DaMo: Data Mixing Optimizer in Fine-tuning Multimodal LLMs for Mobile Phone Agents

Kai Shi^{+} and **Jun Yang**$^{+}$ and **Ni Yang**$^{+}$ and **Binqiang Pan** and **Qingsong Xie***
and **Chao Zhang** and **Zhenyu Yang** and **Tianhuang Su** and **Haonan Lu**
OPPO AI Center
`{shikai,yangjun2,yangni,xieqingsong1}@oppo.com`

Abstract

Mobile Phone Agents (MPAs) have emerged as a promising research direction due to their broad applicability across diverse scenarios. While Multimodal Large Language Models (MLLMs) serve as the foundation for MPAs, their effectiveness in handling multiple mobile phone tasks simultaneously remains limited. Although multitask supervised fine-tuning (SFT) is widely adopted for multitask learning, existing approaches struggle to determine optimal training data compositions for peak performance. To address this challenge, we propose DaMo (Data Mixture Optimizer) – a novel solution employing a trainable network that predicts optimal data mixtures by forecasting downstream task performance for any given dataset ratio. To support comprehensive evaluation, we introduce PhoneAgentBench, the first specialized benchmark to evaluate MLLMs on multimodal mobile phone tasks, comprising 1,235 QA pairs spanning diverse real-world industrial mobile application scenarios. Demonstrating strong predictive capability (R^2=0.81) in small-scale pilot experiments, DaMo efficiently extrapolates optimal data mixing configurations. Our results show DaMo achieves 3.06% average score improvement on PhoneAgentBench and open-source benchmarks, including BFCL-v3, MME-Reasoning, MME-Perception, and OCRBench, compared to alternative methods. Through predicting optimal data mixture only on open-source benchmarks, DaMo outperforms other approaches by 6.70% in terms of average score. Moreover, DaMo improves the metrics by 12.74% than other methods when used solely for MLLM optimization on the BFCL-v3 task. Notably, DaMo maintains robust scalability, preserving its effectiveness when applied to other model architectures.

1 Introduction

Mobile phone agents (MPAs) have attracted huge attention due to their practicability in a multitude of scenarios. An ideal MPA has to master multiple capabilities, such as environment perception (Zhang et al., 2024; Ingold, 2021), task planning (Song et al., 2023; Liu et al., 2024c), multimodal reasoning (Lu et al., 2022; Wang et al., 2024), function call (Chen et al., 2024a; Basu, 2024), and personalized memory (Li et al., 2024a; Yuan et al., 2023).

The advent of multimodal large language models (MLLMs) provides a promising solution for the ideal agent. However, existing MLLMs encounter significant challenges in effectively integrating these diverse capabilities. Consequently, developing a versatile model capable of handling multiple tasks is critical for creating a advanced phone agent.

Multitask supervised fine-tuning (SFT) is the predominant approach utilized to empower MLLMs in addressing multiple tasks. Nevertheless, in light of numerous training datasets and downstream tasks, identifying optimal data blending strategies to maximize model performance remains a significant research challenge. The existing works on data mixture optimization (Xie et al., 2023a; Ge et al., 2024; Albalak et al., 2023; Liu et al., 2024a) focus on the pretraining phase by predicting validation loss of LLM. However, these methods are inadequate to determine the optimal data mixture for fine-tuning MLLMs, as they fail to directly correlate with model performance on downstream tasks. The sampling strategies commonly used in industrial scenarios, e.g., uniform sampling, natural sampling, and random sampling, can not yield optimal mixture duo to fixed or random data mixing ratios. Grid search (Liashchynskyi, 2019) needs huge cost to find optimal mixture when the number of the training datasets becomes large.

We investigate whether downstream task perfor-

---

*Corresponding Author. $^{+}$Equal Contribution

Findings of the Association for Computational Linguistics: ACL 2026, pages 28813–28830
July 2-7, 2026 ©2026 Association for Computational Linguistics

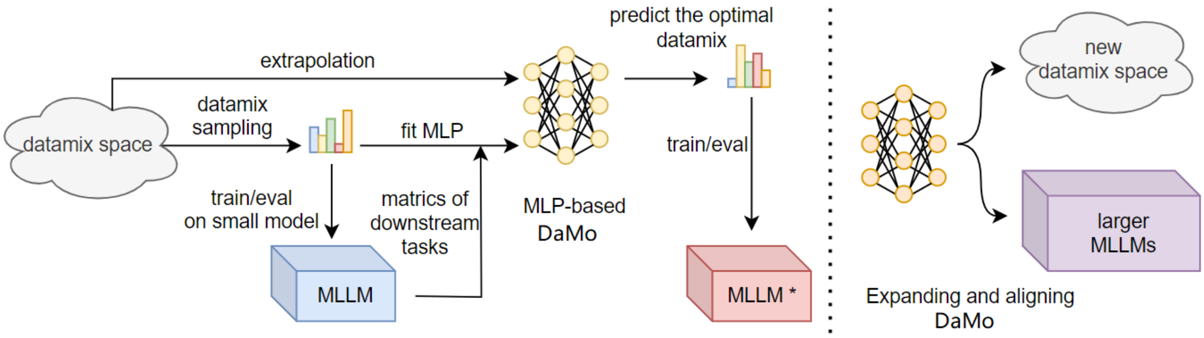


Figure 1: Illustration of our pipeline for obtaining the optimal data mixture. Left: Given m training sets with a batch size of b, all possible mixture combinations constitute the data mixing space. We sample a small number of data mixture from this space, train them on a small MLLM, and then evaluate downstream task performance. Using the data mixture as inputs and the metrics as outputs, we fit a MLP to establish the DaMo. By extrapolating from the data mixing space, we predict the optimal data mixture to train the MLLM. Right: Demonstrates the extension and alignment of DaMo to other MLLMs and new data mixing spaces.

mance can be reliably predicted for any given data mixture prior to actual model training, including identifying the optimal mixture that would yield optimal performance. To this end, we propose the **d**ownstream task **p**erformance **p**rediction (DaPP) method to build **Da**ta **M**ixing **O**ptimizer (DaMo). DaPP leverages a function to straightly predict model performance at downstream tasks. Considering that exponential functions used in (Xie et al., 2023a; Ge et al., 2024; Albalak et al., 2023) are not well-aligned with SFT performance trajectories for specific downstream applications (Huang et al., 2019; Xie et al., 2024; Isik et al., 2024), we propose to utilize a trainable neural network for target fitting. The optimal data mixture is obtained through extrapolation via DaMo.

Another obstacle in developing an ideal mobile phone agent is the absence of comprehensive real-world industrial benchmarks for evaluating MPA performance. Current benchmarks (Gao et al., 2024; Cheng et al., 2024; Li et al., 2025a; Wang et al., 2025) in this domain predominantly focus on Graphical User Interface (GUI) tasks, which fail to capture the full spectrum of practical application scenarios. To address this critical gap, we introduce PhoneAgentBench - a thorough benchmark encompassing four fundamental capabilities: 1) complex task planning, 2) device-native tool usage, 3) multimodal memory, and 4) screen context understanding. Our benchmark comprises 1235 meticulously validated test cases that simulate real-world phone interactions.

Our proposed DaMo demonstrates three key advantages. First, it achieves 3.06% average score gain on PhoneAgentBench and general benchmarks, including BFCL-V3 (Patil et al., 2025a), MME-perception (Fu et al., 2023), MME-reasoning (Yuan et al., 2025), and OCRBench (Liu et al., 2024d), compared to state-of-the-art method, DML (Ye et al., 2024). Second, when only optimized on general benchmarks, DaMo surpasses state-of-the-art method by 6.70% in terms of average score on open-source benchmarks. Third, DaMo exhibits robust scalability across other models, while introducing significant gains on downstream tasks over other methods.

Our core contributions are as follows.

- We propose Downstream Task Performance Prediction method to establish a Data Mixing Optimizer, which directly estimates model performance on downstream tasks for optimal data mixing.

- We construct PhoneAgentBench, a benchmark spanning four critical dimensions: complex task planning, device-native tool usage, multimodal memory, and screen context understanding, mirroring real-world mobile interaction scenarios.

- Through systematic experiments, our method demonstrates exceptional generalization and scalability, outperforming other methods on PhoneAgentBench, and achieving state-of-the-art performance on the BFCL-V3 leaderboard among 4B-scale models, while also maintaining stable prediction accuracy with efficient adaptation to other models.

2 Related work

Data Mixing Early heuristic approaches like uniform sampling (Michel et al., 2021) gave way to learnable solutions; DoReMi (Xie et al., 2023b) uses Group DRO (Sagawa et al., 2020) for domain weights; ODM (Albalak et al., 2023) frames selection as a bandit problem; BiMix (Ge et al., 2024) jointly optimizes domain proportions and data scaling. These approaches are designed for pre-training stage of LLM, which can not be directly applicable to SFT of MLLM since the intricate interactions among downstream tasks. SFTMix (Xiao et al., 2024) optimizes intra-dataset ratios but cannot handle multi-source data. Data from transfer is leveraged to estimate validation loss for LLM (Li et al., 2025b). A dynamic sampling strategy (Zhu et al., 2024) is proposed to fine-tune MOE LLM via recording the routing tokens and calculating the corresponding gate load. Key gaps remain in developing general multi-source mixing schemes for fine-tuning MLLMs.

