data, via an isotropic regularization loss, therefore making it *cross-client rehearsal-free composition*. During inference, DISTMOE performs modular routing over public and private experts, enabling token-wise domain composition without explicit domain labels. Experiments across diverse visual-language benchmarks show that DISTMOE enables flexible expert reuse, effective domain adaptation, and competitive performance while preserving modular control over client-specific knowledge.

1 Introduction

Multimodal Large Language Models (MLLMs) [2] have excelled in visual instruction following and reasoning by aligning visual encoders with Large Language Models (LLMs). These gains are often driven by scaling model size [3, 4] and augmenting visual instruction tuning data [5, 6]. Although impressive, such training paradigms present two challenges from training and deployment perspective: (i) a centralized access to all the training data, and (ii) high computation and memory costs during deployment. Thus, it calls for visual instruction tuning approaches that can be trained in a distributed manner and that result in models with a lower computational footprint during inference.

Mixture-of-Experts (MoE) approach [7, 8, 9] presents a promising solution towards this goal, as opposed to dense models, it allows to scale models while activating only a subset of the parameters (or *experts)* both during training and inference [1], as well as enables distributed training [10]. For example, MoE-LLaVA [1], MoE-based visual instruction tuning approach, have shown that sparse expert routing can improve MLLM efficiency while maintaining competitive visual understanding and reduced hallucination with fewer activated parameters than dense baselines. Despite the advantage, MoE-LLaVA assumes centralized access to training data and requires joint training of experts and

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

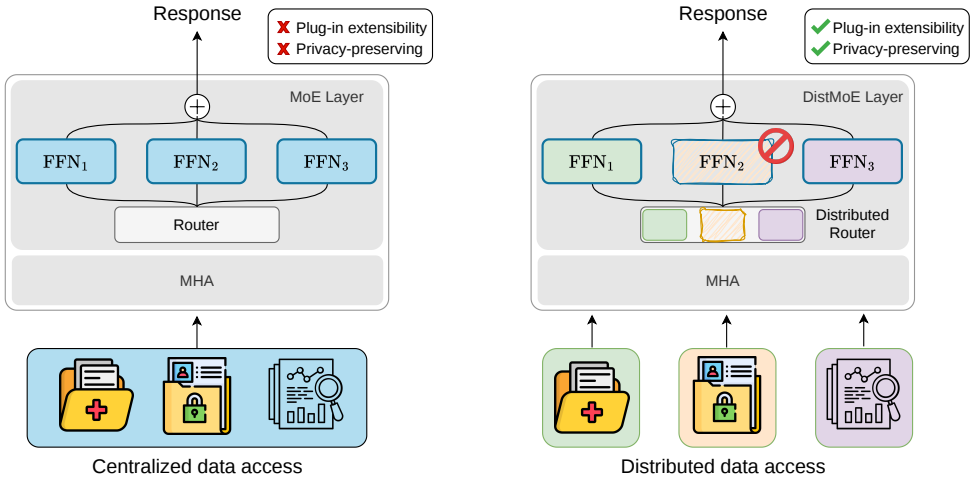


Figure 1: Centralized MoE tuning vs. DISTMOE. **Left:** Centralized MoE methods [1] rely on joint access to domain-specific data. **Right:** DISTMOE independently trains domain-specific private experts with distributed data access and composes them via a distributed router. This enables experts to be flexibly *added*, *removed*, or *reused* at inference time depending on the application needs

routers (see Fig. 1). This assumption is restrictive, as visual instruction tuning data can be privately maintained [11]. Moreover, once such data is utilized to jointly train a MoE model in a centralized setup, it becomes difficult to control, remove, and selectively disable its influence during inference.

A set of works [10, 12, 13] takes an important step towards flexible data use in LLMs, by independently training experts on private *domain-specific* data slices (e.g., coding, maths, news) and merging them into a unified model through the MoE approach. In detail, a router is trained that learns to combine the domain-specific experts during inference. However, it still requires a *rehearsal* of a subset of the private data from all domains to learn the router, which can still be a bottleneck in extremely privacy-sensitive applications. While FlexOlmo [12] proposes a technique to merge the experts without further tuning the router, independent expert tuning can cause the experts to diverge making routing scores during inference not comparable. This issue can further be exacerbated in visual instruction tuning as it has been shown that a model can learn to ignore visual information [14].

In this work, we propose Distributed MoE (DISTMOE), a *cross-client rehearsal-free composition* MoE approach [7] for distributed visual instruction tuning (see Fig. 1). To be noted, *cross-client rehearsal-free composition* refers that each client may reuse its own local data, but expert composition never requires pooling, sharing, or jointly revisiting private data from multiple clients. The novelty of DISTMOE lies in how independently trained experts are calibrated before composition. In detail, following FlexOlmo, we first independently train *private* experts, one per domain, by initializing it from a *public* model, to acquire domain-specific knowledge. However, different from FlexOlmo, after expert-training, we introduce a lightweight and privacy-preserving approach to ensure that the private expert remains anchored to the public expert that is shared across all the clients. This is enforced by imposing a Gaussian structure [15] on the residuals between the public and private feature representations in the intermediate layers of the Transformer. In addition, we calibrate the router using a mix of public and strictly client-specific private data. As a result, the representation and routing scores across domains become comparable, making merging straightforward during inference. The native MoE architecture endows flexibility to DISTMOE by enabling computation-efficient sparse routing and seamless exclusion of sensitive domain-specific knowledge during inference. Here, efficiency refers to a fixed expert-activation budget relative to dense all-expert execution, not to a speedup over FlexOlmo (Sec. 5.2). Likewise, DISTMOE provides *data locality* rather than formal privacy guarantees, and excluding an expert disables its influence at inference.

Our **contributions** are: *i)* We introduce DISTMOE, a MoE architecture-based approach, for distributed visual instruction tuning in MLLMs. *ii)* To facilitate seamless composition of knowledge across experts, we introduce a novel *cross-client rehearsal-free composition* technique that calibrates private experts and the associated router weights without relying on joint access to private data from all clients. *iii)* We conducted experiments on several multimodal evaluation benchmarks to evaluate

in-domain and cross-domain generalization. Our results demonstrate the effectiveness and flexibility of DISTMOE.

2 Related Works

Visual instruction tuning, pioneered by LLaVA [16], has emerged as a successful recipe for building multimodal LLMs from pre-trained models for addressing tasks such as visual question answering (VQA). It aligns visual encoder (e.g., CLIP [17]) with LLM decoder (e.g., Vicuna [18]) using GPT generated instruction-following data [2]. However, methods like LLaVA relies on the assumption that all the instruction tuning data is available simultaneously. Instead, in real-world applications, introduction of new tasks and domains requires the model to learn dynamically over time. This has shifted the research focus towards Continual Visual Instruction Tuning (CVIT) [19, 20, 21, 22, 23, 24], where the goal is to learn continually while avoiding catastrophic forgetting by employing regularization techniques. A characteristic assumption made in CVIT is that tasks arrive sequentially and therefore, the employed regularization techniques are specifically designed towards striking a balance between stability and plasticity. We argue that as training becomes more and more decentralized on heterogeneous hardware [25], enforcing regularization while learning a new task is limiting and impractical. Moreover, as argued in the work [26], very often task boundaries cannot be clearly demarcated in vision-language tasks. Thus, we seek a more general and flexible framework that can enable *distributed* visual instruction tuning, where knowledge can be acquired independently and consolidated post-hoc without explicitly accounting for task type or arrival order.

Model merging provides a practical way to combine independently trained models into a unified system [27] without requiring access to raw training data or expensive computation. Existing methods mainly include weight averaging or output ensembling. Weight averaging approaches, such as Model Soups [28], combine models fine-tuned from the same initialization, while output ensembling methods, such as Branch-Train-Merge (BTM) [29], merge predictions from models trained independently in different domains. While model merging inherently offers more flexibility than regularization-based continual learning approaches, model merging is prone to interference between parameters of constituent models, resulting in performance drops [30]. To resolve this, additional steps are typically involved, such as selective merging or post-hoc fine-tuning. These additional steps introduce challenges in a purely distributed setup, where data is locally maintained.

Mixture-of-Experts (MoE) tuning, originally designed to scale deep networks [7] without increasing computation, dynamically selects feedforward networks or *experts* using Top-K routing scheme that activates only a fraction of the total number of parameters. It has been successfully applied to scale LLM training [9, 31], vision-language tasks [32, 33, 8, 1] and privacy-preserving learning [10, 12, 13]. For instance, MoE-LLaVA [1] extends MoE tuning to visual instruction tuning through training experts on different domains, but requires joint access to the union of all expert-training datasets. Similar to our setup, more privacy-friendly approaches, such as Branch-Train-Mix (BTX) [10], FlexOlmo [12] and BAR [13], were proposed that first independently train models and later combine them using MoE routing. However, all these approaches are designed for LLMs and assume access to *rehearsal* training data to learn the routing mechanism. In contrast, DISTMOE is designed for distributed visual instruction tuning and *composes experts without cross-client rehearsal*.

3 Distributed Visual Instruction Tuning

3.1 Problem Formulation

Let $\mathcal{D} = \{(I_i, q_i, y_i) \mid i \in \mathbb{N}\}$ denote a dataset of image, a textual query, and the target answer tuples. In visual instruction tuning (e.g., LLaVA [16]), the goal is to jointly train and/or fine-tune pre-trained LLM, vision transformer (ViT) [17] and a multimodal projector to predict the target y given image and query (I, q) pair at the input. The prediction is defined as:

$$\hat{y} = f_\theta(I, q) = f_{\theta_{\text{llm}}}(f_{\theta_{\text{mm}}}(f_{\theta_{\text{vis}}}(I)), q)\,, \tag{1}$$

where $f_{\theta_{\text{vis}}}$ is the vision encoder that processes the input image I, $f_{\theta_{\text{mm}}}$ is the multimodal projector, and $f_{\theta_{\text{llm}}}$ is the LLM decoder; and $\theta = \{\theta_{\text{vis}}, \theta_{\text{mm}}, \theta_{\text{llm}}\}$ are the parameters of the model f_θ.

Similar to [12], in distributed visual instruction tuning, we assume that the instruction tuning dataset $\mathcal{D} = \{\mathcal{D}_1, \cdots, \mathcal{D}_M\}$ is a collection of locally maintained datasets, spread among M clients. Each

client m is considered to possess *domain-specific* instruction tuning dataset $\mathcal{D}_m$ that is private. In addition, we assume access to a public dataset $\mathcal{D}_{\text{pub}}$ that is available to all clients. Our objective is to obtain a single model f_θ that is constructed by combining the knowledge of the clients, each independently trained on $\mathcal{D}_m$, without requiring joint access to the private datasets.

3.2 Preliminaries

The idea of MoE [7, 9], originally designed to scale a LLM's capacity without a proportional increase in computation, has been extended for visual instruction tuning in MoE-LLaVA [1]. The main goal remains the same: constructing a sparse multimodal model with high learning capacity but with a constant computational cost. We note that this setup in MoE-LLaVA lends well while learning from diverse instruction tuning datasets as one Feed Forward Network (FFN) or *expert*, in the LLM decoder, can specialize on each domain-specific dataset. Then, as in standard MoE, a learned router in the decoder can redirect the input tokens (both visual and text) to relevant domain experts.

More formally, let $\mathcal{V} = [v_1, \cdots, v_P]$ denote the sequence of P tokens corresponding to the image I from the multimodal projector $f_{\theta_{\text{mm}}}$, and $\mathcal{T} = [t_1, \cdots, t_N]$ be the sequence of N text tokens corresponding to the input query q. These tokens are concatenated and fed to the LLM decoder, which comprise of stacked multi-head self-attention (MSA) and FFN blocks. MoE-LLaVA additionally consists of a MoE layer, comprised of multiple expert FFNs and a router. Denoting the expert FFNs as $\mathcal{E} = [e_1, \cdots, e_M]$; and the router, which is a linear layer, predicts the probability of each token being assigned to the $i-$th expert as:

$$\alpha_i = \text{softmax}(\mathbf{x})_i = \frac{\exp(g(\mathbf{x})_i)}{\sum_j^M \exp(g(\mathbf{x})_j)}, \tag{2}$$

where $\mathbf{x}$ denotes the intermediate representation of a token; $g(\mathbf{x})_i = \mathbf{x}^\intercal \mathbf{r}_i$ and $\mathbf{r}_i$ are the logit score and router weight corresponding to the $i-$th FFN, respectively. For each token in a sequence, the top$-k$ highest probabilities are selected, which are then used to compute a weighted-sum representation as:

$$\mathbf{o} = \text{MoE}(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i \cdot \mathcal{E}_i(\mathbf{x}), \tag{3}$$

where $\mathbf{o}$ is the output of the MoE layer. MoE-LLaVA trains both the experts and the router with next-token prediction and router balancing loss, in a *centralized* fashion, by having joint access to all the domain-specific datasets. However, joint access presents as a bottleneck in the distributed visual instruction tuning task. While in theory, each expert FFN $\mathcal{E}_m$ can be trained independently on $\mathcal{D}_m$, and later merged as in FlexOlmo [12], it still requires rehearsal of the private data to learn the router.

4 DISTMOE: Private-data Rehearsal-free Routing in MoE MLLMs

To support a more modular and privacy-preserving approach to visual instruction tuning, we propose DISTMOE, based on MoE architecture. The key idea behind DISTMOE is to train the experts on private data in a novel way, so that during inference the router does not need any additional training on the combined private data. We split the training phase in two stages: (i) In Stage-I (see Sec. 4.1.1) we acquire domain-specific knowledge by training a "two-expert" MoE network, where we initialize each private expert $\mathcal{E}_m$ from a public FFN $\mathcal{E}_{\text{pub}}$ trained on generic public data. (ii) In Stage-II (see Sec. 4.1.2) we calibrate routing of tokens, so that during inference, the experts can simply be merged to behave like a standard MoE router trained on all the domain-specific data jointly. Fig. 2 and Algorithm 1 in Appendix A summarizes the resulting training and inference procedure.

