CLIVO: DIFFERENTIABLE CLUSTER-BASED INSTRUCTION MIXING WITH LEARNED VALIDATION OBJECTIVES

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal instruction tuning depends on both the quality of individual examples and their composition in the training set. Target-specific data selection must therefore decide which examples are useful and how much of each type of supervision to include. Sample-level rankings leave this composition implicit, while group-level optimization faces three connected challenges: fine-grained units enlarge the search space, predefined validation sets may obscure target-relevant feedback, and fitted mixture-response predictors introduce additional approximation error. We propose CLIVO, which connects semantic allocation, automatically learned validation objectives, and differentiable mixture search. Semantic clusters give distinct instruction contents independently adjustable training shares. To evaluate these choices, CLIVO relates individual examples' merged-proxy losses to real-training probe scores, then selects and weights validation examples to form a target-specific objective. Differentiating this objective through reusable expert-merged proxies updates all shares jointly without a separate response predictor. Selecting 100K instructions from an 8M pool, CLIVO improves the four-target average over the best-performing state-of-the-art data selection baseline by 1.55 points; combining its quotas with ICONS increases this advantage to 1.94 points. Ablations show that moderate-budget experts preserve useful feedback, finer allocation benefits most targets, and sample-level weighting exploits target-relevant response differences obscured by cluster averaging. Semantic visualizations and reusable, modular components further support interpretable and flexible target-specific data selection.

1 INTRODUCTION

Vision-language models (VLMs) have demonstrated strong capabilities in visual understanding, document parsing, multimodal reasoning, and interactive tasks (Comanici et al., 2025; Wang et al., 2025; Bai et al., 2025a; Hong et al., 2025; Huang et al., 2026; Team et al., 2026). Their development combines multimodal representation pretraining with post-training for task adaptation and instruction following (Yin et al., 2025; Team et al., 2025b;a; Radhakrishna et al., 2025; Lin et al., 2024). Within this process, multimodal instruction tuning teaches models to respond to requests about visual inputs using instruction–response examples (Liu et al., 2023a; Dai et al., 2023; Du et al., 2025). Its effectiveness depends on both supervision quality (Luo et al., 2025) and the composition of the training set (Zhang et al., 2025). Large instruction pools cover diverse tasks and response formats (Xu et al., 2024), but can also contain incorrect responses and hallucinated content that degrade model performance (Gou et al., 2025). Data selection helps concentrate training on useful supervision: prior work shows that carefully selected subsets can match full-data performance and even surpass it on individual benchmarks (Liu et al., 2025). Such compact subsets are useful for repeated fine-tuning, where selection can also adapt the training content to the intended application. A document assistant, for example, needs a different balance of text reading, visual recognition, and reasoning from a general visual assistant. The selection problem is therefore not only to identify valuable examples, but also to determine how much of each type of supervision the target model should receive.

Existing data selection methods address these decisions at the sample or group level. *Sample-level methods* rank examples using signals such as gradient influence in ICONS (Wu et al., 2024), necessity in MLLM-Selector (Ma et al., 2025), and importance and diversity in CoIDO (Yan et al., 2026).

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

However, selecting only the highest-scoring examples can concentrate the subset on similar content, reducing diversity and degrading downstream performance (He et al., 2025). Diversity-aware scores help address this risk, but the resulting content shares remain implicit rather than directly optimized for the target. *Group-level methods* expose this choice by partitioning data into sources, domains, or semantic groups and assigning each a training share. Their effectiveness depends on three connected decisions: the allocation *unit*, the evaluation *signal*, and the mixture *learning* rule. For the unit, predefined source partitions bind heterogeneous instructions to a single share, preventing independent adjustment of their contents. Semantic clusters offer finer control, but heuristic allocation based on density and transferability, as in COINCIDE (Lee et al., 2024), is not optimized against measured downstream gains and can favor content that is less useful for a particular target. CLIMB instead searches cluster mixtures using downstream-task validation feedback, but its main setting uses 21 groups and it reports increasing search costs with more groups (Diao et al., 2025). Such coarse search units still tie their internal contents together, leaving the challenge of making **fine-grained composition practical to search**. For the signal, evaluating each mixture through real training and downstream benchmarks is costly, motivating validation-loss proxies. CLIMB uses predefined task validation sets, and CAMEL (Li et al., 2026a) learns a mapping from multiple validation losses to benchmark scores. Both start from separately specified validation collections: **which examples should define the target signal is not itself learned**, and set-level averages remove distinctions among their individual responses. For learning, dynamic reweighting couples mixture updates to ongoing proxy-model training, as in DoReMi (Xie et al., 2023a), making repeated optimization across target and grouping choices costly. Regression-based methods instead evaluate sampled mixtures and fit a response predictor for search (Liu et al., 2024b; Li et al., 2026c; Wang et al., 2026). This requires candidate-labeling computation and introduces **an additional approximation at unseen mixtures**, whose errors can be amplified by selecting predicted optima (Smith & Winkler, 2006).