Agent Benchmark PlanBench (Valmeekam et al., 2023) and REALM-Bench (Geng and Chang, 2025) assess planning capabilities. ToolBench (Qin et al., 2023), BFCL (Patil et al., 2025a), and API-Bank (Li et al., 2023) evaluate tool invocation and ReflectionBench (Li et al., 2024b) measures self-reflection. LTM Benchmark (Castillo-Bolado et al., 2024) tests memory retention. These benchmarks are limited to single-dimensional evaluations, lacking holistic assessment. GAIA (Mialon et al., 2023) uses end-to-end evaluation to assess general agents, but lacks granularity. AgentBench (Liu et al., 2023) and KAgentBench (Pan et al., 2023) are unimodal, ignoring multimodal interaction. ScreenSpot-Pro (Cheng et al., 2024), MobileViews (Gao et al., 2024), VisualAgentBench (Liu et al., 2024b), ScreenSpot-Pro (Li et al., 2025a), and MMBench-GUI L2 (Wang et al., 2025) can evaluate phone agents, but they are designed mainly for GUI tasks. A critical gap remains: the absence of a comprehensive benchmark supporting multimodal interaction while systematically evaluating mobile phone agents across planning, tool usage, memory, and other dimensions.

3 PhoneAgentBench

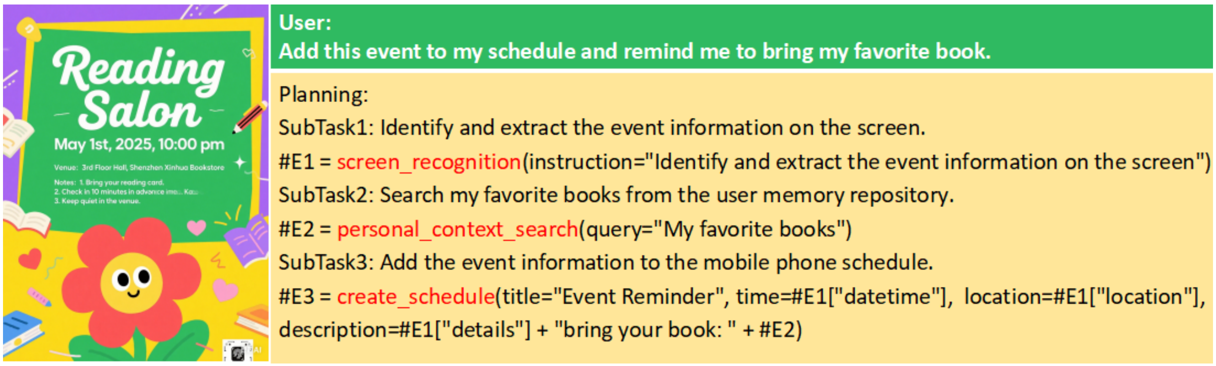
To develop a mobile phone agent benchmark tailored to real-world industrial application scenarios, we design a novel benchmark supporting systematic evaluation across key dimension such as multimodal interaction, planning, tool use, and memory. This benchmark encompasses six carefully curated datasets focusing on key mobile phone application tasks. We use Multimodal Task Planning task (MT-Plan) as a case to describe data construction.

MT-Plan MT-Plan is designed to evaluate multimodal task planning capabilities. Unlike T-Eval planning (Chen et al., 2023), it focuses on multimodal complex task interactions in phone agent scenarios. As shown in Figure 2, MT-Plan takes <image + query> as input and outputs a planning structured as a directed acyclic graph (DAG). Images are sourced from real photos or mobile screenshots, while tools are derived from APIs provided by operating systems or app ecosystems. Queries and plannings are carefully constructed by annotators based on the images. Queries are required to be concise, colloquial, and aligned with real users' daily needs. Meanwhile, tasks must be sufficiently complex to require plannings to invoke at least 2 tools. To ensure data accuracy, three annotators were invited to conduct cross-validation, and the data with inconsistent annotations was removed. Additionally, to evaluate the dataset's complexity and diversity, we compared the metrics of MT-Plan and T-Eval planning, as presented in Table 7.

The evaluator adopts the T-Eval planning evaluator (Chen et al., 2023): it compares the predicted plan with the golden plan, and calculates the score based on the length of the longest ordered action sequence derived from similarity-matched pairs.

The construction methods of the remaining five datasets and the statistics of all the tasks are presented in AppendixA.

Benchmark Scale and Quality Rationale. To ensure data quality and coverage despite the manageable scale, we design PhoneAgentBench with deliberate scalability and representativeness. First, the 1235 samples cover diverse multimodal interactions, systematically evaluating core capabilities such as *tool usage*, *multi-turn reasoning*, and *agentic planning*. The scale of our function set, with 50 core mobile interfaces, aligns with leading benchmarks like BFCL-v4 (Patil et al., 2025c), which focuses on 29 core APIs for agent evaluation, demonstrating that our coverage of functional scenarios is both comprehensive and practically aligned with real-world application ecosystems. Furthermore, as validated in Table 7, the benchmark's overall complexity—measured by the MT-Plan task complex-



User:
Add this event to my schedule and remind me to bring my favorite book.

Planning:
SubTask1: Identify and extract the event information on the screen.
#E1 = screen_recognition(instruction="Identify and extract the event information on the screen")
SubTask2: Search my favorite books from the user memory repository.
#E2 = personal_context_search(query="My favorite books")
SubTask3: Add the event information to the mobile phone schedule.
#E3 = create_schedule(title="Event Reminder", time=#E1["datetime"], location=#E1["location"], description=#E1["details"] + "bring your book: " + #E2)

Figure 2: MT-Plan example.

ity score—is 5.4 times higher than T-Eval (Chen et al., 2023), with 429 dedicated test samples for the Mobile-FC task alone. This high complexity ensures robust evaluation against overfitting, even within a limited sample size.

4 Methodology

This section formalizes multitask fine-tuning optimization as identifying the optimal data mixture to maximize downstream task metrics. We propose predicting unseen mixture performance by fitting the performance of downstream tasks with limited training configurations. The process is shown in Figure 1.

4.1 Problem Formulation

Consider fine-tuning a MLLM using a mixture of m heterogeneous training datasets, denoted as $\mathcal{D} = \cup_{i=1}^{m} \mathcal{D}_i$. Each $\mathcal{D}_i$ contains n_i labeled samples with the total number of samples being $N = \sum_{i=1}^{m} n_i$. We fine-tune the MLLM starting from initial parameters θ_0, using a batch size b, for a maximum of $T = \lceil N/b \rceil$ training steps.

We define the **data mixture proportion** as $\mathbf{p} = [p_1, p_2, ..., p_m]$, where p_i represents the proportion of samples drawn from dataset $\mathcal{D}_i$. The data mixture proportion $\mathbf{p}$ satisfies $\sum_{i=1}^{m} p_i = 1$.

Similarly, we consider k downstream test datasets, denoted as $\mathcal{D}^{test} = \cup_{j=1}^{k} \mathcal{D}_j^{test}$. Let $\mathbf{s} = [s_1, ..., s_k] \in [0, 1]^k$ represent the score of each test dataset. The overall average score of the MLLM with parameters θ is given by $S_\theta = \frac{1}{k} \sum_{j=1}^{k} s_j$.

We aim to find the optimal data mixture proportion $\mathbf{p}^* \in \mathcal{P}$ (where $\mathcal{P}$ denotes the complete data mixing space, $\mathbf{p} \in \mathbb{R}^m$) that maximizes the overall average score of downstream tasks:

$$\mathbf{p}^* = \mathop{\arg\max}_{\mathbf{p} \in \mathcal{P}, t \leq T} \mathbb{E}_{\theta \sim \mathcal{A}(\mathbf{p}, t, \theta_0)} S_\theta, \tag{1}$$

where $\mathcal{A}$ denotes the fine-tuning process that produces the MLLM's parameters θ based on the initial parameters θ_0 for t steps using the data mixture strategy $\mathbf{p}$.

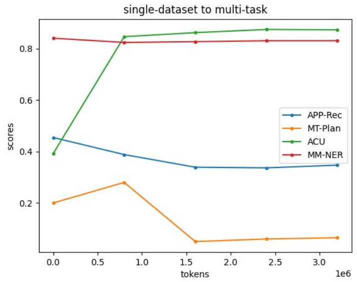
Without any constraints, the size of the set $\mathcal{P}$ that represents batch-wise permutations is given by $|\mathcal{P}| = \frac{N!}{(b!)^T}$, which is computationally intractable. Therefore, we introduce some necessary assumptions to prune the space $\mathcal{P}$. By disregarding the order of samples within the same dataset and keeping the data mixture fixed throughout the entire number of training steps T, we obtain a smaller data mixing space $\mathcal{P}_{fix}$. According to the principle of combination with repetition, the size of this fixed data mixing space $\mathcal{P}_{fix}$ is given by $|\mathcal{P}_{fix}| = C_{m+b-1}^{m-1}$.