4.1 Training

Standard MoE training, as in MoE-LLaVA, trains experts on domain-specific datasets jointly. In the distributed setup, this is infeasible, and training experts in isolation can cause them to diverge from each other, making merging during inference difficult. While FlexOlmo proposes to use the public $\mathcal{E}_{\text{pub}}$ as an *anchor* to induce a coordination effect among private experts, we observe that private experts can still drift from the anchor, and thus from one another. To address this, as shown in Fig. 2, DISTMOE separates *expert knowledge acquisition* (i.e., Stage-I) from *expert composition* (i.e., Stage-II). Unless otherwise stated, private experts are trained within each layer of the decoder.

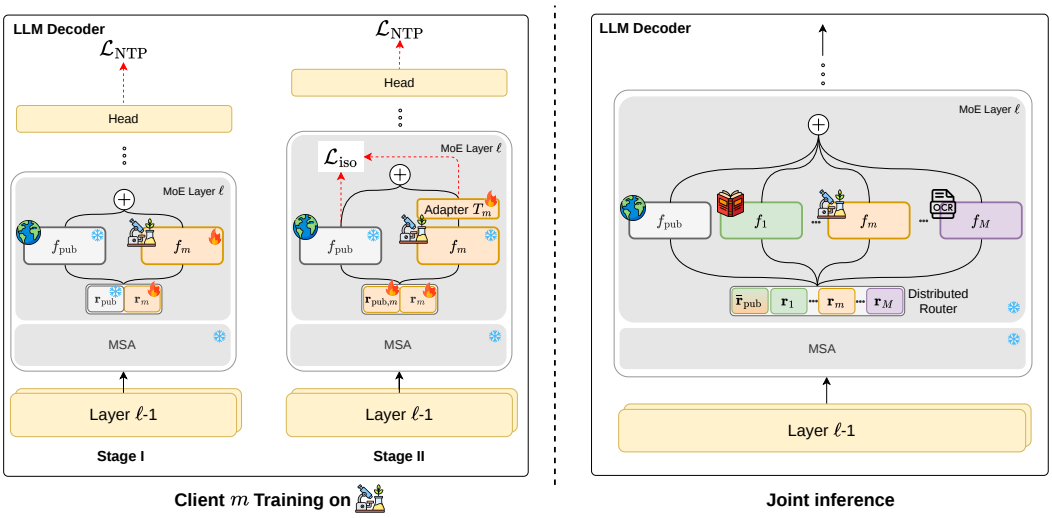


Figure 2: Overview of DISTMOE. **Left**. In Stage-I each client independently trains a private FFN by initializing from a shared public FFN. In Stage-II we freeze the experts, and recalibrate routing using isotropic residual regularization and next-token prediction loss utilizing a mix of public and private data. **Right.** At inference, DISTMOE composes the shared public expert and calibrated private experts through sparse routing without explicit domain labels.

4.1.1 Stage-I: Expert knowledge acquisition

Following FlexOlmo, we dedicate Stage-I to accomplish two goals: (i) acquire domain-specific knowledge on a private dataset $\mathcal{D}_m$, and (ii) prepare the experts to coordinate with each other. More specifically, we construct a MoE model with two experts: a public FFN $\mathcal{E}_{\text{pub}}$ and a private FFN $\mathcal{E}_{\text{m}}$ initialized from pre-trained $\mathcal{E}_{\text{pub}}$. During training, we freeze the parameters of the attention layers and $\mathcal{E}_{\text{pub}}$, and update only the private expert and its corresponding router parameters using the next-token prediction loss $\mathcal{L}_{\text{NTP}}$ on $\mathcal{D}_m$. As a result, for a token with hidden state $\mathbf{x}$, the router learns to predict the probability of being assigned to the public or private expert. The MoE layer output is given as:

$$\mathbf{o} = \alpha_{\text{pub}}\mathcal{E}_{\text{pub}}(\mathbf{x}) + \alpha_m\mathcal{E}_m(\mathbf{x}), \tag{4}$$

where α_{pub} and α_m are the probabilities of being assigned to the public and private experts, respectively. Intuitively, the public expert $\mathcal{E}_{\text{pub}}$ serves as a shared anchor for all experts, and the two-way router learns a mechanism that enables their integration into a unified MoE architecture.

While Stage-I leads to knowledge acquisition, empirically, we found that this approach introduced by FlexOlmo still falls short of producing optimal routers for joint inference across all domains. We hypothesize that this occurs for two reasons. *First*, the public router is never trained on public data, and thus, cannot sufficiently distinguish between information sources. *Second*, Eq. (4) cannot guarantee that the outputs $\mathbf{o}$ from different clients will intrinsically have compatible magnitudes or directions. As a result of this, it makes the private router scores across clients not comparable.

4.1.2 Stage-II: Expert composition

In Stage-II we address the mismatch of router scores across clients. While a straightforward approach is to collect a small subset of private data from each client to fine-tune the merged router [12, 10], we do so *without* collecting private data from other clients. We introduce a lightweight adapter $T_m(\cdot)$ on top of the private expert $\mathcal{E}_m$ with the aim to encourage the output of the private expert to be comparable with that of the public expert $\mathcal{E}_{\text{pub}}$. We intentionally choose the adapter to be *linear* because here the goal is not to acquire new knowledge. In addition, we unlock the parameters of the two-way router in a client m to recalibrate the routing between the public and private expert. All other parameters of the model are kept frozen, making this stage computationally lighter than Stage-I.

Let us denote $\mathbf{h}_{\text{pub}} = \mathcal{E}_{\text{pub}}(\mathbf{x})$ and $\widetilde{\mathbf{h}}_m = T_m(\mathcal{E}_m(\mathbf{x}))$ be the representation of the public expert and the projected representation of the private expert, respectively. We define the *client residual* as:

$$\mathbf{\Delta}_m(\mathbf{x}) = \widetilde{\mathbf{h}}_m - \mathbf{h}_{\text{pub}}. \tag{5}$$

The client residual measures how a private expert output deviates from the shared public expert after linear projection. Ideally, we want the residuals to have a comparable scale across clients. Inspired by LeJEPA [15] that imposes isotropic Gaussian structure on learned representations, we impose a common residual distribution across clients by forcing the residual in each client to follow an isotropic Gaussian distribution. This leads the private experts trained on different datasets produce outputs with comparable residual scale around the same shared public expert. Formally, let $\mathcal{B}_m$ denote the token-level residuals in a mini-batch of client m. For sampled unit directions $\{\mathbf{a}_s\}_{s=1}^{S}$, we define

$$\mathcal{L}_{\text{iso}} = \frac{1}{S}\sum_{s=1}^{S} \text{EP}\left(\{\mathbf{a}_s^\top \boldsymbol{\Delta}_m(\mathbf{x}) : \mathbf{x} \in \mathcal{B}_m\}, \mathcal{N}(0,1)\right), \tag{6}$$

where $\text{EP}(\cdot, \mathcal{N}(0,1))$ is the Epps–Pulley discrepancy between the projected residuals and a standard Gaussian[1]. Because all clients are calibrated against the same public expert and isotropic Gaussian target, DISTMOE makes the private experts more comparable for distributed MoE composition.

In addition to imposing Gaussian structure on the residuals, we recalibrate the router by fine-tuning the router parameters corresponding to both the public and private expert. As we formalize later in Proposition 1, this is crucial because inference-time expert influence is controlled by both the client-residual magnitude and recalibrated routing scores. Formally, the Stage-II objective is

$$\mathcal{L}_{\text{Stage-II}} = \mathcal{L}_{\text{NTP}} + \lambda \mathcal{L}_{\text{iso}}, \tag{7}$$

where both $\mathcal{L}_{\text{NTP}}$ and $\mathcal{L}_{\text{iso}}$ are optimized on a mixed dataset $\mathcal{D}_m^{\text{mix}} = \mathcal{D}_m \cup \mathcal{D}_{\text{pub}}$, where $\mathcal{D}_{\text{pub}}$ is small subset of the publicly available dataset used for training the public model; and λ is a hyperparameter for balancing the two losses. Since, $\mathcal{D}_m^{\text{mix}}$ contains no private data from other clients, expert composition is cross-client rehearsal-free, unlike [12, 10, 13] which fine-tune the merged router on pooled private data. Each client reuses only its own local data $\mathcal{D}_m$. Moreover, $\mathcal{D}_{\text{pub}}$ is optional; with $\mathcal{D}_{\text{pub}} = \emptyset$, Stage-II calibrates on $\mathcal{D}_m$ alone (Fig. 4).

4.2 Inference

During inference, we employ a standard MoE approach to combine the expert models $\mathcal{E}_m$ along with the public expert $\mathcal{E}_{\text{pub}}$. After Stage-II, each client m contributes a calibrated private expert $\mathcal{E}_m$, router weights $\mathbf{r}_m$ and bias b_m, and a linear adapter $T_m(\cdot)$, while the shared public expert remains common across clients. Note that as the public expert is shared among clients, each client possesses their own public router weights $\mathbf{r}_{\text{pub},m}$. Therefore, we average the public-route parameters learned during Stage-II as $\bar{\mathbf{r}}_{\text{pub}} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{r}_{\text{pub},m}$. However, since the softmax in Eq. (2) is invariant to a common shift of its logits, each client's public logit is only defined up to a client-specific offset (a *gauge*), and averaging $\mathbf{r}_{\text{pub},m}$ does not in general preserve the public–private balance learned by each client. We therefore also introduce a gauge-fixed composition rule that only changes inference. The gauge-invariant quantity is the private-vs-public log-odds $\ell_m(\mathbf{x}) = g_m(\mathbf{x}) - g_{\text{pub},m}(\mathbf{x}) = \log(\alpha_m/\alpha_{\text{pub}})$ learned by the two-way router of client m, where $g_{\text{pub},m}$ and g_m are its public and private logits. We fix the gauge by anchoring the public logit at zero,

$$z_{\text{pub}}(\mathbf{x}) = 0, \qquad z_m(\mathbf{x}) = \ell_m(\mathbf{x}), \quad m = 1, \ldots, M, \tag{8}$$

followed by the same top-K softmax as in Eq. (2), which requires neither retraining nor additional data. For FlexOlmo, the public router is identical across clients, so Eq. (8) subtracts a common term from all logits and leaves its routing unchanged. We evaluate this rule in Sec. 5.2. As our method is based on the MoE approach, it also supports different types of routing during inference, such as sparse top-K routing (as noted in Eq. (3)) and soft routing involving all experts.

4.3 Discussion

Next, we discuss why Stage-II can alleviate the issue highlighted in Sec. 4.1.1. Following the MoE routing rule in Eq. (2), we denote by α_{pub} and $\{\alpha_m\}_{m=1}^{M}$ the routing weights assigned to the public expert and the M private experts, respectively. Under sparse top-K routing, non-selected routes are assigned zero weight. The routed output is

$$\mathbf{o} = \alpha_{\text{pub}}\mathbf{h}_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \widetilde{\mathbf{h}}_m. \tag{9}$$

[1]Given $\{u_i\}_{i=1}^{B}$, $\text{EP} = B\int |\widehat{\varphi}(t) - \varphi_0(t)|^2 \varphi_0(t)\, dt$, where $\widehat{\varphi}(t) = \frac{1}{B}\sum_{i=1}^{B} \exp(\mathrm{i}tu_i)$ and $\varphi_0(t) = \exp(-t^2/2)$.

Using the client residual $\mathbf{\Delta}_m = \widetilde{\mathbf{h}}_m - \mathbf{h}_{\text{pub}}$, and the normalization $\alpha_{\text{pub}} + \sum_{m=1}^{M} \alpha_m = 1$, Eq. (9) can be rewritten as

$$\mathbf{o} = \mathbf{h}_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \mathbf{\Delta}_m. \tag{10}$$

The algebraic derivation of Eq. (10) is provided in Appendix B. This equation shows why Stage-II regularizes the client residuals rather than the full expert outputs. At inference, the private experts do not contribute as isolated outputs; they contribute through their residuals relative to the shared public expert. Therefore, the effect of expert m is determined by two factors: the routing weight α_m, which is learned by the router, and the residual $\mathbf{\Delta}_m$, which determines the direction and magnitude of the representation shift introduced by that expert. If two clients receive similar routing weights but produce residuals with very different norms, the larger-residual expert can dominate the mixture regardless of whether it is semantically more relevant.

Proposition 1. *For the routed output in Eq.* (10)*, the deviation from the shared public expert is bounded by the routing-weighted residual magnitudes:*

$$\|\mathbf{o} - \mathbf{h}_{\text{pub}}\|_2 \leq \sum_{m=1}^{M} \alpha_m \|\mathbf{\Delta}_m\|_2 . \tag{11}$$

The proof follows from Eq. (10), the triangle inequality, and the non-negativity of the routing weights; the full proof is given in Appendix B. Proposition 1 motivates calibrating both factors: the adapter and $\mathcal{L}_i so$ act on $\mathbf{\Delta}_m$, the router on α_m.

The router learns how much mass to assign to each private expert, but the actual representation shift also depends on $\|\mathbf{\Delta}_m\|_2$. If different clients produce residuals with incompatible scales, the mixture can be influenced by representation magnitude rather than by the learned routing weights alone. By regularizing client residuals around the same public expert and the same isotropic Gaussian target, Stage-II makes these residuals more comparable while preserving client specialization. This supports cross-client rehearsal-free composition of independently trained experts in the final distributed MoE model.