To address these coupled challenges, we propose **CLIVO** (**CL**uster-based **I**nstruction Mixing with Learned **V**alidation **O**bjectives), which makes fine-grained data composition directly optimizable through a learned target-specific loss. Specifically, **semantic allocation units** extend mixture control to hundred-scale semantic clusters, grouping related content across sources while separating distinct supervision within a source. To evaluate these composition choices, a **learned validation signal** relates individual examples' merged-proxy losses to real probe models' downstream scores, automatically selecting validation examples and learning their nonnegative weights. Reusable expert-merged proxies provide these losses without retraining a model for every candidate mixture (Berasi et al., 2026). Finally, **gradient-based mixture learning** differentiates the resulting objective through fixed experts to update all training shares jointly, turning target-relevant feedback into fine-grained composition decisions.

Across four capability targets, CLIVO selects only 1.25% of an 8M instruction pool and attains 97.2% of the full-data four-target average score with random sampling within clusters. At the same 100K selection budget, it improves on the best-performing state-of-the-art data selection baseline by 1.55 points on average; adding within-quota ranking increases this advantage to 1.94 points. Ablations explain why the components work together: moderate-budget experts preserve informative mixture responses, finer allocation benefits most targets, and sample-level weighting retains and uses target-relevant differences that cluster-level averaging obscures. Further analyses make these composition decisions interpretable through semantic visualizations, demonstrate reuse across targets and search granularities, and show that learned quotas complement existing sample selectors.

Our contributions are:

- **A unified framework for target-aware instruction mixing.** CLIVO connects fine-grained semantic allocation, automatically selected and weighted validation examples, and gradient-based mixture optimization, making training composition directly optimizable without an additional mixture-response predictor.
- **Controlled analysis of the framework's components.** Experiments establish strong fixed-budget selection and show how moderate-budget experts, target-dependent allocation freedom, and preserved sample-level responses support effective mixture optimization.
- **Interpretable, scalable, and flexible data selection.** Semantic visualizations expose composition decisions, shared experts and feedback support reuse across targets and granularities, and modular components accommodate different selection needs. Combining quotas with sample ranking demonstrates this flexibility in practice.

2 Related Work

Sample-level methods. Sample-level methods score individual examples and select a subset according to those scores. Their signals fall into four broad categories. *Quality and alignment* scores assess instruction quality and complexity in DEITA (Liu et al., 2023b), human preferences in Align2LLaVA (Huang et al., 2025), and image–text compatibility in CLIP-Score (Hessel et al., 2021). *Model-response* signals capture necessity in MLLM-Selector and selector confidence in Once-For-All (Dong et al., 2026). *Influence-based* methods estimate training effects: TIVE combines instance influence with task difficulty, ICONS seeks cross-task consensus, and BIDS (Dai et al., 2025) balances task-dependent influence scales in text instruction tuning. *Distribution and diversity* signals address coverage: DSIR (Xie et al., 2023b) resamples toward a reference distribution, PRISM (Bi et al., 2025) uses visual-feature geometry, DataTailor (Yu et al., 2025) combines informativeness, uniqueness, and representativeness, and PreSel (Safaei et al., 2025) clusters images before generating instructions. CoIDO jointly models importance and diversity. These categories can overlap with allocation; TIVE and DataTailor also determine task quotas. CLIVO learns target-specific semantic quotas that can be filled using existing sample-level selectors.

Group-level methods. Group-level methods partition a pool and determine how much training data to draw from each group. They can be classified by how shares are chosen. *Heuristic allocation* converts group statistics into quotas, as in COINCIDE's use of concept–skill density and transferability. *Online adaptive weighting* updates shares during training; DoReMi uses group distributionally robust optimization on reference-relative losses. *Response-model-based search* fits mixture outcomes, using flexible regressors in RegMix and CLIMB or parametric laws in Data Mixing Laws (Ye et al., 2024), BiMix (Ge et al., 2024), and CAMEL. The latter additionally links validation losses to benchmark scores. Expert reuse changes how these response models obtain feedback: MDE (Belenki et al., 2025) constructs features from combined expert predictions, while DeMix and MergeMix evaluate parameter-merged experts to train LightGBM predictors (Ke et al., 2017). *Direct proxy evaluation* compares merged candidates without a fitted mixture-response function (Berasi et al., 2026). Related work on adaptive merging optimizes coefficients using entropy minimization to improve the merged model itself (Yang et al., 2024). CLIVO instead learns which sample losses should evaluate data mixtures and differentiates this objective through fixed experts to optimize training-data shares.