4.2 Performance Prediction of Tasks

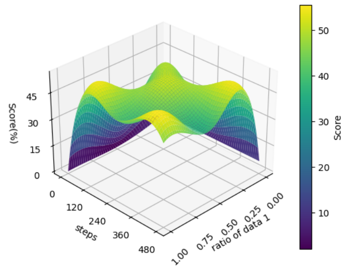
We aim to find the optimal mixture $\mathbf{p}^* \in \mathcal{P}$. Given the high training cost of MLLMs, an exhaustive brute-force search is clearly impractical. To address this problem, we propose DaMo which is able to estimate model performance at downstream tasks without training, given any mixture proportions of training data. Towards this target, we fit a function f to predict performance based on data mixtures. To obtain accurate f, a efficient sampling approach is proposed to generate training samples. The sampling process is detailed as: 1) Randomly select a small set of m-dimensional mixing ratios from $\mathcal{P}_{fix}$. 2) Train MLLM while saving checkpoints at every τ steps. 3) Evaluate each checkpoint to obtain the performance of downstream tasks. This process yields the mapping: (data mixture, training steps) $\rightarrow$ performance of downstream tasks. Based on these samples, we fit f to predict the performance trajectory of unseen mixture:

$$\hat{\mathbf{s}} = f(\mathbf{p}, t; \theta_0), \tag{2}$$

where θ_0 is initial model state and $t = \tau * i$ is train steps of the i-th checkpoint. With an accurate fitting of f, we can extrapolate performance estimates across the entire $\mathcal{P}_{fix}$ space, dramatically reducing the model training costs required to identify the optimal data mixture.



(a) Performance on single-dataset training.



(b) Performance under dual-dataset mixtures

Figure 3: Training dynamics on downstream tasks.

The critical challenge lies in selecting an appropriate function f. While conventional exponential or power-law functions (Achiam et al., 2023; Grattafiori et al., 2024) are widely adopted for pretraining loss convergence, we hypothesize their inadequacy in multi-task fine-tuning scenarios involving interacting datasets. To validate this, we systematically analyze training dynamics under two configurations: (1) single-dataset training (MultiModal-Understanding, MMU) and (2) dual-dataset mixtures (APP Recognition (APP-Rec) + MMU, see Section 5.1).

We trained a MLLM on the MMU dataset and evaluated its performance on PhoneAgentBench. As shown in Figure 3(a), the results reveal distinct task-specific patterns: (1) **Enhancement**: MMU significantly improves ACU performance. (2) **Conflict**: APP-Rec performance degrades with MMU training steps. (3) **Neutrality**: MM-NER shows no correlation with MMU training. (4) **Overfitting**: MT-Plan exhibits initial gains followed by sharp declines, indicating harmful overfitting beyond optimal data volume.

Figure 3(b) demonstrates the complex interaction when training on the mixed dataset of APP-Rec and MMU for the APP-Rec task. The 3D performance surface (X: training steps, Y: APP-Rec training dataset ratio, Z: APP-Rec bench score) exhibits **non-convex topology with non-monotonic fluctuations** along both axes. This nonlinearity fundamentally prevents analytical solutions for Eq. 1 and invalidates conventional exponential and power functions.

Motivated by neural networks' capacity to model high-dimensional nonlinearities, we pioneer their application to DaMo. Our framework implements f as a multi-layer perceptron (MLP) that directly maps data mixture and training step to task performance:

$$\hat{\mathbf{s}} = f_{MLP}(\mathbf{p}, t; \theta_0). \tag{3}$$

4.3 Optimal Data Mixture Extrapolation

When we define the data mixture space as $\mathcal{P}_{fix}$ and employ MLP as the fitting function, the optimization objective in Eq. 1 can be reformulated as follows:

$$\mathbf{p}^*_{\text{fix}} = \underset{\mathbf{p}\in\mathcal{P}_{\text{fix}}, t\leq T}{\arg\max} \frac{1}{k}\sum_{j=1}^{k} f^j_{MLP}(\mathbf{p}, t; \theta_0), \tag{4}$$

where j denotes jth downsteam task. Given the negligible inference cost of MLP models, DaMo can efficiently extrapolate the optimal data mixture. We first iterate through all possible data mixtures in the $\mathcal{P}_{fix}$ space to predict downstream task performance scores. Subsequently, we sort these predicted scores and select the top-k highest-scoring mixtures to train our MLLM. This approach enables us to systematically identify the optimal data mixture without exhaustive empirical testing. The complete algorithm pseudocode is provided in Appendix D.

5 Experiments

5.1 Experiments Settings

Training datasets Please refer to Appendix B.

Implementation Details We used InternVL2.5-4B (Chen et al., 2024b) as the base model. Please refer to Appendix C.1 for details.

Baseline We select baselines to cover representative practices in industrial deployment and state-of-the-art (SOTA) methods. *Uniform Mixture* and *Natural Mixture* serve as simple baselines: the former assigns equal weights to all datasets, while the latter uses sampling weights proportional to dataset size, both representing common unoptimized defaults in practice. *Random Mixture* provides a strong heuristic baseline by exhaustively evaluating 250 diverse mixture weights. Additionally, we compare with *Data Mixing Laws (DML)* (Ye et al., 2024), a recent SOTA method that fits loss-based exponential functions to derive optimal mixtures.

Downstream Task Evaluation Besides PhoneAgentBench, we further evaluated our method on four widely used open-source benchmarks to verify generalization, including BFCL-V3 (Patil et al., 2025a), MME-perception (Fu et al., 2023), MME-reasoning (Yuan et al., 2025) and OCRBench (Liu et al., 2024d). All metrics of these benchmarks are expressed as percentages (0-100%), with higher values indicating superior performance.

5.2 Fitting Score of Neural Network

To efficiently characterize the high-dimensional data mixture space $\mathcal{P}_{\text{fix}}$ (a 12-dimensional probability simplex) with limited samples, we adopt a **boundary-aware stratified random sampling strategy**, rather than pure random sampling. Motivated by the boundary effect in high-dimensional spaces, where most volume concentrates near the simplex boundaries (regions with most coordinates near zero) (Komeilizadeh et al., 2023), our strategy consists of two steps: (1) We first sample dataset combinations (selecting k datasets from 12, $1 \leq k \leq 12$) with smoothed probabilities to up-weight both sparse (small k) and dense (large k) combinations; (2) For each selected combination, we generate valid mixing ratios via random perturbations around uniform proportions to ensure diversity. This structured sampling enables us to span the entire $\mathcal{P}_{\text{fix}}$ space with a small number of samples, avoiding the curse of dimensionality of grid search and the high cost of sequential search methods.

As analyzed theoretically in Section 4.1, for $m = 12$ training datasets and a batch size of $b = 16$, the discrete space $\mathcal{P}_{\text{fix}}$ encompasses approximately 1.3×10^7 potential data mixtures. This combinatorial explosion poses a significant challenge for performance prediction. To determine the minimum sample size N_s required to characterize this vast space, we gradually increased the training samples for the MLP. As shown in Table 1, even with only 250 mixture samples, the MLP achieves an R^2 (Wright, 1921) score of 0.81 in 10-fold cross-validation. This indicates that the MLP can accurately extrapolate the performance landscape from a very sparse sampling set.

The theoretical justification for this sparse sampling efficiency lies in the approximation properties of L-Lipschitz functions(Kolmogorov and Tikhomirov, 1959; Wainwright, 2019). According to the theory of metric entropy, the sample complexity N_s required to approximate such a function with a uniform error ε is governed by the covering number, which scales as $N_s \propto (L/\varepsilon)^m$(Kolmogorov and Tikhomirov, 1959; Yarotsky, 2017). Crucially, this bound is determined by the **intrinsic dimension** m and the **hypersurface smoothness** (Lipschitz constant L), rather than the total cardinality of the discrete domain $|\mathcal{P}_{\text{fix}}|$(Anthony and Bartlett, 1999; Wainwright, 2019). Our empirical results (high R^2 with small N_s) suggest that the mapping function between downstream task performance and data mixture is inherently smooth (possessing a small L). This smoothness allows a limited number of representative samples to effectively capture the global topology, decoupling the required sampling size from the search space's combinatorial explosion.

Mixture number	H20 hours	Score (R^2)
50	872	0.58
100	1817	0.57
150	2581	0.74
200	3521	0.78
250	4225	0.81

Table 1: MLP fitting dynamics.