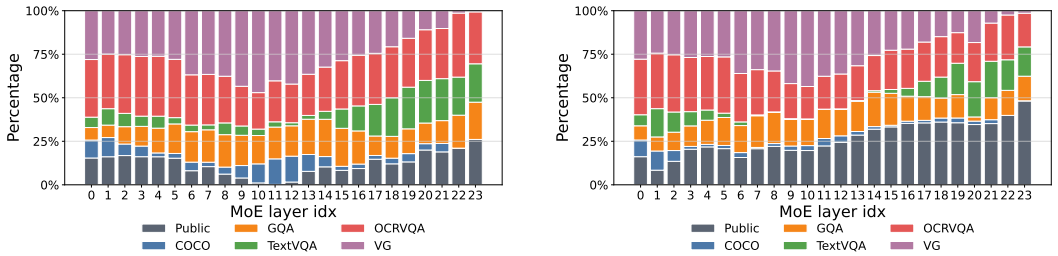
5 Experimental Results

Experimental setup. We train DISTMOE with five client domains: COCO [34], GQA [35], OCRVQA [36], VQA^{T}: TextVQA [37], and Visual Genome (VG) [38], treating each dataset as a separate private client. We initialize the shared public model from the MoE-LLaVA stage-2 [1] checkpoint and keep the vision encoder and multimodal connector frozen. In Stage-II, each client calibrates routers/adapters using its private data plus a fixed 19k public subset; this subset is a design choice rather than a requirement (Fig. 4). We instantiate DISTMOE with Qwen-1.8B [39], and Phi-2.7B [40], and StableLM-1.6B [41] language backbones; all variants use CLIP ViT-L/14 [17] as the frozen vision encoder. We evaluate on image QA benchmarks, including VQA^{v2} [42], GQA, SQA^{I}: ScienceQA-IMG [43], and VQA^{T}, as well as general MLLM benchmarks, including POPE [44], MME [45], SEED-Bench [46], LLaVA^{W}: LLaVA-Bench (in-the-Wild) [16], and MM-Vet [47]. This evaluation covers both in-domain performance on clients' training datasets and out-of-domain generalization on different multimodal benchmarks. Additional implementation details are provided in Appendix C.

Compared methods. The Public model denotes the MoE-LLaVA stage-2 checkpoint used to initialize the MLLM before client-specific expert adaptation. Offline reference models, such as Qwen-VL, LLaVA-Phi, and MoE-LLaVA, are included as centralized reference points. They represent the upperbound of our distributed approach. We report two MoE-LLaVA references: the original results reported by MoE-LLaVA [1] and a variant trained with joint access to all client datasets. We compare DISTMOE against Model Soup, weighted Model Soup, BTM, and BTX using their standard model-merging or expert-composition procedures. We also report the FlexOlmo-style routing baselines, denoted as FlexOlmo-LoRA and FlexOlmo, which correspond to Stage-I expert acquisition without the Stage-II public-anchored router and adapter calibration. Following [12], we additionally report FlexOlmo-RT, which fine-tunes the merged router after Stage-I using 10k private samples per client dataset plus the same 19k public subset used in our Stage-II. This rehearsal-based baseline serves as an upper-bound reference rather than a cross-client rehearsal-free composition distributed baseline. Finally, we include two variants of DISTMOE: a variant that adapts one private FFN expert per client, and a parameter-efficient LoRA variant that replaces each full private FFN with multiple lightweight

Table 1: Main results for DISTMOE with Qwen-1.8B and Phi-2.7B. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the model trained with joint access to the datasets used in our distributed setting. FlexOlmo-RT denotes FlexOlmo with Router Training [12]. Results are from a single seed run; three-seed results are reported in Tab. 15.

LLM	Method	Image Question Answering				Benchmark Toolkit				
		VQA^{v2}	GQA	SQA^{I}	VQA^{T}	POPE	MME	SEED	LLaVA^{W}	MM-Vet
Qwen-1.8B	***Offline reference models (upperbound)***									
	Qwen-VL	78.80	59.30	67.10	63.80	–	–	–	–	–
	MoE-LLaVA†	76.20	61.50	63.10	48.00	87.00	1291.60	–	88.70	25.30
	MoE-LLaVA	72.81	57.29	63.22	44.15	89.32	1230.16	57.08	62.20	15.60
	FlexOlmo-RT	72.36	56.66	62.17	45.19	87.85	1205.85	56.51	64.00	19.70
	Public model	20.96	13.75	58.50	17.72	71.17	402.54	51.17	71.00	18.60
	Model Soup (avg)	68.92	53.18	59.98	40.83	84.55	1154.80	53.82	50.40	14.60
	Model Soup (wtd)	69.18	53.46	60.21	41.18	84.84	1160.35	54.06	51.10	14.70
	BTM	69.73	53.81	60.66	41.87	85.46	1167.92	54.31	52.00	14.80
	BTX	69.46	54.16	60.43	41.52	85.15	1174.63	54.63	52.90	14.90
	FlexOlmo-LoRA	63.45	50.28	59.15	34.46	85.35	938.50	53.20	72.00	21.40
	FlexOlmo	70.01	54.26	60.85	41.70	85.98	1176.35	54.91	52.20	15.00
	DISTMOE-LoRA	65.80	50.51	59.30	33.69	85.68	995.61	53.34	**73.50**	**24.40**
	DISTMOE	**70.27**	**55.35**	**61.03**	**42.66**	**86.38**	**1188.56**	**55.11**	59.90	15.10
Phi-2.7B	***Offline reference models (upperbound)***									
	LLaVA-Phi	71.40	–	68.40	48.60	85.00	1335.10	–	–	28.90
	MoE-LLaVA†	77.60	61.40	68.50	51.40	85.70	1423.00	–	94.10	34.30
	MoE-LLaVA	74.99	56.69	68.37	46.52	85.65	1331.75	60.84	76.20	20.20
	Public model	42.87	32.80	69.26	31.82	80.48	862.53	59.36	73.30	23.20
	Model Soup (avg)	64.89	51.67	59.28	39.57	74.17	1054.37	53.38	42.40	15.70
	Model Soup (wtd)	65.92	52.34	59.89	39.76	74.43	1067.26	54.05	43.20	14.40
	BTM	66.53	52.56	60.56	40.76	72.80	1106.64	52.82	41.70	16.30
	BTX	67.80	52.67	60.24	40.53	73.62	1096.48	51.43	44.40	17.00
	FlexOlmo-LoRA	42.87	32.78	69.41	31.74	80.53	862.53	59.39	72.10	22.30
	FlexOlmo	70.03	54.84	62.77	41.63	74.98	1193.71	58.79	52.00	18.30
	DISTMOE-LoRA	42.89	32.84	**69.52**	31.77	**80.84**	862.60	59.38	**75.30**	**23.80**
	DISTMOE	**71.31**	**56.53**	66.34	**41.87**	80.50	**1291.41**	**59.96**	55.90	19.80



(a) FlexOlmo (perception score: 1176.35). (b) DISTMOE (perception score: 1188.56).

Figure 3: Top-4 routing mass on MME with Qwen-1.8B, computed as the aggregate normalized router weight assigned to each expert. Compared with FlexOlmo, DISTMOE assigns more routing mass to the public expert while activating private experts, supporting the effect of Stage-II.

LoRA experts. For the LoRA variants, we use $R = 16$ LoRA experts per client with rank $r = 16$ and scaling $\alpha = 32$. Unless otherwise stated, we use top-k routing with $k = 4R$ for LoRA variants and $k = 4$ for all other distributed methods; the effect of varying R is analyzed in Appendix F.

5.1 Results

Tab. 1 shows that DISTMOE achieves strong performance across both image question answering and instruction-following MLLM benchmarks. Across Qwen-1.8B and Phi-2.7B, DISTMOE generally provides the strongest results on image QA benchmarks, indicating that public-anchored routing and residual calibration improve the reuse of independently trained visual-language experts without centralized training. DISTMOE-LoRA is also highly competitive despite its lightweight adaptation,

Table 2: Effect of Stage-II components in DISTMOE. Stage-I is included as a reference.

Stage	$T(\cdot)$	$\mathcal{L}_{\text{iso}}$	b_m	GQA	SQA	VQA^{T}	POPE
I	–	–	–	54.26	60.85	41.70	85.98
II	✗	✗	✗	54.21	60.42	41.71	85.54
II	✓	✗	✗	54.39	60.55	41.88	85.67
II	✓	✓	✗	54.62	60.71	42.08	85.82
II	✗	✗	✓	54.93	60.96	42.52	86.08
II	✓	✗	✓	55.16	60.88	42.35	86.24
II	✓	✓	✓	**55.35**	**61.03**	**42.66**	**86.38**

Table 3: Effect of removing individual experts from DISTMOE.

Expert	GQA	SQA	VQA^{T}	POPE
Public model	13.75	58.50	17.72	71.17
w/o COCO	54.51	60.29	41.14	**86.93**
w/o GQA	43.37	61.53	34.82	85.95
w/o OCRVQA	**56.28**	58.55	40.66	86.80
w/o VG	56.23	59.94	41.81	86.14
w/o TextVQA	55.69	**62.42**	**45.19**	84.37
DISTMOE	55.35	61.03	42.66	86.38

and performs especially well on instruction-following benchmarks such as LLaVA^{W} and MM-Vet. FlexOlmo-RT improves over rehearsal-free FlexOlmo by fine-tuning the merged router with pooled private rehearsal data, confirming the value of cross-client router supervision but violating our cross-client rehearsal-free composition setting. Compared with rehearsal-free FlexOlmo, DISTMOE benefits from Stage-II public-private calibration, suggesting that flexible expert composition for MLLMs requires calibration mechanisms tailored to multimodal representations rather than a direct transfer of text-only routing. Fig. 3 further illustrates this behavior. FlexOlmo assigns relatively less routing mass to the public expert and relies more heavily on private experts, which can favor client-specific specialization but weakens the shared public expert that carries broadly trained visual-language knowledge. In contrast, DISTMOE assigns substantially more mass to the public expert, especially in later layers, while still activating private experts when useful. This behavior is consistent with the residual view of DISTMOE: private experts contribute calibrated shifts around a shared public anchor, rather than replacing the public expert. As a result, DISTMOE better preserves out-of-domain and general instruction-following capability while retaining client-specific expert reuse.
Model merging baselines such as Model Soup, BTM, and BTX remain competitive on some QA tasks, but their static parameter-level combination lacks token-wise expert selection and modular control over client-specific knowledge. Offline reference models provide upperbound comparisons, but they assume centralized training or joint access to data and therefore do not address the distributed, cross-client rehearsal-free composition setting targeted by DISTMOE.

Appendices D and G report additional benchmark results and routing visualizations.

5.2 Ablation Studies

Model components. Tab. 2 ablates the Stage-II components of DISTMOE (Qwen-1.8B): the client-private adapter $T(\cdot)$, isotropic residual loss $\mathcal{L}_{\text{iso}}$, and client-private bias b_m. The Stage-I row is the reference before Stage-II calibration; Stage-II without these components does not improve over it, showing that mixed-data router calibration alone is insufficient. The full configuration performs best across benchmarks, showing that residual alignment and router calibration are complementary.

Effect of public-data anchoring. Fig. 4 varies the number of public samples used in Stage-II (Qwen-1.8B). Without public data, routers and adapters are calibrated only on private data, which reduces compatibility with the shared MLLM. Adding public samples improves or stabilizes performance, and the gains quickly saturate, showing that a small public subset is sufficient for public-private calibration. In DISTMOE Stage-II, we used 19k public samples.

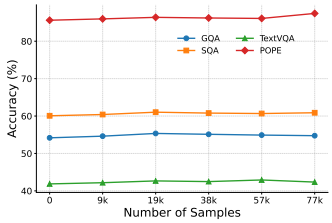


Figure 4: Public-data size in Stage-II.

Top-k expert selection. We study the effect of the sparse routing budget by varying the number of selected experts in DISTMOE (Qwen-1.8B) in Fig. 5. Smaller values of k encourage stronger sparsity and lower inference cost, but may underuse useful domain experts. Larger values allow more expert composition, but can introduce unnecessary computation and dilute specialization. The results show that a moderate top-k routing budget provides the best trade-off, enabling DISTMOE to combine relevant private experts while maintaining efficient inference.

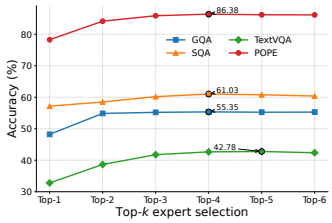


Figure 5: Top-k routing budget.

Table 4: Residual diagnostics on Phi-2.7B computed on the 19k public inputs. Metrics are defined in Appendix E. Last row: mean relative change over three seeds.

Condition	RMS-CV↓	Max/min↓	Eff. rank↑	SWD↓
Stage-I	0.1812	1.577	0.234	0.1012
Stage-II w/o $\mathcal{L}_{\text{iso}}$	0.0094	1.022	0.650	0.1587
Stage-II w/ $\mathcal{L}_{\text{iso}}$	0.0092	1.023	0.652	0.1550
Rel. change (3 seeds)	−5.30%	−0.10%	+0.25%	−1.76%

Table 5: Router composition on Phi-2.7B: averaged vs. gauge-fixed (Eq. (8)), without retraining.

Method (composition)	VQA^{T}	POPE
FlexOlmo	41.63	74.98
DISTMOE (averaged)	41.87	80.50
DISTMOE (gauge-fixed)	41.34	82.19

Table 6: Inference profiling on POPE with top-k routing on the same hardware: latency (s/sample), compute (TFLOPs/sample), peak memory (GB), and trainable parameters per client (B).

		Full FFN experts				LoRA experts			
LLM	Method	Latency	TFLOPs	Memory	Params	Latency	TFLOPs	Memory	Params
Qwen-1.8B	FlexOlmo	0.120	15.02	13.21	0.81	0.893	177.65	37.80	0.14
	DISTMOE	0.126	16.29	14.21	0.91	0.945	188.94	38.68	0.24
Phi-2.7B	FlexOlmo	0.205	28.99	23.53	1.68	1.792	304.53	59.27	0.21
	DISTMOE	0.220	31.65	25.63	1.89	1.864	314.83	62.32	0.42
StableLM-1.6B	FlexOlmo	0.119	15.04	12.87	0.83	1.541	268.39	35.16	0.14
	DISTMOE	0.253	57.37	14.89	0.93	1.625	274.03	36.78	0.24

Expert flexibility. Tab. 3 removes one client expert at a time during inference. All settings remain stronger than the public model, showing that DISTMOE does not rely on any single private expert. The impact is domain-dependent: removing GQA hurts GQA most, while removing some experts can improve certain metrics, suggesting reduced negative transfer. Overall, DISTMOE offers a robust performance trade-off, maintaining balanced results even when individual client experts are removed.