3 Method

In this section, we formulate target-specific subset selection (Section 3.1), then explain semantic allocation (Section 3.2), validation-objective learning (Section 3.3), and gradient-based mixture search (Section 3.4). Figure 1 and Algorithm 1 summarize how these components connect composition choices to real-training feedback and produce a target-specific subset.

3.1 Problem Setup

Given an instruction pool $\mathcal{D}$, an initial VLM θ_0, a target capability d, and a selection budget N, the goal is to select a size-N training subset that maximizes the target score after instruction tuning under a fixed training recipe. For a partition $\{\mathcal{D}_k\}_{k=1}^{K}$, let $\mathbf{w} = (w_1, \ldots, w_K)$ denote the training shares, with $w_k \geq 0$ and $\sum_k w_k = 1$. Integer quotas $n_k(\mathbf{w})$ approximate Nw_k and sum to N. We fix the within-group selection rule π to uniform sampling without replacement, denoted π_{U}, unless stated otherwise. It selects $\mathcal{S}_k = \pi_{\mathrm{U}}(\mathcal{D}_k, n_k(\mathbf{w}))$, yielding the final subset $\mathcal{S}(\mathbf{w}, \pi_{\mathrm{U}}) = \bigcup_k \mathcal{S}_k$.

Let $\theta_{\mathcal{S}} = \mathrm{Train}(\theta_0, \mathcal{S})$ denote the model obtained with the fixed training procedure, and let $R_d(\theta)$ be its target score. With π_{U} fixed, we optimize only $\mathbf{w}$:

$$\mathbf{w}_d^{\star} \in \arg\max_{\mathbf{w} \in \mathcal{W}} F_d(\mathbf{w}; \pi_{\mathrm{U}}), \qquad F_d(\mathbf{w}; \pi_{\mathrm{U}}) = \mathbb{E}_{\mathcal{S}}\left[R_d(\theta_{\mathcal{S}})\right]. \tag{1}$$

The feasible set $\mathcal{W}$ contains mixtures whose quotas can be filled from the pool. The shares $\mathbf{w}$ determine *how many* examples each group contributes, while π_{U} determines *which* examples fill those quotas; the expectation averages over these uniform draws. Fixing π_{U} isolates composition as the optimization variable.