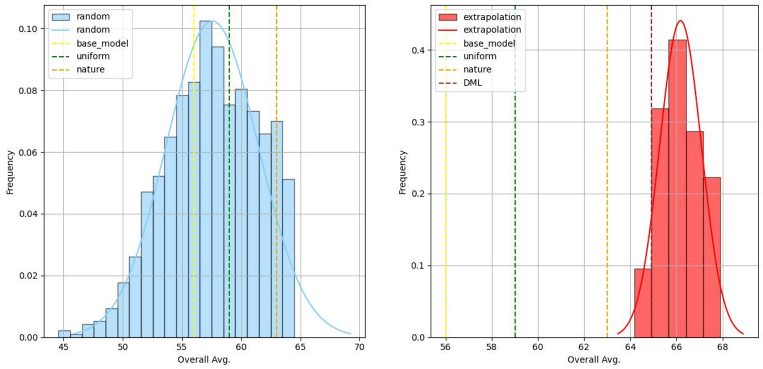
5.3 Downstream Task Performance of Unseen Data Mixtures

Through selecting mixture with top-1 predicting score on PhoneBenchAgent and open-source benchmarks to train MLLM, we obtain the performance on PhoneBenchAgent and open-souce benchmarks, as shown in Table 2. DaMo achieves 23.35% improvement over the native model (without SFT) on PhoneAgentBench. When general capabilities are concurrently considered, DaMo yields an overall average score improvement of 13.73% over the native model. On PhoneAgentBench, DaMo surpasses the previous best-performing Natural mixture by 2.85%. Across overall datasets, it outperforms best-performing

Method	MT-Plan	APP-Rec	MM-RR	ACU	MM-NER	Mobile-FC	OS Avg.	PAB Avg.	Overall Avg.	GPU hours
w/o SFT	20.00	6.00	65.38	39.18	**84.08**	**54.31**	**68.77**	44.83	54.40	/
uniform	54.50	**56.00**	44.62	**86.37**	81.71	45.92	55.76	61.52	59.22	21
natural	47.00	46.00	86.15	83.10	79.83	49.88	59.95	65.33	63.18	21
random(250)	45.00	43.00	83.08	82.77	80.26	47.32	66.36	63.57	64.89	4225
DML	52.00	43.00	85.38	85.72	80.01	42.66	65.48	64.80	65.07	4244
DaMo	**55.50**	51.00	**86.15**	85.30	83.34	47.79	68.05	**68.18**	**68.13**	4242
DaMo(*)	**62.00**	**67.00**	**90.00**	**88.37**	**85.84**	**64.80**	/	/	/	4242

Table 2: Main results on PhoneAgentBench and open-source benchmarks by using top-1 data mixture to train MLLM, predicted by DaMo on PhoneAgentBench and open-source benchmarks. Random(250) denotes the metric is obtained by getting the best performance among 250 fine-tuned models with different random mixtures.

$*$: These scores correspond to different checkpoints, which are optimized by DaMo on a single task.



(a) Random data mixtures

(b) Predicted top 50 data mixtures

Figure 4: Probability distributions of overall average scores across different checkpoints.

Method	BFCL-V3	MME-perception	MME-reasoning	OCRBench	OS Avg.
w/o SFT	29.32	83.82	79.42	82.50	68.77
Uniform	34.69	58.63	64.91	64.80	55.76
Natural	31.41	75.47	67.01	65.90	59.95
Random(250)	33.59	81.11	74.55	78.20	66.36
DML (Ye et al., 2024)	25.47	83.31	76.34	76.8	65.48
DaMo	43.15	84.53	80.94	83.60	**73.06**
DaMo ($*$)	**47.43**	**85.12**	**82.54**	**83.90**	/

Table 3: Main results on open-source benchmarks of MLLMs trained by predicted optimal data mixture on open-source benchmarks.

Method	BFCL-V4	MMBench	DocVQA	MMMU
w/o SFT	17.86	78.79	**91.08**	49.33
Uniform	22.12	76.24	88.73	50.22
Natural	22.94	76.70	88.96	51.33
Random(250)	20.07	76.63	88.19	50.67
DML (Ye et al., 2024)	19.01	75.15	86.56	51.33
DaMo	**28.10**	**79.18**	91.04	**52.44**

Table 4: Main results of DaMo on unseen datasets.

Model	w/o SFT	Uniform	Natural	DML	DaMo (orig.)	DaMo (lin.)
Qwen2.5VL-3B-Inst.	56.25	65.15	64.82	65.03	68.02	**68.66**
Qwen2.5VL-7B-Inst.	59.43	67.48	65.99	66.37	67.79	**69.09**
InternVL3-14B	67.84	63.56	67.8	66.45	68.86	**69.75**

Table 5: Main results of transferability testing on PhoneAgentBench and open-source Benchmarks.

method, DML (Ye et al., 2024), by 3.06%. This performance advantage substantiates the effectiveness of fitting the relationship between data mixtures and downstream performance. When predicting the top-1 mixture on a single dataset, DaMo(*) further outperforms DaMo. This is because DaMo(*) is dedicated to predicting optimal data mixtures for single task, thereby mitigating the performance compromises inherent in multiple tasks.

Figure 4(a) shows the score distribution of Random(250). We can observe two critical characteristics: (1) The absence of a right-side long tail indicates that excellent data mixtures are extremely sparse. (2) The performance of random mixture is predominantly mediocre, and baseline methods (vertical dashed line) show no discernible advantage, demonstrating the inefficiency of heuristic approaches. We used DaMo to predict across $\mathcal{P}_{\text{fix}}$ space, selected the top 50 data mixtures with the best predicted performance, and conducted actual training and evaluation on MLLM. The score distribution of their performance is shown in Figure 4(b), which indicates that DaMo successfully identifies data mixtures with significantly higher average scores compared to all the baselines.

To study the generalization of DaMo on general tasks, we employ DaMo to predict MLLM's performance only on open-source benchmarks, and use the top-1 data mixture to train MLLM, reporting the results in Table 3. It can be observed that our DaMo achieves remarkably superior performance across all open-source benchmarks compared to baselines. Also, it can be observed that focusing on task-specific objectives leads to significantly greater improvements. This is clearly demonstrated by the performance growth from 34.69% obtained by Uniform mixture to 47.43% on the BFCL-V3 benchmark, implemented by DaMo (BFCL-V3) which predicts the performance on BFCL-V3 benchmark only to search optimal data mixture. Crucially, this enhancement is sustained even in the absence of any task-curated training data. We posit that the observed performance benefit is fundamentally driven by DaMo exploring optimal mixtures, which orchestrates a balanced advancement across both specialized and generalizable capabilities.

5.4 Cost-Benefit Analysis

We conduct a GPU-hour analysis in Table 2 to provide a clear cost comparison on NVIDIA H20 GPUs: Uniform/Natural mixtures require only 21 GPU hours for one round of fine-tuning. For Random(250), the total cost of 4225 GPU hours is primarily attributed to the training of 250 data mixture samples. Building on the same 250-mixture sample cost as Random(250), both DML and DaMo add an additional fine-tuning run with their respective optimal mixture ratios, resulting in total costs of 4244 GPU hours and 4242 GPU hours, respectively.

At the same 4200+ hour budget, DaMo outperforms Random(250) and DML by 3.24% and 3.06% on average, with striking gains of 9.56% and 17.68% on the BFCL-V3 benchmark. For low-budget Uniform/Natural mixtures, extending training from 21 to 63 GPU hours yields no performance improvement due to model convergence, indicating their suboptimality stems from poor data ratio design rather than insufficient compute. Notably, DaMo introduces no extra inference overhead for mobile deployment, as the final model size remains unchanged. The one-time sampling cost is fully amortized in large-scale industrial deployment, and DaMo can be seamlessly transferred to other benchmarks and models without additional overhead, as validated in Table 4 and Table 5.

5.5 Results on Unseen Datasets

To assess DaMo's performance on unseen data, we evaluated the model trained using its predicted top-1 data mixture on unseen open-source benchmarks. Table 4 lists the performance of DaMo as well as other methods on BFCL-V4 (Patil et al., 2025b), MMBench (Yuan Liu, 2023), DocVQA (Mathew et al., 2021), and MMMU (Yue et al., 2024). Compared to w/o SFT, DaMo achieves substantial improvements on BFCL-V4 and MMMU, comparable performance on MMBench and DocVQA. The result indicates that DaMo improves model performance on unseen datasets. Moreover, DaMo presents significantly superior performance than Uniform, Natural, Random, and DML mixtures on all datasets, which demonstrates the superiority of DaMo predicting optimal data mixture for unseen datasets.