Residual geometry. Tab. 4 confirms the mismatch predicted by Proposition 1: Stage-I experts drift along their own low-dimensional subspaces with mismatched scales. Anchoring all adapters to the shared public expert removes most of it, while $\mathcal{L}_{\text{iso}}$ refines the *shape* rather than the scale of the residuals, hence its modest downstream effect (Appendix E).

Gauge-fixed router composition. Averaging public routes recalibrated by different clients mixes logits in different gauges and distorts each client's public–private balance; composing per-client log-odds (Eq. (8)) restores it and improves POPE without retraining (Tab. 5).

Computational cost. Extra parameters over FlexOlmo come from one $d\times d$ adapter per layer (Tab. 6). Adapters are applied only to the top-k selected experts, so activated compute is set by k, while storage and router scoring grow with M. DISTMOE does not reduce latency or memory over FlexOlmo.

6 Conclusion

We presented DISTMOE, a cross-client rehearsal-free MoE framework for distributed visual instruction tuning. DISTMOE keeps a shared public MLLM as an anchor, learns client-specific private experts locally, and makes them directly comparable through a Stage-II calibration without using any data across clients. At inference, it composes public and private experts via sparse routing. Across diverse vision-language benchmarks, DISTMOE improves over distributed merging and routing baselines, while DISTMOE-LoRA provides a competitive parameter-efficient alternative. These results show that public-anchored calibration can make independently trained multimodal experts reusable under distributed constraints, offering a practical path toward scalable MLLMs.

Limitations and future work. Centralized or rehearsal-based MoE training can still perform better on some benchmarks, reflecting the trade-off between unrestricted joint optimization and rehearsal-free distributed adaptation. DISTMOE calibrate Stage-II with a small public subset, which may not be available for open-weight models [48], motivating training-free expert composition.

Acknowledgments and Disclosure of Funding

This project has received funding from the European Union's EU Framework Programme for Research and Innovation under the HORIZON-MSCA-DN-2023 Grant Agreement N. 101169439. The work is also supported by the EU Horizon Europe projects ELIAS (No. 101120237) and ELLIOT (No. 101214398), and the FIS project GUIDANCE (No. FIS2023-03251) and Ministero delle Imprese e del Made in Italy (IPCEI Cloud DM 27 giugno 2022 - IPCEI-CL-0000007). We also acknowledge CINECA for providing access to HPC resources on the Leonardo supercomputer through the ISCRA initiative.

References

[1] Bin Lin, Zhenyu Tang, Yang Ye, Jinfa Huang, Junwu Zhang, Yatian Pang, Peng Jin, Munan Ning, Jiebo Luo, and Li Yuan. Moe-llava: Mixture of experts for large vision-language models. *IEEE Transactions on Multimedia*, 2026.

[2] Davide Caffagni, Federico Cocchi, Luca Barsellotti, Nicholas Moratelli, Sara Sarto, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. The revolution of multimodal large language models: A survey. *Findings of the association for computational linguistics: ACL 2024*, pages 13590–13618, 2024.

[3] 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*, pages 24185–24198, 2024.

[4] Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. Llava-interleave: Tackling multi-image, video, and 3d in large multimodal models. In *International Conference on Learning Representations*, volume 2025, pages 81182–81199, 2025.

[5] Bo Zhao, Boya Wu, Muyang He, and Tiejun Huang. Svit: Scaling up visual instruction tuning. *arXiv preprint arXiv:2307.04087*, 2023.

[6] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*, pages 370–387. Springer, 2024.

[7] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

[8] Sheng Shen, Zhewei Yao, Chunyuan Li, Trevor Darrell, Kurt Keutzer, and Yuxiong He. Scaling vision-language models with sparse mixture of experts. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11329–11344, 2023.

[9] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[10] Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Roziere, Jacob Kahn, Shang-Wen Li, Wen tau Yih, Jason E Weston, and Xian Li. Branch-train-mix: Mixing expert LLMs into a mixture-of-experts LLM. In *First Conference on Language Modeling*, 2024.

[11] Hejie Cui, Lingjun Mao, Xin Liang, Jieyu Zhang, Hui Ren, Quanzheng Li, Xiang Li, and Carl Yang. Biomedical visual instruction tuning with clinician preference alignment. *Advances in neural information processing systems*, 37:96449–96467, 2024.

[12] Weijia Shi, Akshita Bhagia, Kevin Farhat, Niklas Muennighoff, Jacob Morrison, Evan Walsh, Dustin Schwenk, Shayne Longpre, Jake Poznanski, Allyson Ettinger, et al. Flexolmo: Open language models for flexible data use. *Advances in Neural Information Processing Systems*, 38:165943–165974, 2026.

[13] Jacob Morrison, Sanjay Adhikesaven, Akshita Bhagia, Matei Zaharia, Noah A Smith, and Sewon Min. Train separately, merge together: Modular post-training with mixture-of-experts. *arXiv preprint arXiv:2604.18473*, 2026.

[14] Zhihan Zhou, Feng Hong, Jiaan Luo, Yushi Ye, Jiangchao Yao, Dongsheng Li, Bo Han, Ya Zhang, and Yanfeng Wang. Learning to instruct for visual instruction tuning. *Advances in Neural Information Processing Systems*, 38:47382–47409, 2026.

[15] Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

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

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[18] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[19] Cheng Chen, Junchen Zhu, Xu Luo, Heng T Shen, Jingkuan Song, and Lianli Gao. Coin: A benchmark of continual instruction tuning for multimodel large language models. *Advances in neural information processing systems*, 37:57817–57840, 2024.

[20] Jinpeng Chen, Runmin Cong, Yuzhi Zhao, Hongzheng Yang, Guangneng Hu, Horace Ip, and Sam Kwong. SEFE: Superficial and essential forgetting eliminator for multimodal continual instruction tuning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 7982–8001. PMLR, 13–19 Jul 2025.

[21] Chongyang Zhao, Mingsong Li, Haodong Lu, and Dong Gong. On token's dilemma: Dynamic moe with drift-aware token assignment for continual learning of large vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3941–3952, June 2026.

[22] Lingyun Song, Ziyao Chen, Kang Pan, Xiaolin Han, Xinbiao Gan, Yudai Pan, Xiaofan Sun, Xiaoqi Wang, and Xuequn Shang. Kss-moe: Knowledge space synergy framework in mixture of experts for continual visual instruction tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 25536–25544, 2026.

[23] Jinghan He, Haiyun Guo, Kuan Zhu, Ming Tang, and Jinqiao Wang. Continual instruction tuning for large multimodal models. *IEEE Transactions on Image Processing*, 2026.

[24] Didi Zhu, Zhongyisun Sun, Zexi Li, Tao Shen, Ke Yan, Shouhong Ding, Chao Wu, and Kun Kuang. Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. In *International Conference on Machine Learning*, pages 62581–62598. PMLR, 2024.

[25] Arthur Douillard, Qixuan Feng, Andrei A Rusu, Rachita Chhaparia, Yani Donchev, Adhiguna Kuncoro, Marc'Aurelio Ranzato, Arthur Szlam, and Jiajun Shen. Diloco: Distributed low-communication training of language models. *arXiv preprint arXiv:2311.08105*, 2023.

[26] Mavina Nikandrou, Lu Yu, Alessandro Suglia, Ioannis Konstas, and Verena Rieser. Task formulation matters when learning continually: A case study in visual question answering. *arXiv preprint arXiv:2210.00044*, 2022.

[27] Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications, and opportunities. *ACM Computing Surveys*, 58(8):1–41, 2026.

[28] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR, 2022.

[29] Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models. In *First Workshop on Interpolation Regularizers and Beyond at NeurIPS 2022*, 2022.

[30] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, 36:7093–7115, 2023.

[31] Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

[32] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

[33] Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. Multimodal contrastive learning with limoe: the language-image mixture of experts. *Advances in Neural Information Processing Systems*, 35:9564–9576, 2022.

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[35] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

[36] Anand Mishra, Shashank Shekhar, Ajeet Kumar Singh, and Anirban Chakraborty. Ocr-vqa: Visual question answering by reading text in images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 947–952, 2019.

[37] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326, 2019.

[38] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

[39] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[40] Mojan Javaheripi and Sébastien Bubeck. Phi-2: The surprising power of small language models. `https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/`, December 2023. Microsoft Research Blog.

[41] Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, et al. Stable lm 2 1.6 b technical report. *arXiv preprint arXiv:2402.17834*, 2024.

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

[43] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in neural information processing systems*, 35:2507–2521, 2022.

[44] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 292–305, 2023.

[45] Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *Advances in Neural Information Processing Systems*, 38, 2026.

[46] Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. Seed-bench: Benchmarking multimodal large language models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13299–13308. IEEE, 2024.

[47] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. MM-vet: Evaluating large multimodal models for integrated capabilities. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 57730–57754. PMLR, 21–27 Jul 2024.

[48] Stephen Casper, Kyle O'Brien, Shayne Longpre, Elizabeth Seger, Kevin Klyman, Rishi Bommasani, Aniruddha Nrusimha, Ilia Shumailov, Sören Mindermann, Steven Basart, Frank Rudzicz, Kellin Pelrine, Avijit Ghosh, Andrew Strait, Robert Kirk, Dan Hendrycks, Peter Henderson, J Zico Kolter, Geoffrey Irving, Yarin Gal, Yoshua Bengio, and Dylan Hadfield-Menell. Open technical problems in open-weight AI model risk management. *Transactions on Machine Learning Research*, 2026. Survey Certification.

[49] Junke Wang, Lingchen Meng, Zejia Weng, Bo He, Zuxuan Wu, and Yu-Gang Jiang. To see is to believe: Prompting gpt-4v for better visual instruction tuning. *arXiv preprint arXiv:2311.07574*, 2023.

[50] Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. Mitigating hallucination in large multi-modal models via robust instruction tuning. In *International Conference on Learning Representations*, volume 2024, pages 57689–57733, 2024.

[51] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Mimic-it: Multi-modal in-context instruction tuning. *arXiv preprint arXiv:2306.05425*, 2023.

[52] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024.

[53] Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pages 606–610. IEEE, 2007.

[54] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015.

A Pseudocode Algorithm for DISTMOE

Algorithm 1 DISTMOE: Distributed Visual Instruction Tuning with Cross-client Rehearsal-free Composition

Require: Public MLLM backbone with public FFN layers $\{\mathcal{E}^{\ell}_{\text{pub}}\}_{\ell=1}^{L}$ in the LLM; private datasets $\{\mathcal{D}_m\}_{m=1}^{M}$; public data $\mathcal{D}_{\text{pub}}$.

Ensure: Client experts $\{\mathcal{E}^{\ell}_m\}$, routers $\{\mathbf{r}^{\ell}_{\text{pub},m}, \mathbf{r}^{\ell}_m, b^{\ell}_m\}$, and adapters $\{T^{\ell}_m\}$.

Freeze the vision encoder, multimodal projector, and public FFNs $\{\mathcal{E}^{\ell}_{\text{pub}}\}_{\ell=1}^{L}$.
for each client $m = 1, \dots, M$ **do**
  for each layer $\ell = 1, \dots, L$ **do**
    Initialize $\mathcal{E}^{\ell}_m \leftarrow \mathcal{E}^{\ell}_{\text{pub}}$.
    Initialize $\mathbf{r}^{\ell}_{\text{pub},m}, \mathbf{r}^{\ell}_m \leftarrow \mathbf{0}$ and $b^{\ell}_m \leftarrow 0$.
  end for
  Stage-I: Expert knowledge acquisition
  while not converged **do**
    Sample a mini-batch $(I, q, y) \sim \mathcal{D}_m$.
    for each FFN layer ℓ and token hidden state $\mathbf{x}^{\ell}$ **do**
      $z_{\text{pub}} = (\mathbf{x}^{\ell})^{\top}\mathbf{r}^{\ell}_{\text{pub},m}, \quad z_m = (\mathbf{x}^{\ell})^{\top}\mathbf{r}^{\ell}_m + b^{\ell}_m.$
      $[\alpha^{\ell}_{\text{pub},m}, \alpha^{\ell}_m] = \text{softmax}([z_{\text{pub}}, z_m]).$
      $\mathbf{o}^{\ell} = \alpha^{\ell}_{\text{pub},m}\mathcal{E}^{\ell}_{\text{pub}}(\mathbf{x}^{\ell}) + \alpha^{\ell}_m\mathcal{E}^{\ell}_m(\mathbf{x}^{\ell}).$
    end for
    Update only $\{\mathcal{E}^{\ell}_m\}_{\ell=1}^{L}$ and the client routers by minimizing $\mathcal{L}_{\text{NTP}}$ on $\mathcal{D}_m$.
  end while
  Stage-II: Public-anchored composition calibration
  Freeze $\{\mathcal{E}^{\ell}_{\text{pub}}\}_{\ell=1}^{L}$ and $\{\mathcal{E}^{\ell}_m\}_{\ell=1}^{L}$.
  Construct $\mathcal{D}^{\text{mix}}_m = \mathcal{D}_m \cup \mathcal{D}_{\text{pub}}$.
  while not converged **do**
    Sample a mini-batch $(I, q, y) \sim \mathcal{D}^{\text{mix}}_m$.
    for each FFN layer ℓ and token hidden state $\mathbf{x}^{\ell}$ **do**
      $\mathbf{h}^{\ell}_{\text{pub}} = \mathcal{E}^{\ell}_{\text{pub}}(\mathbf{x}^{\ell}), \quad \widetilde{\mathbf{h}}^{\ell}_m = T^{\ell}_m(\mathcal{E}^{\ell}_m(\mathbf{x}^{\ell})).$
      $z_{\text{pub}} = (\mathbf{x}^{\ell})^{\top}\mathbf{r}^{\ell}_{\text{pub},m}, \quad z_m = (\mathbf{x}^{\ell})^{\top}\mathbf{r}^{\ell}_m + b^{\ell}_m.$
      $[\alpha^{\ell}_{\text{pub},m}, \alpha^{\ell}_m] = \text{softmax}([z_{\text{pub}}, z_m]).$
      $\mathbf{o}^{\ell} = \alpha^{\ell}_{\text{pub},m}\mathbf{h}^{\ell}_{\text{pub}} + \alpha^{\ell}_m\widetilde{\mathbf{h}}^{\ell}_m.$
      $\boldsymbol{\Delta}^{\ell}_m = \widetilde{\mathbf{h}}^{\ell}_m - \mathbf{h}^{\ell}_{\text{pub}}.$
    end for
    Compute $\mathcal{L}_{\text{iso}}$ from the residuals $\{\boldsymbol{\Delta}^{\ell}_m\}_{\ell=1}^{L}$.
    Update $\{\mathbf{r}^{\ell}_{\text{pub},m}, \mathbf{r}^{\ell}_m, b^{\ell}_m, T^{\ell}_m\}_{\ell=1}^{L}$ by minimizing $\mathcal{L}_{\text{Stage-II}} = \mathcal{L}_{\text{NTP}} + \lambda\mathcal{L}_{\text{iso}}$.
  end while
end for
Inference: Rehearsal-free modular routing
for each layer ℓ and token hidden state $\mathbf{x}^{\ell}$ **do**
  $\bar{\mathbf{r}}^{\ell}_{\text{pub}} = \frac{1}{M}\sum_{m=1}^{M}\mathbf{r}^{\ell}_{\text{pub},m}.$
  $z^{\ell}_{\text{pub}} = (\mathbf{x}^{\ell})^{\top}\bar{\mathbf{r}}^{\ell}_{\text{pub}}.$
  for each client expert $m = 1, \dots, M$ **do**
    $z^{\ell}_m = (\mathbf{x}^{\ell})^{\top}\mathbf{r}^{\ell}_m + b^{\ell}_m.$
  end for
  $\mathbf{z}^{\ell} = [z^{\ell}_{\text{pub}}, z^{\ell}_1, \dots, z^{\ell}_M].$
  Select active routes $\mathcal{S}^{\ell}$ from $\mathbf{z}^{\ell}$ using the inference routing policy (e.g. top-k routing).
  Normalize routing weights over $\mathcal{S}^{\ell}$ to obtain $\alpha^{\ell}_{\text{pub}}$ and $\{\alpha^{\ell}_m\}_{m=1}^{M}$.
  Assign zero weight to non-selected routes.
  $\mathbf{o}^{\ell} = \alpha^{\ell}_{\text{pub}}\mathcal{E}^{\ell}_{\text{pub}}(\mathbf{x}^{\ell}) + \sum_{m=1}^{M}\alpha^{\ell}_m T^{\ell}_m\left(\mathcal{E}^{\ell}_m(\mathbf{x}^{\ell})\right).$
end for