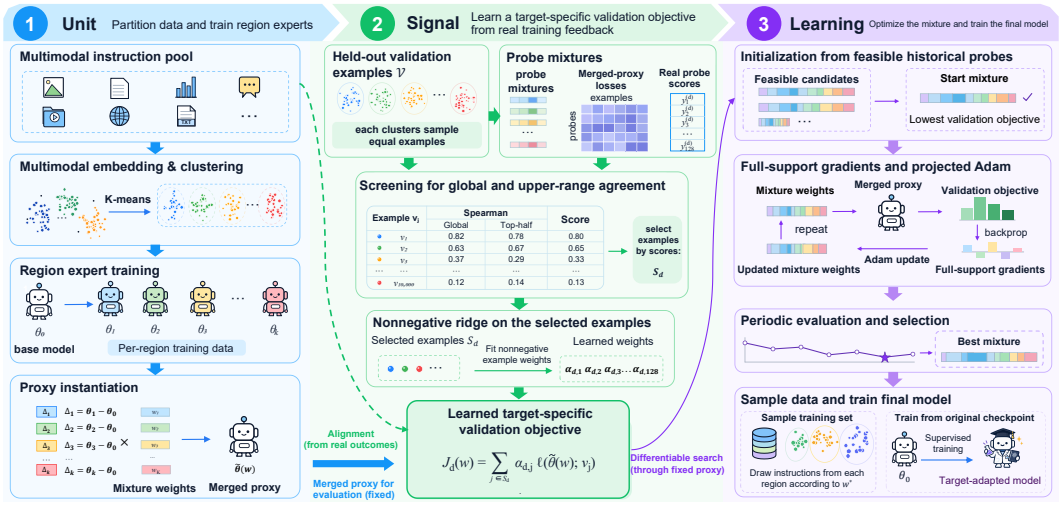

Figure 1: **CLIVO overview.** CLIVO connects semantic allocation, validation-objective learning, and differentiable search to select instruction data for a target capability. **(1) Unit:** an independent multimodal encoder embeds the instruction pool, and K-means organizes it into semantic clusters. Each cluster receives an adjustable training share, allowing distinct contents within a data source to be allocated independently. **(2) Signal:** sampled mixtures define training subsets through uniform within-cluster sampling. Models trained on these subsets provide real downstream scores. In parallel, cluster experts trained from a common initialization are merged at the same mixtures, providing proxy models without additional candidate training. Their per-example losses on a cluster-balanced validation candidate pool are compared with the real scores to identify target-relevant examples. Nonnegative weights combine the selected responses into a learned validation objective. **(3) Learning:** with experts and validation weights frozen, gradients propagate through the merged proxy to jointly update the training shares under budget and capacity constraints. Periodic evaluations of the learned objective retain the lowest-loss mixture. Its shares are converted into integer quotas, filled by within-cluster sampling, to construct the final instruction subset for training from the original base model.

3.2 Unit: Semantic Allocation

A source-level share ties heterogeneous instructions to the same allocation decision. To make semantic content independently adjustable, we encode instructions with a fixed multimodal encoder independent of the VLM being tuned, then apply K-means (McQueen, 1967) to obtain $\{\mathcal{D}_k\}_{k=1}^{K}$. These groups cross source and task boundaries: distinct content within one source can receive different shares, while related content across sources can be allocated together. For nested groups, releasing fixed internal ratios enlarges the feasible allocation family (Section C.2); our tied-share ablation tests the value of this freedom on a fixed expert bank (Section 4.3).

3.3 Signal: Learning the Validation Objective

Real-training feedback for validation selection. To identify validation losses that reflect the target capability, we collect real-training feedback. For each probe mixture $\mathbf{w}_i$, we construct $\mathcal{S}(\mathbf{w}_i, \pi_{\mathrm{U}})$, train from θ_0 under the shared recipe, and evaluate the target score $y_i^{(d)}$. These Y mixture–score pairs supervise validation selection and weighting. Uniform within-cluster sampling keeps the selection policy fixed so that the feedback measures changes in composition (Section C.1).

Low-cost merged-proxy losses. Following prior work on expert-based mixture proxies (Li et al., 2026c; Wang et al., 2026), we train one expert θ_k on each cluster from the common base θ_0 and define $\Delta_k = \theta_k - \theta_0$. A candidate mixture gives

$$\widetilde{\theta}(\mathbf{w}) = \theta_0 + \sum_{k=1}^{K} w_k \Delta_k = \sum_{k=1}^{K} w_k \theta_k. \tag{2}$$

Once the experts are available, each proxy requires merging but no additional training. We form a candidate pool $\mathcal{V} = \{v_j\}_{j=1}^{H}$ with equal numbers of uniformly sampled examples from each cluster, preserving content coverage without large-cluster dominance. At the same Y mixtures as the real probes, we evaluate each candidate's negative loss, $X_{ij} = -\ell(\widetilde{\theta}(\mathbf{w}_i); v_j)$, where ℓ averages cross-entropy over that example's answer tokens. Comparing these per-example responses with the real scores identifies losses that track the target capability.

Algorithm 1 CLIVO: target-specific instruction mixing

Require: $\mathcal{D}, \theta_0, R_d, N; K, m, T, \{\mathbf{w}_i\}_{i=1}^{Y}$
Ensure: Shares $\widehat{\mathbf{w}}_d$; subset $\mathcal{S}_d^\star$

1. Unit: semantic allocation
Embed $\mathcal{D}$; partition into K clusters $\{\mathcal{D}_k\}$.
2. Signal: learned validation objective
Train probes at $\{\mathbf{w}_i\}$; evaluate $\mathbf{y}^{(d)}$.
Train cluster experts $\{\theta_k\}$ from θ_0.
$\mathcal{V} \leftarrow \mathrm{BalancedSample}(\{\mathcal{D}_k\})$
$X_{ij} \leftarrow -\ell(\widetilde{\theta}(\mathbf{w}_i); v_j)$
$S_d \leftarrow \mathrm{Top}_m((\rho_{\mathrm{all,j}} + \rho_{\mathrm{upper\ half,j}})/2)$
Fit Eq.(3); rescale and normalize to $\boldsymbol{\alpha}_d$.
3. Learning: mixture search
Freeze $\{\theta_k\}, \boldsymbol{\alpha}_d$.
$\mathbf{w}^0 \leftarrow \arg\min_{\mathbf{w}_i \in \mathcal{W}} J_d(\mathbf{w}_i)$; $\widehat{\mathbf{w}}_d \leftarrow \mathbf{w}^0$
for $t = 1, \ldots, T$ **do**
$\mathbf{g}^t \leftarrow \nabla_{\mathbf{w}} J_d(\mathbf{w}^{t-1})$ by Eq.(5)
$\mathbf{w}^t \leftarrow \Pi_{\mathcal{W}}(\mathrm{Adam}(\mathbf{w}^{t-1}, \mathbf{g}^t))$
Update best $\widehat{\mathbf{w}}_d$ at checkpoints.
end for
$\mathbf{n} \leftarrow \mathrm{LargestRemainder}(N\widehat{\mathbf{w}}_d)$
$\mathcal{S}_d^\star \leftarrow \bigcup_k \pi_{\mathrm{U}}(\mathcal{D}_k, n_k)$