5.6 Extension to Other Models

We are concerned with the effective generalization of the DaMo to other models. Most current works on data mixture during the pre-training phase assume that data mixture strategies can be directly transferred from smaller models to larger ones (Xie et al., 2023a), but their applicability in the supervised fine-tuning phase remains unverified. To this end, we con-

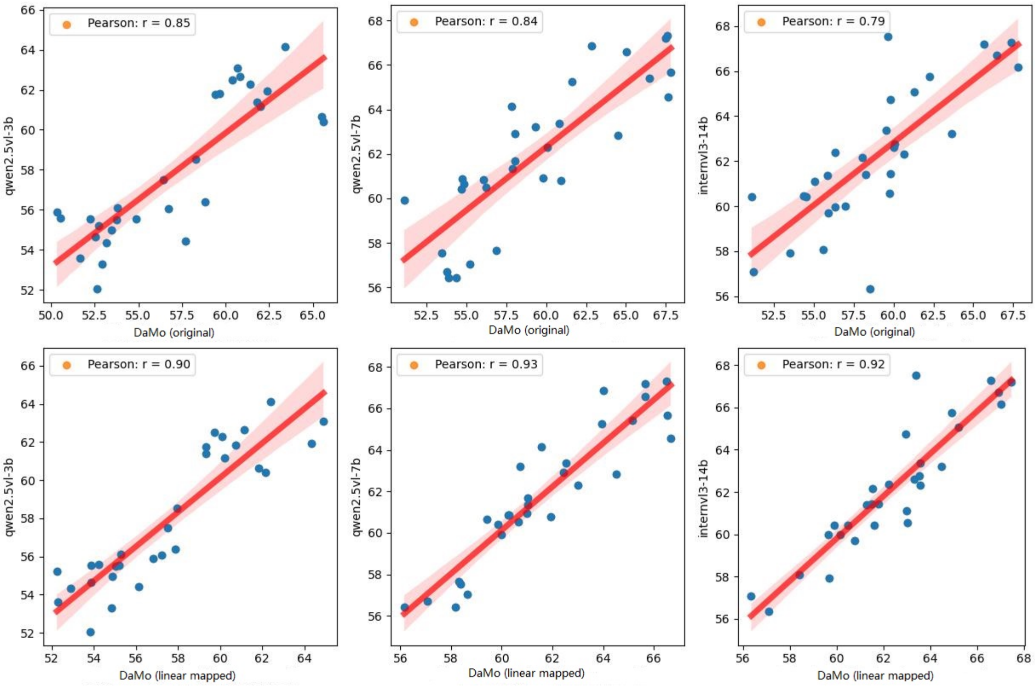


Figure 5: Transferability Analysis. Top: Scatter plot comparing the predicted overall average scores by the original DaMo against the actual scores of target models. Bottom: Apply linear-mapped correction to DaMo.

ducted experiments on transferring DaMo obtained from InternVL2.5-4B to Qwen2.5VL-3B-Instruct, Qwen2.5VL-7B-Instruct (Bai et al., 2025b), and InternVL3-14B (Zhu et al., 2025) with zero or minimal extra training cost.

As outlined in Table 5, directly using the predicted best data mixture for InternVL2.5-4B to train Qwen2.5VL-3B, Qwen2.5VL-7B, and InternVL3-14B, DaMo still outperforms Uniform, Natural, and DML by 0.31%∼2.87% score for all models, demonstrating the stable transferability of DaMo and it is more efficient than other methods. Note that we omit Random mixture for comparison as it needs large training resource for a new model to get a good mixture. To mitigate biases caused by discrepancies in model capabilities, we fit a compensating linear layer using only 20 calibration samples for each new model. The linear-mapped DaMo is defined as $g = f(.)\mathbf{w} + b$, termed as DaMo (lin). It can be seen in Table 5 that DaMo (lin) further improves Uniform, Natural and DML by 1.61%∼3.51% score.

Figure 5 shows the Pearson correlation coefficients (r) between the predicted overall average scores by DaMo and the actual scores of target models. The coefficients r are generally above 0.75, demonstrating the robust cross-model applicability of DaMo even without extra training cost. This suggests that optimal mixtures identified for the base model likely remain near-optimal for the target models. After applying linear mapping, the discrepancies between models are reduced, leading to a further enhancement in correlation with r increasing to above 0.9.

6 Conclusion

In this paper, we present the Data Mixing Optimizer (DaMo) to optimize data mixtures in multitask fine-tuning of multimodal large language models. By introducing downstream task performance prediction with neural network-based modeling, DaMo can predict model performance for any given data mixture. To support comprehensive evaluation, we introduce PhoneAgentBench for evaluation of multimodal large language models on phone agentic tasks. Moreover, DaMo can be extended to other models and tasks. Experimental results demonstrate the efficacy of DaMo not only on PhoneAgentBench, but also on general benchmarks, outperforming state-of-the-art methods.

Limitation

Due to the limited computing resources and time, we can not test the method's effectiveness on recently released large models, e.g., Qwen3-VL-235B-A22B-Instruct (Bai et al., 2025a).

References

Abhishek Shrivastava. 2023. sentiment-analysis-dataset. https://www.kaggle.com/datasets/abhi8923shriv/sentiment-analysis-dataset. Accessed: 2025-03-12.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Alon Albalak, Liangming Pan, Colin Raffel, and William Yang Wang. 2023. Efficient online data mixing for language model pre-training. *arXiv preprint arXiv:2312.02406*.

Martin Anthony and Peter L Bartlett. 1999. *Neural network learning: Theoretical foundations*. Cambridge University Press.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025a. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025b. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Kinjal Basu. 2024. Bridging knowledge gaps in llms via function calls. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 5556–5557.

David Castillo-Bolado, Joseph Davidson, Finlay Gray, and Marek Rosa. 2024. Beyond prompts: Dynamic conversational benchmarking of large language models. *Preprint*, arXiv:2409.20222.

Mingyang Chen, Haoze Sun, Tianpeng Li, Fan Yang, Hao Liang, Keer Lu, Bin Cui, Wentao Zhang, Zenan Zhou, and Weipeng Chen. 2024a. Facilitating multi-turn function calling for llms via compositional instruction tuning. *arXiv preprint arXiv:2410.12952*.

Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, Kai Chen, and 1 others. 2023. T-eval: Evaluating the tool utilization capability of large language models step by step. *arXiv preprint arXiv:2312.14033*.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, and 1 others. 2024b. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. 2024. Seeclick: Harnessing gui grounding for advanced visual gui agents. *arXiv preprint arXiv:2401.10935*.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, and 1 others. 2023. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*.

Longxi Gao, Li Zhang, Shihe Wang, Shangguang Wang, Yuanchun Li, and Mengwei Xu. 2024. Mobileviews: A large-scale mobile gui dataset. *arXiv preprint arXiv:2409.14337*.

Ce Ge, Zhijian Ma, Daoyuan Chen, Yaliang Li, and Bolin Ding. 2024. Bimix: Bivariate data mixing law for language model pretraining. *arXiv preprint arXiv:2405.14908*.

Longling Geng and Edward Y Chang. 2025. Realmbench: A real-world planning benchmark for llms and multi-agent systems. *arXiv preprint arXiv:2502.18836*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shuhao Gu, Jialing Zhang, Siyuan Zhou, Kevin Yu, Zhaohu Xing, Liangdong Wang, Zhou Cao, Jintao Jia, Zhuoyi Zhang, Yixuan Wang, Zhenchong Hu, Bo-Wen Zhang, Jijie Li, Dong Liang, Yingli Zhao, Yulong Ao, Yaoqi Liu, Fangxiang Feng, and Guang Liu. 2024. Infinity-mm: Scaling multimodal performance with large-scale and high-quality instruction data. *Preprint*, arXiv:2410.18558.

Hamed Rahimi. 2024. Sujetfinancevision10k. https://huggingface.co/datasets/sujet-ai/Sujet-Finance-Vision-10k. Accessed: 2025-03-12.

Chen Huang, Shuangfei Zhai, Walter Talbott, Miguel Angel Bautista, Shih-Yu Sun, Carlos Guestrin, and Josh Susskind. 2019. Addressing the loss-metric mismatch with adaptive loss alignment. *Preprint*, arXiv:1905.05895.

Tim Ingold. 2021. *The perception of the environment: essays on livelihood, dwelling and skill*. routledge.

Berivan Isik, Natalia Ponomareva, Hussein Hazimeh, Dimitris Paparas, Sergei Vassilvitskii, and Sanmi Koyejo. 2024. Scaling laws for downstream task performance of large language models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Andrei N Kolmogorov and Vladimir M Tikhomirov. 1959. ε-entropy and ε-capacity of sets in function spaces. *Uspekhi Matematicheskikh Nauk*, 14(2):3–86.

Koushyar Komeilizadeh, Arne Kaps, and Fabian Duddeck. 2023. Isovolumetric adaptations to space-filling design of experiments. *Optimization and Engineering*, 24(2):1267–1288.

Hao Li, Chenghao Yang, An Zhang, Yang Deng, Xiang Wang, and Tat-Seng Chua. 2024a. Hello again! llm-powered personalized agent for long-term dialogue. *arXiv preprint arXiv:2406.05925*.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. 2025a. Screenspot-pro: Gui grounding for professional high-resolution computer use. *arXiv preprint arXiv:2504.07981*.

Lingyu Li, Yixu Wang, Haiquan Zhao, Shuqi Kong, Yan Teng, Chunbo Li, and Yingchun Wang. 2024b. Reflection-bench: probing ai intelligence with reflection. *arXiv preprint arXiv:2410.16270*.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. Api-bank: A comprehensive benchmark for tool-augmented llms. *Preprint*, arXiv:2304.08244.

Yuan Li, Zhengzhong Liu, and Eric Xing. 2025b. Data mixing optimization for supervised fine-tuning of large language models. *arXiv preprint arXiv:2508.11953*.

Liang Xu. 2024. Superclue-agent. `https://github.com/CLUEbenchmark/SuperCLUE-Agent`. Accessed: 2025-03-12.

P Liashchynskyi. 2019. Grid search, random search, genetic algorithm: A big comparison for nas. *arXiv preprint ArXiv:1912.06059*.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. 2024a. Regmix: Data mixture as regression for language model pre-training. *arXiv preprint arXiv:2407.01492*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, and 1 others. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, and 1 others. 2024b. Visualagentbench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*.