B Derivation of the Residual Influence Bound

For clarity, we omit the layer index and consider one token representation. The same derivation applies independently at every layer. At inference, routing is performed over $M + 1$ routes: one shared public route and M private routes. Top-K selection assigns zero weight to non-selected routes, so the normalized weights satisfy $\alpha_{\text{pub}} + \sum_{m=1}^{M} \alpha_m = 1$.

B.1 Mathematical derivation of Eq. (10)

The inference-time routed output is

$$\mathbf{o} = \alpha_{\text{pub}} \mathbf{h}_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \widetilde{\mathbf{h}}_m. \tag{12}$$

The routing weights are normalized over the selected routes, with non-selected routes assigned zero weight:

$$\alpha_{\text{pub}} + \sum_{m=1}^{M} \alpha_m = 1. \tag{13}$$

For each private expert m, the client residual is defined as

$$\boldsymbol{\Delta}_m = \widetilde{\mathbf{h}}_m - \mathbf{h}_{\text{pub}}. \tag{14}$$

Equivalently,

$$\widetilde{\mathbf{h}}_m = \mathbf{h}_{\text{pub}} + \boldsymbol{\Delta}_m. \tag{15}$$

Substituting this expression into the routed output gives

$$\mathbf{o} = \alpha_{\text{pub}} \mathbf{h}_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \left(\mathbf{h}_{\text{pub}} + \boldsymbol{\Delta}_m \right) \tag{16}$$

$$= \left(\alpha_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \right) \mathbf{h}_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \boldsymbol{\Delta}_m. \tag{17}$$

Using the normalization of the routing weights, we obtain

$$\mathbf{o} = \mathbf{h}_{\text{pub}} + \sum_{m=1}^{M} \alpha_m \boldsymbol{\Delta}_m, \tag{18}$$

which is Eq. (10).

B.2 Proof of Proposition 1

Proposition 1. *For the routed output in Eq.* (10)*, the deviation from the shared public expert is bounded by the routing-weighted residual magnitudes:*

$$\left\| \mathbf{o} - \mathbf{h}_{\text{pub}} \right\|_2 \leq \sum_{m=1}^{M} \alpha_m \left\| \boldsymbol{\Delta}_m \right\|_2 . \tag{19}$$

Proof. From Eq. (10), we have

$$\mathbf{o} - \mathbf{h}_{\text{pub}} = \sum_{m=1}^{M} \alpha_m \boldsymbol{\Delta}_m. \tag{20}$$

Taking the Euclidean norm on both sides gives

$$\left\| \mathbf{o} - \mathbf{h}_{\text{pub}} \right\|_2 = \left\| \sum_{m=1}^{M} \alpha_m \boldsymbol{\Delta}_m \right\|_2 . \tag{21}$$

Applying the triangle inequality,

$$\left\| \sum_{m=1}^{M} \alpha_m \mathbf{\Delta}_m \right\|_2 \leq \sum_{m=1}^{M} \|\alpha_m \mathbf{\Delta}_m\|_2 . \tag{22}$$

Since routing weights are non-negative, $\alpha_m \geq 0$, we have

$$\|\alpha_m \mathbf{\Delta}_m\|_2 = \alpha_m \|\mathbf{\Delta}_m\|_2 . \tag{23}$$

Therefore,

$$\|\mathbf{o} - \mathbf{h}_{\text{pub}}\|_2 \leq \sum_{m=1}^{M} \alpha_m \|\mathbf{\Delta}_m\|_2 , \tag{24}$$

which proves the proposition. □

C Additional Experimental Details

C.1 Dataset Details

We construct a distributed visual-instruction tuning setup with five private client domains: COCO [34], GQA [35], OCRVQA [36], VQA^{T} [37], and Visual Genome (VG) [38]. In Stage-I, each client is trained only on its own private dataset, without accessing data from other clients. In Stage-II, we keep the client-specific dataset for each client and add a small public subset for router and adapter calibration. The public subset contains 19k samples drawn from the Hybrid-FT data group used in MoE-LLaVA, including SViT-157k [5], LVIS-220k [49], LRV-331k [50] and MIMIC-IT-256k [51]. In addition, the public MLLM backbone i.e. Stage-2 MoE-LLaVA checkpoint is already pretrained on approximately 1.5M multimodal instruction data including LLaVA 1.5-558k [52] and the Hybrid-FT data group, while DISTMOE only performs distributed client-specific adaptation on top of this backbone. Tab. 7 summarizes the training data used for public model initialization, private and public data distribution. Below, we also discuss the test dataset along with the evaluation metrics used in our experiments.

C.2 Evaluation Metrics

Academic-task benchmark. In Tab. 9, 11, 13, we use the following multiple evaluation metrics of each dataset. (i) **VQA**$^{\text{v2}}$ [42] - Accuracy for *Yes/No*, *Number*, and *Other* questions, together with *Overall* accuracy, which evaluate binary recognition, numerical reasoning, and open-ended visual question answering respectively are reported. (ii) **GQA** [35] - We report *Binary*, *Open*, and overall accuracy (*Acc.*), along with the distribution score (*Dist.*). Binary and open accuracy measure closed-form and open-ended compositional reasoning, while lower *Dist.* indicates less answer-distribution bias. (iii) **SQA**: ScienceQA [43] - We use standard answer accuracy (*Acc.*) and image-grounded accuracy (*IMG-Acc.*). The latter focuses on questions requiring visual evidence rather than language priors. (iv) **VQA**^{T}: TextVQA [37] - We report answer accuracy, which evaluates the model's ability to read and reason over text appearing in images. However, in main Tab. 1, we showcase only overall accuracy for VQA$^{\text{v2}}$ GQA, and VQA^{T}, whereas image-grounded accuracy for SQA dataset.

Instruction-following benchmark. In Tab. 10, 12, 14, we use the following multiple evaluation metrics of each dataset. (i) **POPE** [44] - We report accuracy under *Random*, *Popular*, and *Adversarial* settings, together with their average, that evaluate object hallucination robustness under increasingly challenging negative samples. (ii) **MME** [45] - We report *Perception* and *Cognition* scores. Perception measures visual understanding such as recognition, counting, position, and OCR, while cognition measures higher-level multimodal reasoning. (iii) **SEED-Bench** [46] - We report *All*, *Image*, and *Video* scores, measuring overall performance, static-image understanding, and video-related reasoning. (iv) **LLaVA**^{W}: LLaVA-Bench (in-the-Wild) [16] - We report *All*, *Complex Reasoning* (*Comp.*), *Conversation* (*Conv.*), and *Detailed Description* (*Det.*) scores for open-ended multimodal instruction following. (v) **MM-Vet** [47] - We report overall accuracy, which evaluates integrated multimodal abilities such as recognition, OCR, spatial reasoning, knowledge reasoning, and language generation. However, in main Tab. 1, we report only average accuracy for POPE, perception score for MME, *All* for SEED-Bench and LLaVA^{W}, and overall accuracy for MM-Vet dataset.

Table 7: **Training data used in DISTMOE.** Stage-I trains each client expert using only its own private dataset. Stage-II uses the same client data together with a small public subset sampled from the Hybrid-FT data group.

Data group	Usage	Source	# Samples
Public backbone pretraining	Initialization	LLaVA-PT group: LLaVA 1.5-558k Hybrid-FT group: SViT-157k, LVIS-220k, LRV-331k, MIMIC-IT-256k	$\sim$1.5M
Private client data	Stage-I / II for $\mathcal{E}_1$	COCO	364k
	Stage-I / II for $\mathcal{E}_2$	GQA	72k
	Stage-I / II for $\mathcal{E}_3$	OCRVQA	80k
	Stage-I / II for $\mathcal{E}_4$	VQA^{T}	22k
	Stage-I / II for $\mathcal{E}_5$	Visual Genome (VG)	86k
Public data	Stage-II only	Subset of Hybrid-FT group: SViT-157k, LVIS-220k, LRV-331k, MIMIC-IT-256k	19k

Table 8: **Zero-shot object hallucination evaluation in POPE** dataset for DISTMOE with Qwen-1.8B, Phi-2.7B, and StableLM-1.6B. We report accuracy, F1-score, and the yes-answer ratio under Random, Popular, and Adversarial settings. "Yes" indicates the proportion of positive responses to the given question.

LLM	Method	Random			Popular			Adversarial		
		Acc. ↑	F1 ↑	Yes ↓	Acc. ↑	F1 ↑	Yes ↓	Acc. ↑	F1 ↑	Yes ↓
Qwen-1.8B	Public model	68.38	75.57	77.87	75.90	79.74	68.97	69.23	75.51	75.63
	FlexOlmo-LoRA	86.80	86.48	46.11	86.63	85.54	45.30	82.63	82.51	49.30
	FlexOlmo	86.84	85.75	**40.79**	86.26	84.83	**40.53**	**84.83**	82.99	**42.53**
	DISTMOE-LoRA	87.08	86.43	43.64	86.67	85.68	43.13	83.30	82.69	46.50
	DISTMOE	**88.18**	**87.51**	43.09	**86.73**	**85.83**	43.60	84.23	**83.59**	46.10
Phi-2.7B	Public model	83.54	84.89	57.42	79.37	81.30	60.37	78.53	80.69	61.20
	FlexOlmo-LoRA	83.54	84.85	57.42	79.43	81.35	60.30	78.63	80.77	61.10
	FlexOlmo	74.91	67.98	**26.80**	75.30	67.66	**26.37**	74.73	67.15	**26.93**
	DISTMOE-LoRA	**83.95**	**84.89**	57.43	79.82	**81.42**	60.37	78.76	**80.92**	61.34
	DISTMOE	80.86	77.39	33.09	**80.80**	76.79	32.73	**79.83**	75.91	33.70
StableLM-1.6B	Public model	58.76	71.32	92.23	53.80	68.28	95.67	53.37	68.08	96.10
	FlexOlmo-LoRA	75.62	70.37	**28.56**	77.45	70.15	**28.78**	76.04	70.83	**30.05**
	FlexOlmo	77.84	72.82	30.00	78.07	72.42	29.53	76.87	71.34	30.73
	DISTMOE-LoRA	84.06	82.45	37.96	83.92	81.23	37.65	81.45	78.19	38.20
	DISTMOE	**84.98**	**83.29**	38.32	**84.27**	**82.19**	38.33	**82.27**	**80.37**	40.33

C.3 Additional Implementation Details

We implement DISTMOE on three MoE-LLaVA-Stage2 backbones, using Qwen-1.8B, StableLM-1.6B, and Phi-2.7B as the LLMs, with CLIP ViT-L/14 as the shared vision encoder, in which the input image resolution is set to 336×336. For each backbone, we replace every LLM FFN with a routed public-private FFN, while keeping the vision encoder and multimodal projector frozen. In the full fine-tuning setting, each private expert is initialized as a full copy of the corresponding public FFN. In the LoRA setting, each client uses $R = 16$ LoRA experts on the linear projection matrices, with rank $r = 16$ and scaling factor $\alpha = 32$. Stage-II freezes all private experts and updates only the routers, router biases, and lightweight adapter $T_m(\cdot)$, which is taken as a linear layer of same input-output dimension of the FFN blocks of the corresponding LLMs. We use top-k routing with $k = 4$ for full experts and $k = 4R = 64$ for LoRA experts. For expert composition, we use $|\mathcal{D}_{\text{pub}}| = 19\text{k}$ public samples, and $\lambda = 10^{-4}$ in Eq. (7). All experiments are conducted on 4 NVIDIA A100 GPUs with 64GB memory each. We train for 1 epoch using AdamW with learning rate 2×10^{-5}, cosine learning-rate scheduling, warmup ratio 0.03, and per-device batch size 4 in both of the training stages.