Selecting and weighting validation examples. For each candidate, we average two Spearman correlations between its negative losses and target scores: one over all fitting probes and one over their higher-scoring half. Retaining the top m candidates as S_d favors responses that track both broad variation and stronger mixtures. Let $\mathcal{I}$ index all fitting probes. We standardize the selected features as $\widetilde{X}_{ij} = (X_{ij} - \mu_j)/s_j$, where μ_j and $s_j > 0$ are their fitting means and standard deviations, and learn nonnegative coefficients:

$$\widehat{\boldsymbol{\beta}}_d = \arg\min_{\boldsymbol{\beta} \geq 0} \left\| \widetilde{X}_{\mathcal{I}, S_d} \boldsymbol{\beta} - (\mathbf{y}_{\mathcal{I}}^{(d)} - \bar{y}_d \mathbf{1}) \right\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2. \tag{3}$$

Here $\bar{y}_d$ is the mean fitting score and $\lambda > 0$ regularizes the coefficients. Screening identifies individually informative responses through rank agreement; fitting combines them to predict centered target scores, with nonnegative coefficients preserving the direction that lower loss predicts higher performance. We return a nonzero fit to the original loss scale and normalize it, $\alpha_{d,j} = (\widehat{\beta}_{d,j}/s_j) / \sum_{h \in S_d} (\widehat{\beta}_{d,h}/s_h)$, obtaining

$$J_d(\mathbf{w}) = \mathcal{L}_d(\widetilde{\theta}(\mathbf{w})) = \sum_{j \in S_d} \alpha_{d,j}\, \ell(\widetilde{\theta}(\mathbf{w}); v_j). \tag{4}$$

Individual weights retain response differences between examples in the same cluster that averaging would remove before learning (Section C.3). Lower J_d corresponds to a higher fitted score, preserving the fitted ranking after conversion (Section C.4). The validation weights $\boldsymbol{\alpha}_d$ define how candidates are evaluated; the training shares $\mathbf{w}$ define their composition. We freeze $\boldsymbol{\alpha}_d$ before search to compare all mixtures under the same objective.

3.4 Learning: Gradient-Based Mixture Search

Before search, we freeze every expert θ_k and all validation weights $\boldsymbol{\alpha}_d$. To accelerate convergence, we initialize $\mathbf{w}^0$ with the lowest-loss merged-proxy mixture among previously evaluated feasible candidates. At each step, the current shares construct a merged proxy, and all positive-weight validation examples contribute to its loss. The chain rule gives

$$\frac{\partial J_d}{\partial w_k} = \left\langle \nabla_\theta \mathcal{L}_d(\theta)|_{\theta = \widetilde{\theta}(\mathbf{w})}, \Delta_k \right\rangle. \tag{5}$$

One parameter-gradient computation supplies all K expert directions, enabling joint Adam updates of $\mathbf{w}$ while model parameters remain fixed. Each update is projected onto $\mathcal{W}$, the simplex with capacity bounds $w_k \leq \min(1, |\mathcal{D}_k|/N)$, to keep the quotas feasible (Section C.8). Periodic full-objective evaluations select the lowest-loss iterate $\widehat{\mathbf{w}}_d$, including the start. Thus, the learned objective directly guides both updates and candidate selection without an additional mixture-response approximation.

Finally, largest-remainder rounding converts $N\widehat{\mathbf{w}}_d$ into integer quotas. Uniform selection π_{U} fills these quotas, and we train from θ_0 on $\mathcal{S}(\widehat{\mathbf{w}}_d, \pi_{\mathrm{U}})$.

4 EXPERIMENTS

In this section, we describe the experimental setup (Section 4.1), compare fixed-budget selection quality (Section 4.2), and conduct ablation studies of CLIVO's components (Section 4.3).