Yanming Liu, Xinyue Peng, Jiannan Cao, Shi Bo, Yuwei Zhang, Xuhong Zhang, Sheng Cheng, Xun Wang, Jianwei Yin, and Tianyu Du. 2024c. Tool-planner: Task planning with clusters across multiple tools. *arXiv preprint arXiv:2406.03807*.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. 2024d. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12).

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

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. *Preprint*, arXiv:2311.12983.

Paul Michel, Sebastian Ruder, and Dani Yogatama. 2021. Balancing average and worst-case accuracy in multitask learning. *arXiv: Learning,arXiv: Learning*.

minyang. 2024. invoices-and-receipts_ocr_v1. `https://huggingface.co/datasets/mychen76/invoices-and-receipts_ocr_v1`. Accessed: 2025-03-12.

Niccolò Zanichelli. 2024. arxiv-ocr-v0.1.2. `https://huggingface.co/datasets/nz/arxiv-ocr-v0.1.2`. Accessed: 2025-03-12.

Haojie Pan, Zepeng Zhai, Hao Yuan, Yaojia Lv, Ruiji Fu, Ming Liu, Zhongyuan Wang, and Bing Qin. 2023. Kwaiagents: Generalized information-seeking agent system with large language models. *arXiv preprint arXiv:2312.04889*.

Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025a. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025b. The berkeley function calling leaderboard (bfcl) v4: From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. 2025c. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

qgyd2021. 2024a. chinese_ner_sft. https://huggingface.co/datasets/qgyd2021/chinese_ner_sft. Accessed: 2025-03-12.

qgyd2021. 2024b. few_shot_ner_sft. https://huggingface.co/datasets/qgyd2021/few_shot_ner_sft. Accessed: 2025-03-12.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *Preprint*, arXiv:2307.16789.

Shiori Sagawa, PangWei Koh, Tatsunori Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks. *International Conference on Learning Representations,International Conference on Learning Representations*.

shibing624. 2023. sharegpt_gpt4. https://huggingface.co/datasets/shibing624/sharegpt_gpt4. Accessed: 2025-03-12.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2998–3009.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2023. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *Advances in Neural Information Processing Systems*, 36:38975–38987.

Martin J Wainwright. 2019. *High-dimensional statistics: A non-asymptotic viewpoint*. Cambridge University Press. See Chapter 5 on Metric Entropy and Covering Numbers.

Xuehui Wang, Zhenyu Wu, JingJing Xie, Zichen Ding, Bowen Yang, Zehao Li, Zhaoyang Liu, Qingyun Li, Xuan Dong, Zhe Chen, and 1 others. 2025. Mmbench-gui: Hierarchical multi-platform evaluation framework for gui agents. *arXiv preprint arXiv:2507.19478*.

Yiqi Wang, Wentao Chen, Xiaotian Han, Xudong Lin, Haiteng Zhao, Yongfei Liu, Bohan Zhai, Jianbo Yuan, Quanzeng You, and Hongxia Yang. 2024. Exploring the reasoning abilities of multimodal large language models (mllms): A comprehensive survey on emerging trends in multimodal reasoning. *arXiv preprint arXiv:2401.06805*.

Sewall Wright. 1921. Correlation and causation. *Journal of agricultural research*, 20(7):557.

Yuxin Xiao, Shujian Zhang, Wenxuan Zhou, Marzyeh Ghassemi, and Sanqiang Zhao. 2024. Sftmix: Elevating language model instruction tuning with mixup recipe. *arXiv preprint arXiv:2410.05248*.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2023a. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2023b. Doremi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems*, volume 36, pages 69798–69818. Curran Associates, Inc.

Shiming Xie, Hong Chen, Fred Yu, Zeye Sun, and Xiuyu Wu. 2024. Minor sft loss for llm fine-tune to increase performance and reduce model deviation. *arXiv preprint arXiv:2408.10642*.

Dmitry Yarotsky. 2017. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. 2024. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*.

Jiakang Yuan, Tianshuo Peng, Yilei Jiang, Yiting Lu, Renrui Zhang, Kaituo Feng, Chaoyou Fu, Tao Chen, Lei Bai, Bo Zhang, and 1 others. 2025. Mme-reasoning: A comprehensive benchmark for logical reasoning in mllms. *arXiv preprint arXiv:2505.21327*.

Ruifeng Yuan, Shichao Sun, Yongqi Li, Zili Wang, Ziqiang Cao, and Wenjie Li. 2023. Personalized large language model assistant with evolving conditional memory. *arXiv preprint arXiv:2312.17257*.

Yuanhan Zhang Bo Li Songyang Zhnag Wangbo Zhao Yike Yuan Jiaqi Wang Conghui He Ziwei Liu Kai Chen Dahua Lin Yuan Liu, Haodong Duan. 2023. Mmbench: Is your multi-modal model an all-around player? *arXiv:2307.06281*.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, and 3 others. 2024. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of The IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Jin Zhang, Jianyang Xue, and Bochao Cao. 2024. Improving agent performance in fluid environments by perceptual pretraining. *Physics of Fluids*, 36(12).

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, and 1 others. 2025. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*.

Tong Zhu, Daize Dong, Xiaoye Qu, Jiacheng Ruan, Wenliang Chen, and Yu Cheng. 2024. Dynamic data mixing maximizes instruction tuning for mixture-of-experts. *arXiv preprint arXiv:2406.11256*.

A Evaluation datasets

Table 6 summarizes the statistic information of the tasks in PhoneAgentBench. To guarantee the faithfulness of the proposed PhoneAgentBench, we implemented a rigorous workflow encompassing data filtering, synthetic data generation, and manual verification. Details information about our evaluation datasets for PhoneAgentBench are as follows.

Dataset	Evaluation ability	Data size
MT-Plan	Mulitmodal Task Planning	100
MM-RR	Multimodal Reference Resolution	130
ACU	Agent Context Understand	100
MM-NER	Multimodal Named Entity Recognition	376
APP-Rec	APP Recognition	100
Mobile-FC	Mobile Function Calling	429

Table 6: The statistics of PhoneAgentBench.

A.0.1 MultiModal Task Planning

We introduce the two metrics of complexity and diversity to evaluate the quality of the benchmark for the task planning.

- **Complexity**: The answer of MT-Planning can be viewed as a directed acyclic graph (DAG), where each subtask is a node and the dependency relationship between subtasks are edges. Thus, complexity can be expressed as n_{edge}/n_{node}.
- **Diversity**: The higher the similarity between queries in a dataset, the lower the diversity of that dataset. We use Rough-L to calculate the similarity between every pair of queries, and diversity can be expressed as $1 - \frac{1}{N(N-1)/2} \sum_{i \neq j} \text{Rough-L}(q_i, q_j)$.

Based on this, we compared the data complexity and diversity between MT-Plan and T-Eval planning.

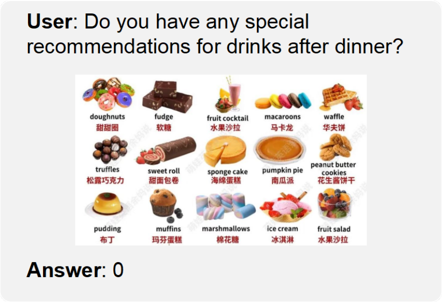
Benchmark	complexity↑	diversity ↑
MT-Plan	0.661	0.82
T-Eval (Chen et al., 2023)	0.122	0.73

Table 7: Benchmark metrics.

A.0.2 MultiModal Reference Resolution

The MultiModal Reference Resolution (MM-RR) task requires the model to determine whether the current question refers to information in the image, which is a binary classification task.

As shown in Figure 6, the question in Figure 6(a) does not refer to the content in the image, so the answer is 0; while the question in Figure 6(b) refers to the drinks on the shelf in the image, so the answer is 1.



User: Do you have any special recommendations for drinks after dinner?

Answer: 0

(a) A negative sample example.

User: Are there any drinks you like on the shelf?

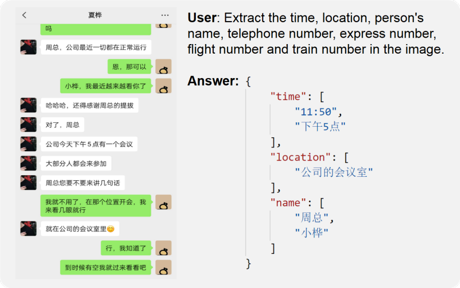
Answer: 1

(b) A positive sample example.

Figure 6: Examples of RR dataset.

A.0.3 Multimodal NER

Multimodal NER (MM-NER) benchmark quantitatively measures MLLMs' ability in understanding and extracting key entities. The dataset comprises 376 image-only samples sourced from Baidu's publicly available image repositories, where each image underwent a rigorous curation process: professional annotators manually filtered the raw visual data to retain high-quality, clearly discernible images, which were subsequently annotated with precise labels focusing on seven critical entity categories—temporal references, geographical locations, personal identifiers, contact numbers, tracking Number, flight Number, train Number to establish a structured benchmark for multimodal entity recognition. We adopt the entity F1-score as the evaluation metric. Figure 7 demonstrates time, location and person extraction from chat logs.