Table 9: **Detailed academic-task benchmark results with Qwen-1.8B.** We report fine-grained results on VQA^{v2}, GQA, SQA, and VQA^{T}. The Public model denotes the shared backbone before client-specific expert composition. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the offline MoE model trained with joint access to the same client datasets used in our distributed setting.

Method	VQA^{v2}				GQA				SQA		VQA^{T}
	Yes/No ↑	Num. ↑	Other ↑	Overall ↑	Binary ↑	Open ↑	Acc. ↑	Dist. ↓	Acc. ↑	IMG-Acc. ↑	Acc. ↑
Offline reference models (upperbound)											
Qwen-VL	–	–	–	78.80	–	–	59.30	–	–	67.10	63.80
MoE-LLaVA†	–	–	–	76.20	–	–	61.50	–	–	63.10	48.00
MoE-LLaVA	88.26	53.46	63.10	72.81	73.56	44.41	57.29	2.07	64.37	63.22	44.15
Individual experts											
COCO	84.41	51.84	58.28	69.94	62.01	34.03	46.88	2.34	60.56	59.24	35.51
GQA	82.65	46.47	56.98	66.36	68.15	37.38	51.62	2.62	38.91	29.75	32.27
VQA^{OCR}	70.10	29.08	45.21	53.63	39.29	17.13	27.30	2.41	60.25	59.11	36.16
VG	0.01	0.16	6.00	2.87	0.12	0.84	0.51	4.23	41.22	37.83	16.62
VQA^{T}	4.27	1.88	39.09	20.52	9.28	16.31	13.09	2.54	56.87	58.75	0.88
Public model	5.65	1.76	38.85	20.96	10.03	16.91	13.75	2.45	55.34	58.50	17.72
Model Soup (avg)	84.92	50.61	59.18	68.92	69.54	39.52	53.18	2.35	60.52	59.98	40.83
Model Soup (wtd)	85.21	50.94	59.43	69.18	69.81	39.86	53.46	2.33	60.83	60.21	41.18
BTM	85.47	51.22	59.72	69.73	70.08	40.18	53.81	2.28	61.36	60.66	41.87
BTX	85.74	51.48	60.03	69.46	70.41	40.54	54.16	2.28	61.12	60.43	41.52
FlexOlmo-LoRA	74.39	45.25	52.94	63.45	61.65	40.63	50.28	1.85	59.33	59.15	34.46
FlexOlmo	85.15	51.67	60.04	70.01	70.32	40.50	54.26	2.26	61.68	60.85	41.70
DistMoE-LoRA	77.28	47.16	55.47	65.80	62.03	40.73	50.51	**1.79**	59.21	59.30	33.69
DistMoE	**86.38**	**52.04**	**60.64**	**70.27**	**71.25**	**41.87**	**55.35**	2.11	**61.94**	**61.03**	**42.66**

Table 10: **Detailed instruction-following benchmark results with Qwen-1.8B.** We report fine-grained results on POPE, MME, SEED-Bench, LLaVA^{W}, and MM-Vet. The Public model denotes the shared backbone before client-specific expert composition. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the offline MoE model trained with joint access to the same client datasets used in our distributed setting.

Method	POPE				MME		SEED-Bench			LLaVA^{W}				MM-Vet
	Rand. ↑	Pop. ↑	Adv. ↑	Avg. ↑	Perc. ↑	Cog. ↑	All. ↑	Img. ↑	Vid. ↑	All ↑	Comp. ↑	Conv. ↑	Det. ↑	Acc. ↑
Offline reference models (upperbound)														
MoE-LLaVA	90.48	88.82	88.67	89.32	1230.16	272.47	57.08	61.47	40.52	62.20	57.20	74.60	52.50	15.60
Individual experts														
COCO	86.49	85.93	82.97	85.13	1057.79	241.52	53.40	58.35	36.81	71.70	73.10	72.50	68.40	20.10
GQA	82.92	83.13	82.07	82.71	1109.63	252.50	14.76	11.76	26.14	58.60	51.90	64.80	63.30	18.10
VQA^{OCR}	84.47	79.07	77.67	80.40	1122.81	212.14	51.83	55.13	35.42	69.10	68.50	71.50	67.00	17.10
VG	51.58	50.00	50.03	50.54	1.34	2.86	36.09	36.61	34.12	44.70	44.80	50.80	35.60	18.30
VQA^{T}	69.28	73.83	67.80	70.30	455.94	190.36	52.23	56.25	37.02	64.30	65.90	67.20	57.70	18.20
Public model	68.38	75.90	69.23	71.17	402.54	176.79	51.17	55.14	36.15	71.00	70.30	79.30	62.50	18.60
Model Soup (avg)	85.31	85.14	83.21	84.55	1154.80	235.42	53.82	57.91	37.84	50.40	45.80	62.40	41.90	14.60
Model Soup (wtd)	85.67	85.39	83.46	84.84	1160.35	239.18	54.06	58.24	38.02	51.10	46.30	63.70	42.40	14.70
BTM	86.39	86.03	83.96	85.46	1167.92	243.61	54.31	58.55	38.21	52.00	46.70	64.90	42.90	14.80
BTX	86.02	85.72	83.71	85.15	1174.63	247.86	54.63	58.91	38.43	52.90	47.10	66.10	43.40	14.90
FlexOlmo-LoRA	86.80	86.63	82.63	85.35	938.50	251.07	53.27	57.35	37.82	72.00	73.00	75.20	66.30	21.40
FlexOlmo	86.84	86.26	**84.83**	85.98	1176.35	249.29	54.91	59.19	38.67	52.20	47.20	65.60	43.80	15.00
DistMoE-LoRA	87.08	86.67	83.30	85.68	995.61	231.43	53.34	57.30	38.33	**73.50**	**73.80**	**79.10**	**65.30**	**24.40**
DistMoE	**88.18**	**86.73**	84.23	**86.38**	**1188.56**	**259.29**	**55.11**	**59.38**	**38.94**	59.90	53.30	77.50	51.00	15.10

D Additional Experiments

This section provides the object hallucination evaluation results in Tab. 8 and the full per-benchmark breakdowns underlying the aggregate results in Tab. 1. Tabs. 9–12 report detailed results for Qwen-1.8B and Phi-2.7B, while Tabs. 13–14 provide the complete StableLM-1.6B results.

Object Hallucination Evaluation. We evaluate object hallucination using POPE, which formulates hallucination detection as a polling-based object-existence question answering task. Tab. 8 reports results under the *Random*, *Popular*, and *Adversarial* settings, together with F1-score and yes-answer ratio. Across the three backbones, DistMoE generally improves hallucination robustness over the corresponding public model and distributed baselines. In particular, the full DistMoE variant

Table 11: **Detailed academic-task benchmark results with Phi-2.7B.** We report fine-grained results on VQA^{v2}, GQA, SQA, and VQA^{T}. The Public model denotes the shared backbone before client-specific expert composition. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the offline MoE model trained with joint access to the same client datasets used in our distributed setting.

Method	VQA^{v2}				GQA				SQA		VQA^{T}
	Yes/No ↑	Num. ↑	Other ↑	Overall ↑	Binary ↑	Open ↑	Acc. ↑	Dist. ↓	Acc. ↑	IMG-Acc. ↑	Acc. ↑
Offline reference models (upper bound)											
LLaVA-Phi	–	–	–	71.40	–	–	–	–	–	68.40	48.60
MoE-LLaVA†	–	–	–	77.60	–	–	61.40	–	–	68.50	51.40
MoE-LLaVA	90.30	56.96	66.00	74.99	71.21	44.38	56.69	1.60	73.03	68.37	46.52
Individual experts											
COCO	84.73	47.33	58.96	65.33	62.83	36.86	48.78	2.33	63.43	59.26	36.49
GQA	85.02	48.50	59.68	68.26	67.81	40.99	51.68	2.28	39.99	27.66	32.30
VQA^{OCR}	71.46	30.57	46.14	51.46	52.40	27.70	39.04	2.15	63.17	59.21	30.67
VG	0.05	0.20	6.25	2.19	31.61	4.00	16.67	3.95	48.15	43.48	1.22
VQA^{T}	5.26	1.67	38.42	20.73	0.12	0.37	0.25	4.27	68.22	65.69	0.12
Public model	53.36	7.90	42.13	42.87	51.65	16.81	32.80	2.54	70.69	69.26	31.82
Model Soup (avg)	82.90	45.79	57.18	64.89	66.11	39.11	51.67	2.31	64.68	59.28	39.57
Model Soup (wtd)	83.04	46.28	58.67	65.92	67.26	39.95	52.34	2.29	65.83	59.89	39.76
BTM	83.98	47.07	59.28	66.53	67.54	40.07	52.56	2.27	66.21	60.56	40.76
BTX	83.56	47.29	59.34	67.80	67.82	40.16	52.67	2.26	65.98	60.24	40.53
FlexOlmo-LoRA	53.37	7.89	42.13	42.87	51.60	16.81	32.78	2.55	70.74	69.41	31.74
FlexOlmo	85.18	49.85	61.69	70.03	68.73	43.06	54.84	2.29	67.67	62.77	41.63
DistMoE-LoRA	53.39	7.91	42.14	42.89	51.67	16.86	32.84	2.55	**70.78**	**69.52**	31.77
DistMoE	**87.93**	**50.39**	**61.89**	**71.31**	**70.95**	**44.29**	**56.53**	**2.13**	69.32	66.34	**41.87**

Table 12: **Detailed instruction-following benchmark results with Phi-2.7B.** We report fine-grained results on POPE, MME, SEED-Bench, LLaVA^{W}, and MM-Vet. The Public model denotes the shared backbone before client-specific expert composition. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the offline MoE model trained with joint access to the same client datasets used in our distributed setting.

Method	POPE				MME		SEED-Bench			LLaVA^{W}				MM-Vet
	Rand. ↑	Pop. ↑	Adv. ↑	Avg. ↑	Perc. ↑	Cog. ↑	All. ↑	Img. ↑	Vid. ↑	All ↑	Comp. ↑	Conv. ↑	Det. ↑	Acc. ↑
Offline reference models (upper bound)														
LLaVA-Phi	–	–	–	85.00	1335.10	–	–	–	–	–	–	–	–	28.90
MoE-LLaVA†	–	–	–	85.70	1423.00	–	–	–	–	94.10	–	–	–	34.30
MoE-LLaVA	87.39	85.83	83.73	85.65	1331.75	274.64	60.84	66.40	39.79	76.20	78.70	74.40	73.70	20.20
Individual experts														
COCO	80.07	76.13	73.29	76.50	1096.75	250.00	56.01	61.24	36.19	65.70	69.50	63.20	60.70	24.40
GQA	78.47	74.47	73.37	75.44	1074.07	251.79	29.61	27.46	37.74	52.50	47.90	55.10	57.60	16.70
VQA^{OCR}	63.95	64.93	64.80	64.56	994.06	262.86	58.34	63.44	39.02	61.30	58.80	67.50	57.80	17.60
VG	66.05	73.07	68.27	69.13	598.64	236.07	49.57	53.72	33.83	49.70	48.50	63.40	32.90	16.80
VQA^{T}	51.75	50.13	50.13	50.67	71.55	31.07	58.37	63.38	39.39	61.00	59.30	71.30	51.90	23.00
Public model	83.54	79.37	78.53	80.48	862.53	281.07	59.36	64.42	40.17	73.30	77.90	72.90	65.30	23.20
Model Soup (avg)	76.26	73.36	72.89	74.17	1054.37	242.38	53.38	58.17	33.50	42.40	43.70	45.20	44.90	15.70
Model Soup (wtd)	76.72	73.54	73.04	74.43	1067.26	250.40	54.05	59.92	34.78	43.20	44.70	46.20	45.10	14.40
BTM	74.81	72.57	71.03	72.80	1106.64	256.64	52.82	60.16	35.81	41.70	44.30	45.40	44.80	16.30
BTX	75.37	73.14	72.35	73.62	1096.48	252.10	51.43	58.36	33.58	44.40	45.10	46.80	46.30	17.00
FlexOlmo-LoRA	83.54	79.43	78.63	80.53	862.53	281.06	59.39	64.44	40.24	72.10	75.40	72.90	65.00	22.30
FlexOlmo	74.91	75.30	74.73	74.98	1193.71	250.00	58.79	63.80	39.79	52.00	52.60	56.00	45.70	18.30
DistMoE-LoRA	**83.95**	79.82	78.76	**80.84**	862.60	281.07	59.38	64.46	40.14	**75.30**	**77.20**	**78.60**	**66.20**	**23.80**
DistMoE	80.86	**80.80**	**79.83**	80.50	**1291.41**	**287.50**	**59.96**	**65.00**	**40.88**	55.90	54.10	69.10	42.10	19.80

achieves strong accuracy and F1-score on Qwen-1.8B and StableLM-1.6B, showing that public-anchored routing helps preserve reliable object grounding while reusing domain-specific experts. However, DistMoE-LoRA is also competitive, especially for Phi-2.7B, indicating that lightweight expert adaptation can still provide effective hallucination mitigation. We also observe that the yes-answer ratio becomes more balanced compared with the public model in several settings, suggesting that DistMoE reduces the tendency to over-answer "yes" and produces responses more consistent with the visual evidence.

Detailed Results. Tabs. 9 and 10 show the detailed Qwen-1.8B results. Across the fine-grained academic-task metrics, DistMoE generally improves over the distributed baselines. On instruction-

Table 13: **Detailed academic-task benchmark results with StableLM-1.6B.** We report fine-grained results on VQA^{v2}, GQA, SQA, and VQA^{T}. The Public model denotes the shared backbone before client-specific expert composition. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the offline MoE model trained with joint access to the same client datasets used in our distributed setting.