4.1 EXPERIMENTAL SETUP

Model and evaluation. Following the multimodal training approach of LLaVA-1.5 (Liu et al., 2024a), we build a 7B VLM with a Qwen2.5-VL-style architecture (Bai et al., 2025b) and conduct our own multimodal representation pretraining followed by mid-training. The resulting base θ_0 initializes all experts, probes, and final models. Evaluation covers General visual understanding (G: MMBench-EN, MMStar, GQA), Parsing (P: OCRBench, ChartQA), and Knowledge and text-centric reasoning (K: MMLU, MMLU-Pro). Each domain's equally weighted benchmark mean defines a target; Overall (O) averages all seven benchmarks. We learn a separate mixture for each target, and Average summarizes their four target scores. More details are in Section A.

Data pool. We collect approximately 40M instructions from public datasets, covering diverse visual contents, tasks, and response formats. For manageable processing and selection costs, experiments use an 8M candidate pool from this collection. All methods select 100K instructions from the same pool, isolating selection quality at a fixed data budget. Source records, pool provenance, and representative examples are provided in Section E and Section B.7.

Baselines. Our *sample-level baselines* are PPL, CLIP-Score, MM-Selector, PRISM, ICONS, DSIR, and DataTailor, covering model responses, alignment, influence, and distribution-based selection. The *group-based baselines* are COINCIDE for concept–skill allocation and SemDeDup for cluster-based redundancy removal. Random uniformly samples 100K examples; Full trains on all 8M. ICONS and DSIR use CLIVO's selected validation examples to keep reference content consistent while retaining their own scoring rules. CLIVO samples uniformly within clusters to isolate allocation quality; CLIVO + ICONS holds those quotas fixed and ranks within each cluster to test complementarity. Random and CLIVO report three-run means, while the remaining results use single runs. Configurations and run-level details are in Sections B.1 and B.2.

Implementation details. We embed instructions with Qwen3-VL-Embedding-8B (Li et al., 2026b), use $K = 100$ clusters, and train experts with the nominal step-10 budget. Validation learning uses $Y = 128$ real probes and $H = 10{,}000$ candidates, sampled equally across clusters, retaining $m = 64$ examples per target. Five repeated four-fold evaluations assess the rule, with screening and fitting restricted to training folds; the search objective is then refitted on all development probes and frozen. We run 120 projected Adam updates, retain the best periodically evaluated mixture, and train its 100K-example subset for 100 optimizer steps. Budget definitions, optimizer settings, and full protocols are in Sections A and B.

4.2 TARGET-SPECIFIC SELECTION QUALITY

Strong allocation without within-cluster ranking. CLIVO achieves an Average of 69.20 using random samples within learned quotas, exceeding the strongest baseline, ICONS, by 1.55 points (Table 1). It leads the baselines on Overall, Parsing, and Knowledge with scores of 68.13, 79.64, and 61.76. This shows that allocating the budget across semantic content is valuable even without ranking individual training examples. Baseline strengths vary with the selection signal: image–text alignment gives CLIP-Score the strongest baseline Parsing score (79.42), while reference-distribution matching gives DSIR the highest baseline General score (67.56). ICONS and DSIR also obtain competitive results using CLIVO's automatically selected validation examples as their reference, demonstrating compatibility with existing selectors.

Within-cluster ranking further improves learned allocation. With CLIVO's quotas fixed, within-cluster ICONS reaches 68.27 on General and 69.59 on Average, versus 67.28 and 69.20 with random selection. It exceeds global ICONS on all four targets, including a 4.15-point General gain and a 1.94-point Average gain. The two components serve distinct roles: CLIVO determines how much each semantic cluster contributes, while ICONS selects examples within those quotas. Thus, learned allocation can strengthen existing sample-level selectors without changing their scoring rules (Section C.9).

Table 1: **Selection quality at a 100K budget.** O/G/P/K are the four targets; Average is their mean. Random and CLIVO average three runs. Bold/underline: best/second-best among 100K methods.

Method	O	G	P	K	Average
Full (8M)	71.01	69.23	80.87	63.81	71.23
Random	66.61	63.57	77.73	60.05	66.99
PPL	64.26	60.78	75.65	58.09	64.70
CLIP-Score	66.16	62.23	79.42	58.80	66.65
MM-Selector	66.46	64.05	76.96	59.59	66.77
PRISM	67.27	65.73	76.97	59.86	67.46
ICONS	67.32	64.12	79.30	59.88	67.65
DSIR	67.31	<u>67.56</u>	72.62	60.42	66.98
DataTailor	65.72	61.50	77.58	60.20	66.25
SemDeDup	64.00	57.53	77.87	59.84	64.81
COINCIDE	66.20	62.65	78.14	59.60	66.65
CLIVO	**68.13**	67.28	<u>79.64</u>	<u>61.76</u>	<u>69.20</u>
CLIVO + ICONS	<u>68.05</u>	**68.27**	**79.96**	**62.09**	**69.59**

4.3 Ablation Studies

Learned signals guide mixture search. At $K = 100$, all four learned objectives positively rank held-out probes (Table 2). Gradient updates lower the complete proxy loss by 0.056–3.724%, supporting optimization beyond observed mixtures. Final-training means exceed historical probe maxima by 0.30–0.68 points on the domain targets and 0.21 on Overall. We next examine expert feedback, allocation freedom, and validation weighting.