吗

周总，公司最近一切都在正常运行

恩，那可以

小桦，我最近越来越看你了

哈哈哈，还得感谢周总的提拔

对了，周总

公司今天下午 5 点有一个会议

大部分人都会来参加

周总您要不要来讲几句话

我就不用了，在那个位置开会，我来看几眼就行

就在公司的会议室里😊

行，我知道了

到时候有空我就过来看看吧

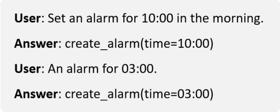
User: Extract the time, location, person's name, telephone number, express number, flight number and train number in the image.

Answer:
```
{
    "time": [
        "11:50",
        "下午5点"
    ],
    "location": [
        "公司的会议室"
    ],
    "name": [
        "周总",
        "小桦"
    ]
}
```

Figure 7: An example of MM-NER dataset.

A.0.4 Mobile Function Call

The Mobile Function Call (Mobile-FC) task is designed to evaluate the ability of MLLMs to call mobile API functions. The task requires the model to select appropriate functions from a given set of application functions to call according to the user's app instruction questions and output the parameters required for the function calls. We define 50 function call interfaces for different scenarios, such as setting an alarm, checking the weather, and setting navigation. The questions in the data are manually constructed by annotators, simulating real-world scenarios of apps on smartphone operating systems and forming complete multi-round dialogues. The evaluation method mainly compares the predicted function names and parameter names with the annotated results. A perfect match scores 1 point; otherwise, 0 points. As shown in the Figure 8, we define the function name create alarm for setting an alarm, with the time field as the input parameter.



User: Set an alarm for 10:00 in the morning.

Answer: create_alarm(time=10:00)

User: An alarm for 03:00.

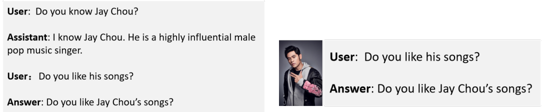
Answer: create_alarm(time=03:00)

Figure 8: An example of Mobile-FC dataset.

A.0.5 Agent Context Understanding

The Agent Context Understanding (ACU) task is used to assess the context-aware dialogue comprehension ability of MLLMs. The data is presented in the form of multi-turn conversations (including text and image).

The model is required to resolve the anaphoric information in the user's final question based on multi-turn conversations or image information, and output a question that contains no anaphora. As shown in the Figure 9, the user asks "Do you like his songs?". If no image is provided, the model needs to determine who "he" refers to based on the historical conversation. Otherwise, the model needs to recognize the person in the image. Model's output is a question that contains no referential information. We use the BLEU of the output answer with the reference answer to evaluate task performance, with scores ranging from 0 to 1.



User: Do you know Jay Chou?

Assistant: I know Jay Chou. He is a highly influential male pop music singer.

User： Do you like his songs?

Answer: Do you like Jay Chou's songs?

User: Do you like his songs?

Answer: Do you like Jay Chou's songs?

(a) Pure-text conversation sample.

(b) Multimodal conversation sample.

Figure 9: Examples of ACU dataset.

A.0.6 APP Recognition

The APP Recognition (APP-Rec) task, similar to the APP-Rec training set, is used to evaluate the ability of MLLMs to identify mobile applications. The model is required to directly output the APP name based on the content of the input mobile APP interface image, as illustrated in the Figure 11. The performance evaluation is conducted by comparing the overlap between the predicted application name and the annotated result. A correct prediction scores 1 point; otherwise, 0 points.

B Training datasets

The open source data includes: ShareGPT4 (shibing624, 2023), NER (composed of Chinese-NER-SFT (qgyd2021, 2024a), Sentiment-Analysis (Abhishek Shrivastava, 2023), and Few-Shot-NER-SFT (qgyd2021, 2024b)), Infinity-MM (Gu et al., 2024), OCR (consisting of Vision-OCR-Financial-Reports-10K (Hamed Rahimi, 2024), Arxiv-OCR-v0.1-SFT (Niccolò Zanichelli, 2024) and Invoices-and-Receipts-OCR-v1 (minyang, 2024)), and SuperCLUE-Agent (Liang Xu, 2024).

The self-built datasets include MultiModal-Instruction-Evolution (MMIE), APP Recognition (APP-Rec), Reference-Resolution (RR), MultiModal-Understanding (MMU), Function-Calling (FC), Task-Planning (TP), and Image-Text-Relevance (ITR), which are primarily derived from data synthesis and real-world industrial scenarios. The size of samples in all the training data is shown in Table 8.

B.0.1 Multimodal Instruction Evolution

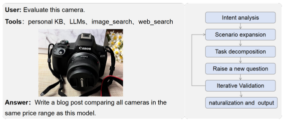
The Multimodal Instruction Evolution (MMIE) task consists of 1.8K pieces of multimodal

Dataset	Source	Data size	Dataset	Source	Data size
MMIE	self-built	1.8k	APP-Rec	self-built	22.8k
MMU	self-built	21.1k	RR	self-built	10.5k
TP	self-built	26.8k	FC	self-built	10.4k
ITR	self-built	9.7k	ShareGPT4	open-source	36k
NER	open-source	8k	Infinity-MM	open-source	37.2k
OCR	open-source	33k	SuperCLUE-Agent	open-source	1.5k

Table 8: Training dataset sizes.

question-answering data. As shown in Figure 10, given an initial query and image with several available tools, the methodology requires the model to generate more sophisticated and diversified questions. The generation pipeline comprises six structured phases:

- **Intent analysis**: Analyze the user's potential needs from multiple perspectives.
- **Scenario expansion**: Expand the scenario to increase the diversity and complexity of the initial question.
- **Task decomposition**: Decompose the scenario into multiple subtasks which can be executed correctly by provided tools.
- **Raise new question**: Propose a new question based on the expanded scenario and subtasks.
- **Iterative Validation**: Evaluate completeness and complexity, where completeness indicates whether the question adequately covers the steps of the subtasks.
- **Naturalization and output**: Refine questions to be more colloquial and output the final result.



User: Evaluate this camera.

Tools：personal KB、LLMs、image_search、web_search

Answer：Write a blog post comparing all cameras in the same price range as this model.

Figure 10: An example of MMIE dataset.

B.0.2 APP Recognition

The APP Recognition (APP-Rec) task consists of 22.8K pieces of multimodal question-answering data, which are composed of images and task instructions. The task requires the model to identify the interface information of mobile apps in the input images and directly output the app names. To obtain diverse app interface data, we install 100 different applications on a mobile phone, such as WeChat, QQ, Little Red Book, Weibo, Alipay, Pinduoduo, Taobao, and TikTok. Annotators are then required to manually capture screenshots of different functional interfaces of each application, which serve as the image source for the APP-Rec task, as illustrated in Figure 11. The default input task instruction is "Identify which app the screenshot belongs to?", and the answer is the name of the app corresponding to the image.

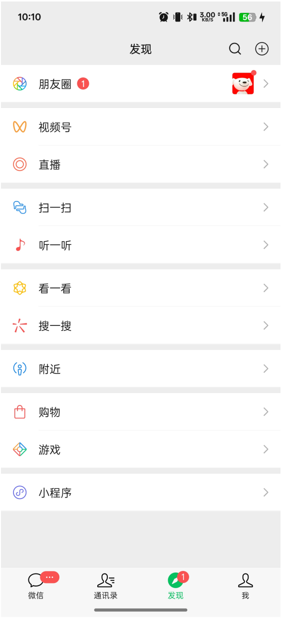


Figure 11: An example of APP-Rec dataset.

B.0.3 Reference Resolution

The Reference Resolution (RR) task corresponds to the MM-RR task in Section A.0.2, which contains 10.5k pieces of multimodal question-answering data. We collect various images containing text information from the internet, with sources including academic papers, test questions, news, company official websites, Wikipedia, etc. Annotators design corresponding questions based on the text content in the given images as positive examples, while negative examples are obtained by replacing the images with different types, as shown in the following Figure 6, which provides one positive and one negative example respectively.

B.0.4 Multimodal Understanding

The Multimodal Understanding (MMU) tasks are consistent with ACU tasks in Section A.0.5. It takes the form of multimodal or text-only multi-round dialogues, with 1-4 rounds and a total of 21.1K samples. The images are sourced from publicly available internet data, covering various fields such as people, animals, plants, architecture, and digital products. The dialogue data is manually constructed by annotators based on the given images, focusing on reference problems. The task requires the model to combine the images and historical dialogue content to rewrite the user's final input text. This is achieved by replacing pronouns or supplementing omitted content to make the text semantically complete.

B.0.5 Function Calling

The Function Calling (FC) task consists of plain text instructions, which requires selecting appropriate tools from given tool set and filling in correct parameters for executing. The tools involve practical mobile applications such as unit conversion, weather inquiry, time calculation, text creation, recipe search, mobile phone bill inquiry, and other 500 types of useful tools. Notably, 90% of the instructions only require the invocation of a single tool.