Method	VQA^{v2}				GQA				SQA		VQA^{T}
	Yes/No ↑	Num. ↑	Other ↑	Overall ↑	Binary ↑	Open ↑	Acc. ↑	Dist. ↓	Acc. ↑	IMG-Acc. ↑	Acc. ↑
Offline reference models (upper bound)											
MoE-LLaVA†	–	–	–	76.70	–	–	60.30	–	–	62.60	50.10
MoE-LLaVA	89.37	55.63	67.28	76.65	72.30	44.28	57.90	2.07	58.23	57.96	59.15
Individual experts											
COCO	85.06	50.21	60.35	69.26	62.01	36.25	48.08	2.26	50.04	38.28	44.35
GQA	84.94	49.52	58.93	68.57	66.38	41.39	52.28	2.21	48.01	40.31	37.84
VQA^{OCR}	84.04	46.41	53.26	65.16	55.38	25.63	39.28	2.25	50.08	47.94	39.35
VG	60.17	16.60	43.19	47.17	30.95	12.51	20.97	2.91	41.22	37.83	11.47
VQA^{T}	5.14	2.05	40.51	22.48	0.33	0.60	0.48	4.24	49.75	50.28	2.54
Public model	2.38	6.39	47.95	24.46	12.02	20.69	16.71	2.40	50.70	54.73	27.24
Model Soup (avg)	82.58	49.50	58.99	67.63	64.93	38.78	50.79	2.28	51.93	50.96	42.81
Model Soup (wtd)	83.45	50.02	59.61	68.34	65.62	39.19	51.32	2.26	52.48	51.50	43.26
BTM	84.32	50.57	60.23	69.05	66.98	40.01	52.39	2.21	53.14	52.03	43.71
BTX	85.19	51.07	60.86	69.76	66.30	39.60	51.86	2.24	53.57	52.48	44.16
FlexOlmo-LoRA	72.48	43.68	53.40	62.61	57.38	39.58	47.92	1.96	49.06	42.05	37.42
FlexOlmo	84.05	53.35	63.21	70.67	62.45	42.01	51.39	2.13	50.29	42.64	44.50
DISTMOE-LORA	76.21	46.82	56.27	64.93	60.25	39.15	49.17	**1.85**	51.48	50.25	39.40
DISTMOE	**86.93**	**52.11**	**62.10**	**71.19**	**68.35**	**40.82**	**53.46**	2.19	**54.66**	**53.64**	**45.06**

following benchmarks, DISTMOE-LoRA is particularly strong on LLaVA^{W} and MM-Vet, showing that the parameter-efficient variant can provide competitive expert composition with lower adaptation cost. The individual experts are often strong on specific domains but vary substantially across benchmarks, motivating routing-based composition rather than relying on a single client expert.

Tabs. 11 and 12 show the same evaluation for Phi-2.7B. The trends are consistent with Qwen-1.8B: DISTMOE improves the strongest distributed performance on most academic-task metrics, while DISTMOE-LoRA remains competitive on instruction-following benchmarks. These results indicate that the gains are not tied to a single language backbone and that public-anchored calibration transfers across MLLM configurations.

StableLM-1.6B Results. Tabs. 13 and 14 report the complete StableLM-1.6B results. The results further support the main findings: DISTMOE improves over model-merging and FlexOlmo-style baselines on most academic-task metrics, while DISTMOE-LoRA provides strong instruction-following performance, especially on LLaVA^{W} and MM-Vet. This confirms that the proposed public-private calibration remains effective on a third MLLM backbone.

E Additional Analyses of DISTMOE

This section examines *why* DISTMOE works beyond single-run accuracy: whether its gains survive training randomness, how each Stage-II component contributes to composition, what role $\mathcal{L}_{\text{iso}}$ actually plays, and whether the improvements can be attributed to additional capacity.

Residual diagnostics. All statistics are computed on $\Delta_m(x)$ from the same 19k public inputs for every client and layer, then averaged over layers. RMS-CV is the coefficient of variation (std/mean) of the residual RMS across the M clients, and Max/min is the ratio between the largest and smallest client RMS. Effective rank is the entropy-based rank [53] of the residual covariance, normalized by d. SWD is the mean pairwise sliced Wasserstein distance [54] between the M clients' residual distributions.

E.1 Robustness across random seeds

Tab. 15 repeats the comparison with FlexOlmo over three seeds and shows that the advantage of DISTMOE is systematic rather than the product of a favorable run. More revealing than the means

Table 14: **Detailed instruction-following benchmark results with StableLM-1.6B.** We report fine-grained results on POPE, MME, SEED-Bench, LLaVA^{W}, and MM-Vet. The Public model denotes the shared backbone before client-specific expert composition. MoE-LLaVA† denotes the original results reported in [1], while MoE-LLaVA denotes the offline MoE model trained with joint access to the same client datasets used in our distributed setting.

Method	POPE				MME		SEED-Bench			LLaVA^{W}				MM-Vet
	Rand. ↑	Pop. ↑	Adv. ↑	Avg. ↑	Perc. ↑	Cog. ↑	All. ↑	Img. ↑	Vid. ↑	All ↑	Comp. ↑	Conv. ↑	Det. ↑	Acc. ↑
Offline reference models (upper bound)														
MoE-LLaVA†	–	–	–	85.70	1318.20	–	–	–	–	86.80	–	–	–	26.90
MoE-LLaVA	85.36	85.61	83.59	84.85	1304.56	233.04	55.42	61.56	36.91	54.20	53.70	68.60	53.50	18.40
Individual experts														
COCO	83.74	83.56	78.28	81.86	1230.27	209.48	48.47	52.31	31.08	69.50	71.00	73.00	60.30	20.60
GQA	83.82	84.01	79.32	82.38	1243.68	211.70	36.69	36.82	29.24	53.00	50.30	66.40	39.80	17.70
VQA^{OCR}	74.36	74.17	72.90	73.81	1013.82	201.32	47.92	51.51	30.28	61.10	61.80	57.70	61.10	15.50
VG	84.78	77.50	74.20	78.83	649.23	125.71	4.56	5.77	0.05	51.70	51.60	60.30	41.40	20.60
VQA^{T}	55.70	53.47	52.57	53.91	462.42	131.43	46.80	50.31	33.48	61.40	58.60	69.30	56.40	21.10
Public model	58.76	53.80	53.37	55.31	518.52	243.57	48.10	52.02	33.27	68.00	71.30	71.30	57.10	20.00
Model Soup (avg)	79.96	79.29	77.41	78.89	1126.25	205.08	50.14	54.96	31.98	48.80	44.70	59.50	43.00	12.70
Model Soup (wtd)	80.73	80.05	78.15	79.64	1134.57	203.11	50.63	55.49	32.29	49.30	45.10	60.10	43.40	12.90
BTM	81.48	80.80	79.61	80.63	1179.35	208.97	51.11	56.02	32.60	49.80	45.60	60.70	43.80	13.00
BTX	82.24	81.55	78.88	80.89	1190.47	210.92	51.60	56.55	32.91	50.20	46.00	61.20	44.20	13.10
FlexOlmo-LoRA	75.62	77.45	76.04	76.37	956.41	200.57	51.49	55.29	31.48	70.40	72.60	73.50	60.20	21.70
FlexOlmo	77.84	78.07	76.87	77.59	1171.74	221.79	52.18	57.01	33.91	50.40	44.50	60.00	49.50	17.70
DistMoE-LoRA	84.06	83.92	81.45	83.14	1023.57	203.42	52.41	57.05	33.78	**73.60**	**74.00**	**75.10**	**63.70**	**24.50**
DistMoE	**84.98**	**84.27**	**82.27**	**83.84**	**1274.28**	**216.43**	**53.94**	**59.10**	**34.39**	52.80	48.30	64.20	46.50	14.80

Table 15: Three-seed comparison (mean $\pm$ std) of DistMoE and FlexOlmo. Δ: difference of means. Wins: matched seed-benchmark comparisons won by DistMoE.

LLM	Method	VQA$^{\text{v2}}$	GQA	SQA^{I}	VQA^{T}	POPE	MME	SEED	LLaVA^{W}	MM-Vet	Wins
Qwen-1.8B	FlexOlmo	69.95 $\pm$0.24	54.25 $\pm$0.07	60.92 $\pm$0.07	41.69 $\pm$0.07	85.72 $\pm$0.36	1173.80 $\pm$3.08	54.93 $\pm$0.30	52.23 $\pm$0.45	14.93 $\pm$0.12	–
	DistMoE	**70.25** $\pm$0.16	**55.38** $\pm$0.11	**61.28** $\pm$0.22	**42.70** $\pm$0.05	**86.41** $\pm$0.13	**1188.87** $\pm$1.32	**55.36** $\pm$0.25	**59.57** $\pm$0.31	**15.23** $\pm$0.15	27/27
	Δ	+0.30	+1.13	+0.36	+1.01	+0.70	+15.06	+0.43	+7.33	+0.30	
Phi-2.7B	FlexOlmo	67.57 $\pm$2.13	49.41 $\pm$4.75	63.12 $\pm$1.64	33.63 $\pm$6.93	81.18 $\pm$5.41	**1113.72** $\pm$69.30	59.00 $\pm$0.18	56.57 $\pm$4.82	16.60 $\pm$1.75	–
	DistMoE	**69.95** $\pm$1.22	**50.18** $\pm$5.74	**66.34** $\pm$0.25	**35.42** $\pm$5.68	**82.21** $\pm$2.43	1053.75 $\pm$210.19	**59.06** $\pm$0.79	**59.30** $\pm$3.14	**19.40** $\pm$0.53	21/27
	Δ	+2.38	+0.77	+3.22	+1.79	+1.03	-59.97	+0.06	+2.73	+2.80	

is the variance. Composing independently trained experts is intrinsically fragile: on Phi-2.7B, both methods fluctuate considerably across seeds, indicating that small differences in how each expert drifts during Stage-I propagate into the composed model. Stage-II dampens this effect, and DistMoE is generally more stable than FlexOlmo, most visibly on 12/18 benchmarks. This suggests that recalibrating every expert against the same public anchor does not only shift performance upward, but also makes the outcome of composition less dependent on the idiosyncrasies of each client's training run. Consistently, the advantage tends to be larger on the less stable backbone, i.e., calibration matters most precisely where uncalibrated composition is least reliable; we do not extrapolate this trend beyond the evaluated model range. MME on Phi-2.7B is the exception, where a single unstable seed dominates the mean, and we therefore do not claim a uniform improvement on this backbone.

E.2 Component attribution on Phi-2.7B

Tab. 16 extends Tab. 2 to Phi-2.7B and is best read through the two factors of Proposition 1: the adapter $T(\cdot)$ and $\mathcal{L}_{\text{iso}}$ shape the residual $\mathbf{\Delta}_m$, i.e., *what* an expert contributes, whereas the bias b_m sets the routing prior α_m, i.e., *how much* it contributes. Three lessons emerge. *First, extra flexibility without a constraint is harmful:* an adapter trained only with next-token prediction tends to fall below Stage-I, as it is free to distort expert outputs in client-specific ways that make them even less comparable. *Second, the constraint is what turns the adapter into a calibration tool:* pairing it with $\mathcal{L}_{\text{iso}}$ ties every client to the same reference geometry, and the adapter starts to help consistently. *Third, both factors are needed:* recalibrating the routing prior on top of the adapter yields the largest jump on perception-heavy benchmarks such as MME, yet the prior alone cannot compensate for residuals of mismatched geometry, and the full model is only reached when both what and how much are calibrated. Once the routing prior is in place, the marginal gain of $\mathcal{L}_{\text{iso}}$ becomes smaller, consistent with the two factors partially compensating each other.

Table 16: Effect of Stage-II components of DISTMOE with Phi-2.7B (single run, averaged router composition). Stage-I corresponds to FlexOlmo.

Stage	$T(\cdot)$	$\mathcal{L}_{\text{iso}}$	b_m	VQA$^{\text{v2}}$	GQA	SQA^{I}	VQA^{T}	POPE	MME	SEED	LLaVA^{W}	MM-Vet
I	–	–	–	70.03	54.84	62.77	41.63	74.98	1193.71	58.79	52.00	18.30
II	✓	✗	✗	68.72	54.04	63.05	41.04	74.83	1120.46	59.25	52.20	17.10
II	✓	✓	✗	69.96	55.75	65.28	41.70	78.46	1133.58	59.74	52.80	17.60
II	✗	✗	✓	69.89	53.24	62.92	41.53	75.68	1144.60	59.89	53.20	17.30
II	✓	✗	✓	71.27	54.92	64.75	41.49	78.65	**1294.44**	59.88	54.60	19.60
II	✓	✓	✓	**71.31**	**56.53**	**66.34**	**41.87**	**80.50**	1291.41	**59.96**	**55.90**	**19.80**

Table 17: Three-seed comparison of Stage-II with Phi-2.7B, with and without $\mathcal{L}_{\text{iso}}$. Both variants use T_m+b_m and the same seeds. Paired Δ: with minus without, mean $\pm$ std over seeds.

Benchmark	w/o $\mathcal{L}_{\text{iso}}$	w/ $\mathcal{L}_{\text{iso}}$	Paired Δ
VQA$^{\text{v2}}$	69.94 ± 1.47	69.95 ± 1.22	$+0.01 \pm 0.59$
GQA	50.62 ± 5.26	50.18 ± 5.74	-0.45 ± 0.80
SQA^{I}	66.56 ± 0.92	66.34 ± 0.25	-0.23 ± 0.68
VQA^{T}	36.21 ± 4.80	35.42 ± 5.68	-0.80 ± 1.26
POPE	82.68 ± 2.47	82.21 ± 2.43	-0.47 ± 1.18
MME	1084.13 ± 177.99	1053.75 ± 210.19	-30.38 ± 44.46
SEED	59.00 ± 0.77	59.06 ± 0.79	$+0.06 \pm 0.16$
LLaVA^{W}	58.83 ± 3.72	59.30 ± 3.14	$+0.47 \pm 0.85$
MM-Vet	16.97 ± 2.35	19.40 ± 0.53	$+2.43 \pm 1.94$

E.3 Controlled multi-seed effect of the isotropic loss

Tab. 17 isolates $\mathcal{L}_{\text{iso}}$ by comparing Stage-II with and without it across seeds, all else being equal. On short-answer QA benchmarks, many of which coincide with client domains, the effect is within seed noise. On open-ended generation benchmarks, in contrast, $\mathcal{L}_{\text{iso}}$ helps consistently, and on MM-Vet it both improves every run and makes the outcome markedly more stable. We attribute this asymmetry to how answers are composed. A short answer is typically decided by a few tokens routed to a single in-domain expert, so the relative geometry of the other experts matters little. A long-form answer, instead, mixes many tokens routed to different experts, and comparable residual geometry becomes essential for these contributions to combine coherently. Together with Tab. 4, where $\mathcal{L}_{\text{iso}}$ consistently improves residual geometry, this positions $\mathcal{L}_{\text{iso}}$ as a complementary regularizer whose benefit is task-dependent, rather than the main source of the gains of DISTMOE. With only three paired runs, we report these trends as consistent cross-seed evidence and do not claim statistical significance.