Table 2: **From feedback to final training.** $K = 100$. CV-ρ: held-out Spearman; $t^\star$: best step; D_J: within-run loss decrease (%); B: best probe; $\bar{F}$: three-run mean. $\Delta F = \bar{F} - B$: cross-budget comparison (Section B.5).

Target	CV-ρ	$t^\star$	D_J	B	$\bar{F}$	ΔF
Gen.	0.714	100	3.724	66.600	67.28	+0.68
Pars.	0.461	120	0.293	79.310	79.64	+0.33
Know.	0.596	40	0.056	61.465	61.76	+0.30
Ovrl.	0.591	100	2.040	67.919	68.13	+0.21

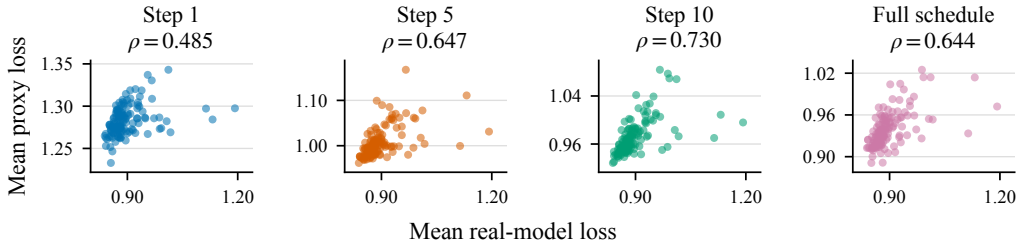

Figure 2: **Expert budget and mixture-response fidelity.** Across training budgets, each point compares mean proxy and real-model losses at the same mixture on a shared 10,000-example panel. ρ: Spearman across 128 mixtures. Budget details: Section B.4.

Moderate expert budgets provide useful mixture feedback. To determine how much expert training the validation objective needs, we compare nominal checkpoint steps $b \in \{1, 5, 10\}$ and a full-training reference on the same mixtures and candidate panel (Figure 2). Proxy–real loss Spearman rises from 0.485 at step 1 to 0.730 at step 10, but falls to 0.644 with full training despite its lower absolute loss. Useful proxies must therefore preserve mixture-dependent responses, not simply minimize average loss. A separate held-out analysis also finds that step 10 improves downstream-score prediction over step 1 across all targets (Table 10). We choose this moderate budget to support both response fidelity and validation selection; budget definitions and protocols are in Section B.4.

Finer allocation supports target-specific composition. To isolate allocation freedom, we tie the same 100 expert coefficients into 20 or 50 parent shares, or optimize all 100 independently, keeping the proxy bank, validation objective, starting composition, and training budgets fixed.

Tying is equivalent to premerging child experts at fixed ratios and optimizing the resulting parent proxies (Section B.5). In Table 3, 100 shares give the highest single-seed scores on most targets: Parsing and Knowledge improve by 0.45 and 0.41 points over 20 shares, while General performs best at 20. Finer control thus helps separate useful supervision for several capabilities, with the required granularity depending on the target. These comparisons reuse the same bank-wide computations; Section 5.2 examines the search costs.

Table 3: **Allocation freedom and search.** $J^\star$: lowest loss; $t^\star$: first best step; $C^\star$: H100-eq. GPU-hours to $t^\star$; F: s0 score (best in bold). Details: Sections B.5 and D.3.

Target	K	$J^\star$	$t^\star$	$C^\star$	F
Gen.	20	0.32639	120	42.40	**67.197**
	50	0.32801	80	$\geq$ 21.92	66.833
	100	0.32145	100	51.73	67.017
Pars.	20	0.85952	0	0.00	79.810
	50	0.85965	100	53.33	80.120
	100	0.85866	120	63.53	**80.260**
Know.	20	0.68731	40	$\geq$ 23.64	61.415
	50	0.68743	120	72.17	61.570
	100	0.68760	40	25.70	**61.825**
Ovrl.	20	0.37324	40	23.05	67.994
	50	0.37097	100	55.33	67.846
	100	0.36904	100	52.37	**68.021**

Sample weighting improves ranking. To test which validation information should be retained and weighted, we compare cluster/sample aggregation with equal/learned weights on the same 64 examples per fitting fold (Table 4). Cluster rules average responses within each represented cluster before weighting; sample rules keep individual responses. This isolates aggregation choices from example selection.