Here is an example in Figure 12: The user inquires how much 500 US dollars is in Japanese yen, and the answer includes thoughts and actions. The thought process briefly outlines the current step, while the action first provides the name of the selected tool and sets the actual parameters in the action input.

User: I am planning a trip to Japan next month and wonder how much Japanese yen I can exchange for 500 USD.

```
Answer: [
            {
                "Thought": "Check how many Japanese yen can be exchanged for 500 USD.",
                "Action": "exchange_rate",
                "Action Input": {
                    "money": "500",
                    "fromcoin": "USD",
                    "tocoin": "JPY"
                }
            }
        ]
```

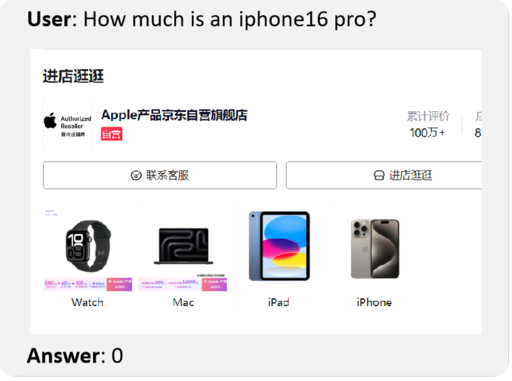
Figure 12: An output example of FC dataset.

B.0.6 Task Planning

The Task Planning (TP) dataset, also targeting tool calling scenarios, places greater emphasis on multi-stage operations with inter-dependent steps compared to FC. It involves 26.8K pieces of multimodal question-answering data. This dataset requires models to properly decompose complex problems into solvable subtasks while ensuring correct tool selection and execution. In multistep scenarios, managing inter-parameter dependencies becomes critical.

The input is a complex question requiring calling apps on moblie phone. Output contains multistep thinking and actions similar to FC, and symbols start with #E are used to receive parameter for cited in subsequent tasks. (as demonstrated in Figure 2).

B.0.7 Image-Text Relevance



User: How much is an iphone16 pro?

Answer: 0

(a) A negative sample example.

User: What's the contact number?

Answer: 1

(b) A positive sample example.

Figure 13: Examples of ITR dataset.

The Image-Text Relevance (ITR) task involves 9.7K pieces of multimodal question-answering data. The task requires the model to analyze the relevance between the question and the image based on the characteristics of the question. If the image is relevant to the question and can be used to answer the user's question, the model should answer 1; otherwise, 0. The images are sourced from publicly available internet data, the same as those used in the MMU task. Annotators manually construct questions related to the image content as positive examples. For example, for images of people, questions about names, works, or family relationships can be asked. Negative examples are constructed by replacing the images with different types, as shown in the following Figure 13, which presents one positive and one negative example respectively.

C Details of Experiments Setting

C.1 Implement Details

We applied a series of experiments to verify the effectiveness of DaMo. Initially, we conducted training and evaluation on InternVL2.5-4B (Chen et al., 2024b) to obtain fitting samples for the MLP. Specifically, we first sampled 250 random data mixtures $\mathbf{p}$ from $\mathcal{P}_{\text{fix}}$. For each mixture, training was performed on 8 NVIDIA H20 GPUs, and checkpoints were saved at 4 distinct training steps—resulting in a total of 1000 checkpoints. All 1000 checkpoints were then evaluated on downstream tasks, which generated 1000 sample points in the format of $(\mathbf{p}, t, \mathbf{s})$. The hyperparameters for training the MLLM are listed in Table 9.

Subsequently, we fitted the MLP on these 1000 sample points. MLP is structured as a two-layer multi-layer perceptron (MLP) built upon `sklearn.MLPRegressor`, where each of the two hidden layers contains 100 neurons. To verify the model's fitting score, we assessed the coefficient of determination (R^2) (Wright, 1921) of DaMo via 10-fold cross-validation. More details of MLP are provided in Table 9.

Then, we utilized DaMo to predict the downstream task performance of unseen data mixtures. Leveraging the low inference cost of the MLP, we conducted performance predictions for all mixtures $\mathbf{p} \in \mathcal{P}_{\text{fix}}$. Among these, the data mixture with the optimal predicted performance was selected for further model training and validation, aiming to obtain actual performance metrics.

Finally, to verify the transferability of DaMo on other models, we extended DaMo (based on InternVL2.5-4B) to Qwen2.5VL-3B-Instruct, Qwen2.5VL-7B-Instruct (Bai et al., 2025b), and InternVL3-14B (Zhu et al., 2025). For these new models, we trained a small number of random mixtures, analyzed the correlation between DaMo's predicted performance and the actual training performance, and meanwhile used DaMo to find the optimal mixtures on the new models to verify whether it still maintains competitiveness compared with the baselines.

Model	**Hyperparameters**	**setting**
MLLMs	AdamW β_1	0.9
	AdamW β_2	0.95
	AdamW ϵ	$1e-6$
	Max Sequence Length	16384
	Batch Size	16
	Gradient Accumulation Steps	8
	Training Steps	1440
	Warmup Steps	144
	Peak Learning Rate	$1e-5$
	Weight Decay	0.1
	Gradient Clipping	1.0
MLP	Input Layer Dimension	12
	Hidden Layer 1 Dimension	100
	Hidden Layer 2 Dimension	100
	Output Layer Dimension	10
	Activation Function	ReLU
	Optimizer	Adam
	Learning Rate	$1e-6$
	Training Steps	1500

Table 9: Hyperparameters of training.

D Algorithm

Algorithm 1 illustrates the complete pipeline of DaMo, with explicit separation between MLP predictor fitting data and final MLLM evaluation data to avoid test set tuning.

E LLM usage

In this paper, we used LLMs to polish the content of the main text and appendices.

Algorithm 1: DaMo: Data Mixing Optimizer for Multimodal LLM Fine-tuning

```
Input:
    $\mathcal{D}$: Training datasets of MLLM (12 source datasets);
    $\mathcal{D}^{test}$: Fixed downstream test set (only for ground-truth downstream task performance labels and
final MLLM evaluation, not used for MLLM or MLP training);
    $\theta_0$: Initial parameters of MLLM;
    $\mathcal{P}$: High-dimensional data mixing ratio space;
    $t$: MLLM Fine-tuning steps;
    $K = 10$: Fold for cross-validation;
    $\mathcal{M}$: Dataset for MLP fitting ($< (\mathbf{p}, t), \mathbf{s} >$ pairs).
Output:
    $\theta^*$: Optimized MLLM with optimal data mixture $\mathbf{p}^*$;
    $\mathbf{s}^*$: Final downstream performance on unseen $\mathbf{p}^*$;
    $R^2$: Average $R^2$ score of MLP via 10-fold cross-validation.

Step 1: Boundary-Aware Stratified Sampling
Initialize $\mathcal{M} \leftarrow \emptyset$
Randomly sample dataset combinations $C_k$ ($1 \leq k \leq 12$) via smoothed probability
    (uphold sparse and dense combinations equally);
Generate valid mixing ratios $\mathbf{p}$ via uniform proportion perturbation;
Construct sampled mixing space $\mathcal{P}_{mlp} \subset \mathcal{P}$;

Step 2: Collect Samples for MLP Dataset
Initialize $\mathcal{M} \leftarrow \emptyset$
foreach $\mathbf{p}^i \in \mathcal{P}_{mlp}$ do
    $\theta_t^i \leftarrow$ Trainer($\mathcal{D}, \mathbf{p}^i, t, \theta_0$);
    $\mathbf{s}^i \leftarrow$ Evaluator($\theta_t^i, \mathcal{D}^{test}$);
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{(\mathbf{p}^i, t, \mathbf{s}^i)\}$;
end

Step 3: MLP Training & Evaluation with 10-Fold Cross-Validation
Initialize total $R^2_{sum} \leftarrow 0$
for $k = 1$ to $K$ do
    Split $\mathcal{M}$ into $\mathcal{M}^k_{\text{train}}$ and $\mathcal{M}^k_{\text{test}}$ (fold $k$);
    $f^k_{MLP} \leftarrow$ fit($\mathcal{M}^k_{\text{train}}$);
    $\hat{\mathbf{s}}^k \leftarrow f^k_{MLP}(\mathcal{M}^k_{\text{test}})$;
    $R^2_k \leftarrow$ Compute $R^2$ between $\hat{\mathbf{s}}^k$ and ground-truth;
    $R^2_{sum} \leftarrow R^2_{sum} + R^2_k$;
end
$R^2 \leftarrow R^2_{sum}/K$; // Average $R^2$ over 10 folds
$f_{MLP} \leftarrow$ fit($\mathcal{M}$); // Train final MLP on all data

Step 4: Optimal Data Mixture Search & Final Evaluation
$\mathbf{p}^* \leftarrow \arg\max_{\mathbf{p} \in \mathcal{P}} f_{MLP}(\mathbf{p}, t)$;
$\theta^* \leftarrow$ Trainer($\mathcal{D}, \mathbf{p}^*, t, \theta_0$);
$\mathbf{s}^* \leftarrow$ Evaluator($\theta^*, \mathcal{D}^{test}$);
return $\theta^*, \mathbf{s}^*, R^2$;
```