E.4 Are the gains explained by additional parameters?

DISTMOE adds trainable parameters per client relative to FlexOlmo (Tab. 6), entirely due to the Stage-II adapters, which raises the question of whether the gains simply reflect more capacity. Two controlled comparisons in Tab. 16 suggest otherwise. Adding the adapter alone, which accounts for all the extra parameters, does not improve over Stage-I, whereas $\mathcal{L}_{\text{iso}}$, which adds no parameters at all, is what makes the adapter effective. The improvements therefore stem from *how* the added parameters are constrained, i.e., anchored to the public expert, regularized toward a common residual geometry, and paired with a recalibrated routing prior, rather than from capacity itself. This also implies that the adapter could be made considerably lighter, e.g., low-rank, without changing the underlying mechanism.

F Number of LoRA experts in DISTMOE LoRA

We analyze the effect of using multiple LoRA experts per client in Fig. 6. Increasing the number of selected LoRA experts improves performance at first, showing that multiple lightweight experts provide complementary specialization paths and enable richer expert composition. The best performance is obtained with a moderate number of LoRA experts, while further increasing the number of experts slightly degrades performance. This suggests that too many selected LoRA experts may

introduce redundant or noisy expert contributions, weakening sparse specialization. Therefore, we use $R = 16$ as the default setting, which provides the best trade-off between adaptation capacity and routing stability.

G Routing Distributions

We present the routing distributions of DISTMOE in Figs. 7, 8, and 9, corresponding to Qwen-1.8B, Phi-2.7B, and StableLM-1.6B. For each backbone, we compute the top-4 routing mass on representative image-QA and instruction-following benchmarks, where routing mass is the aggregate normalized router weight assigned to each expert across tokens at each MoE layer. The plots show how the router distributes computation between the shared public expert and the client-specific private experts trained on the datasets described in Section C.1.

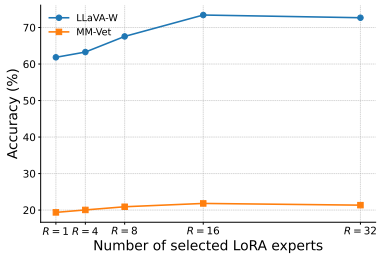
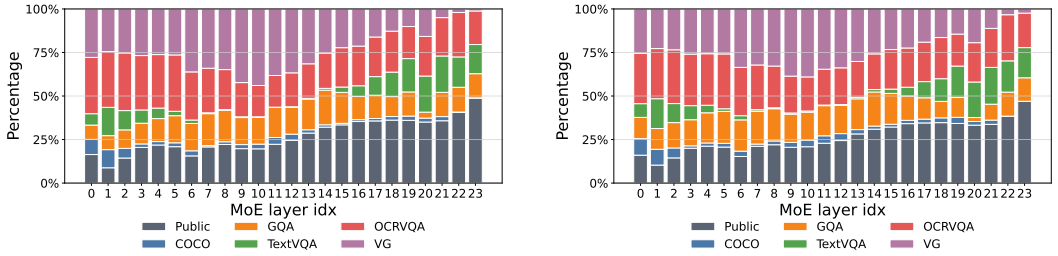


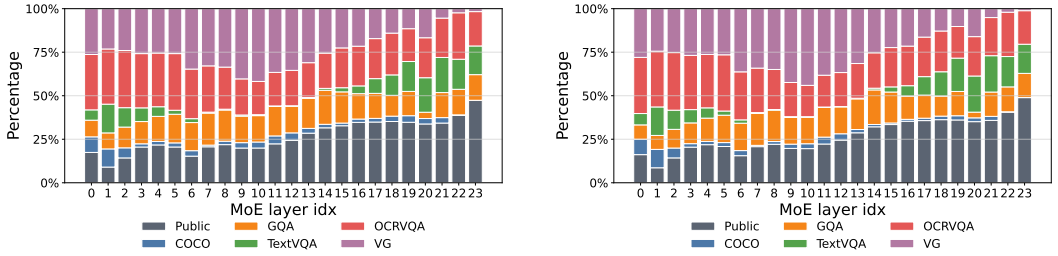
Figure 6: Number of LoRA experts.

Across all three backbones, DISTMOE assigns non-trivial routing mass to the public expert while still activating private experts in a benchmark-dependent manner. This indicates that the router does not simply replace the public expert with domain experts; instead, private experts act as calibrated residual contributors around the shared public expert. In most of the datasets, the public route becomes more dominant in later layers, suggesting that later layers rely more on general visual-language and instruction-following knowledge. Conversely, middle layers often show stronger private-expert activity, reflecting the use of domain-specific visual grounding and reasoning features. The comparison among Figs. 7, 8, and 9 further shows that routing behavior is backbone-dependent. Nevertheless, the same qualitative trend holds: DISTMOE produces sparse but non-exclusive expert usage, preserves a stable public pathway, and composes multiple private experts without requiring explicit domain labels.



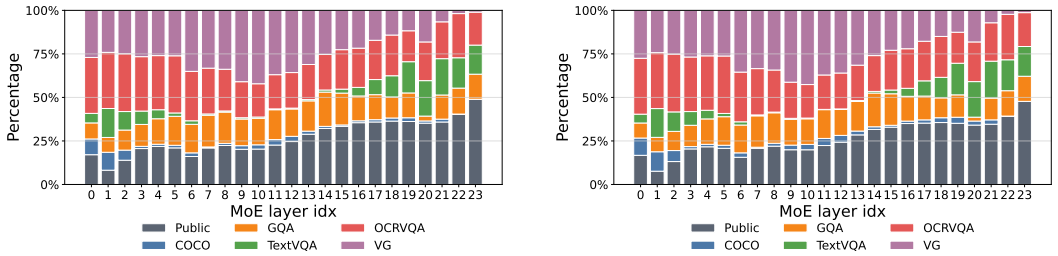
(a) GQA

(b) ScienceQA



(c) TextVQA

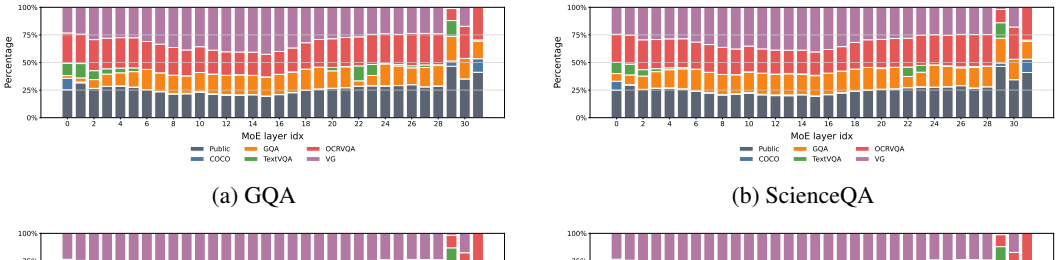
(d) POPE



(e) LLaVA-Bench

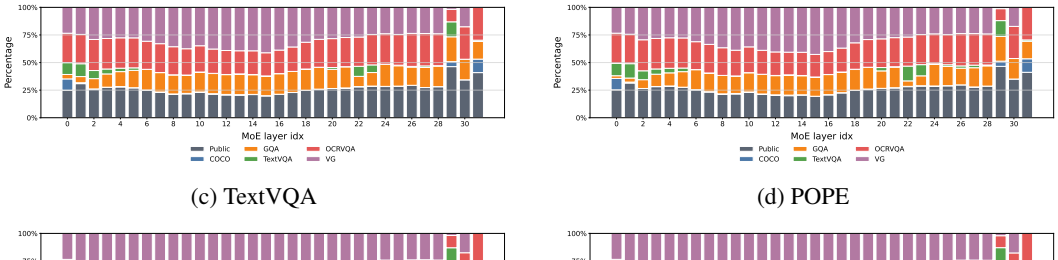
(f) MM-Vet

Figure 7: Top-4 routing mass of DISTMOE with Qwen-1.8B.



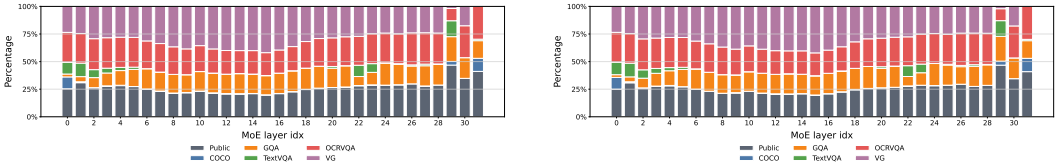
(a) GQA

(b) ScienceQA



(c) TextVQA

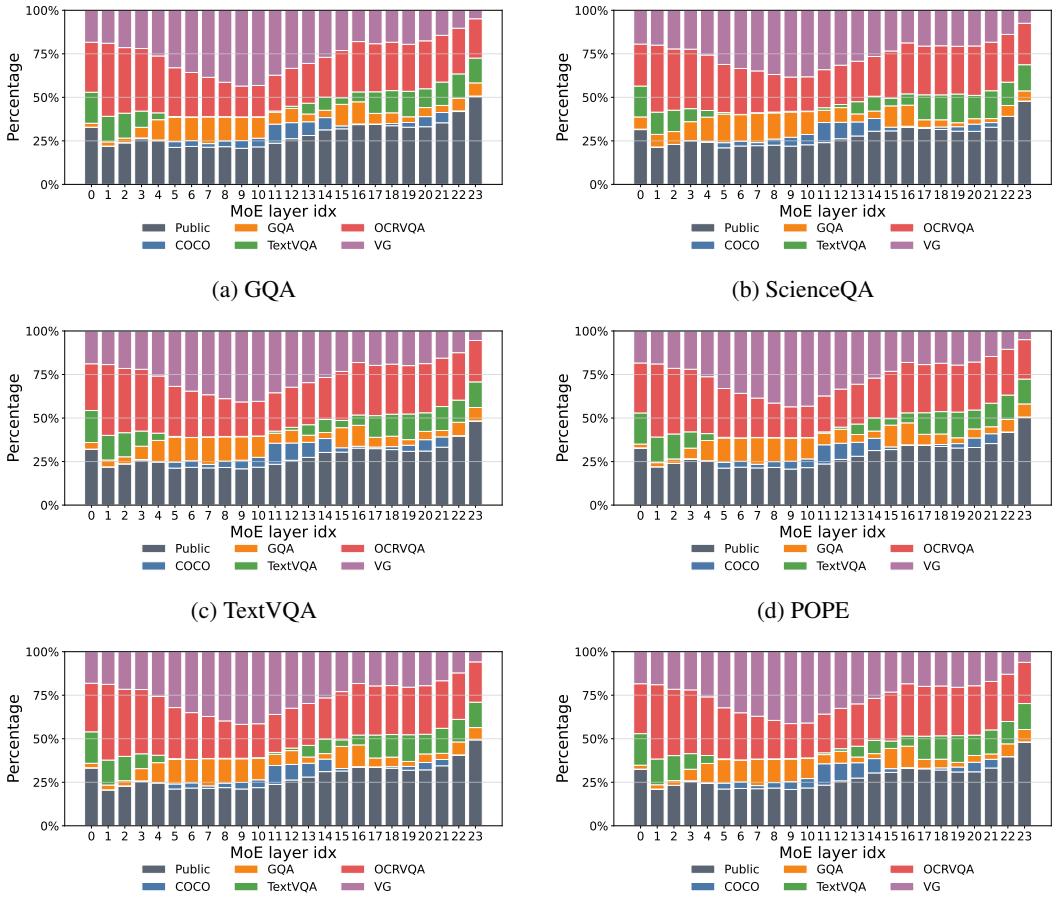
(d) POPE



(e) LLaVA-Bench

(f) MM-Vet

Figure 8: Top-4 routing mass of DISTMOE with Phi-2.7B.



(a) GQA (b) ScienceQA

(c) TextVQA (d) POPE

(e) LLaVA-Bench (f) MM-Vet

Figure 9: Top-4 routing mass of DistMoE with StableLM-1.6B.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We state the claims and contributions in the abstract and introduction sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the Limitations in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: We provide the theoretical justifications in Proposition 1 and Appendix B.2.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: For the details to reproduce the experimental results, please refer to Section 5 and for further details refer Appendix C, and D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

Answer: [Yes]

Justification: Please refer to Section 5 and Appendix C for the datasets used , all are publicly available. Further code will be available on the project website.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Please refer to the Section 5 and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Tabs. 15 and 17 report mean $\pm$ std over three training seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We use 4 NVIDIA A100 GPUs for our experiments and discussed the details in Appendix C.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We confirm the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss the impacts of the work in Section 6.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This work poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cite all datasets and models utilized in our experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: This work does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: LLMs are used as the language backbones of the evaluated MLLMs, including Qwen-1.8B, Phi-2.7B, and StableLM-1.6B. We also use GPT-4 as an LLM judge, requiring an OpenAI API key, for two datasets i.e. LLaVA^{W} and MM-Vet following the standard benchmark protocols. Please note that GPT-4 is used only for evaluation and not for training, data generation, or method development.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.