Sample weighting improves held-out global Spearman over equal sample averaging by 0.007–0.121 and outperforms weighted cluster averages, including gains of 0.030 and 0.031 on General and Parsing. The comparison indicates that cluster averaging hides target-relevant differences between individual responses, which learned sample weights can exploit. Weights concentrate on a subset of examples, reflecting unequal contributions to the target signal (Sections B.6 and D.1).

Table 4: **Sample responses and learned weights.** Held-out Spearman, averaged over 5×4 folds with matched 64-example supports. C/S: cluster/sample; E/W: equal/learned weights. Bold: best per target.

Target	C–E	C–W	S–E	S–W
General	0.621	0.684	0.619	**0.714**
Parsing	0.397	0.430	0.379	**0.461**
Knowledge	0.586	0.594	0.589	**0.596**
Overall	0.469	0.569	0.470	**0.591**

5 Interpretable and Modular Data Selection

In this section, we visualize coarse and fine target-specific content choices (Section 5.1), then examine reuse across targets and granularities and modular flexibility (Section 5.2).

5.1 Data Visualization and Management

Targets favor different, concentrated mixtures. We use GPT-based annotations to organize fine-grained capabilities into coarse categories for interpretation (Section B.7). Figure 3 shows distinct preferences: General and Overall emphasize short visual questions in c60, Parsing favors chart-reading c27, and Knowledge favors c36, illustrated by text-based STEM supervision. Allocation is concentrated but target-dependent: the entropy-based effective cluster count is 10.13 for Parsing and 3.02–4.36 for the other targets, out of 100 clusters. Linking these distributions to labels and representative examples makes both the content and concentration of selection decisions inspectable (Table 17).

Within-category contrasts highlight the value of fine-grained allocation. Clusters within the same coarse capability receive markedly different shares (Figure 3(b)). Within *Charts & Data*, Parsing assigns 30.77% to c27, illustrated by direct value reading, and 0.32% to c14, illustrated by a long-form chart response. Within *Knowledge & Science*, Knowledge assigns 59.42% to c36, illustrated by text-only STEM rules, and zero to c20, illustrated by diagram-based mechanics. These choices distinguish supervision forms that a single category share would bind together, providing interpretable support for the allocation ablation in Table 3.

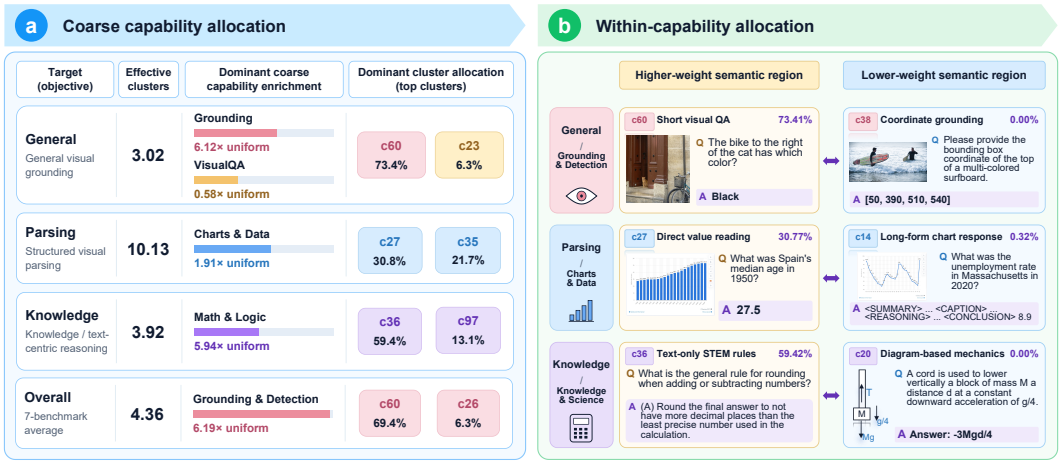

Figure 3: **Target-specific composition, from distributions to examples. (a)** Allocation concentration and dominant semantic regions. **(b)** Paired clusters within the same coarse capability receive different training shares. Percentages give each cluster's share of the complete training budget. Uniform allocates 1% to each of the 100 clusters; × uniform compares a category's total learned share with its share under this equal-cluster allocation. Definitions and cases: Section B.7.

5.2 Scalability and Flexibility

Scalability across targets and allocation granularities. With the base and partition fixed, CLIVO reuses embeddings, experts, trained probes, and cached candidate losses for a new target. Evaluating existing probes on the new benchmark supplies the scores needed to select and weight validation examples, after which gradient search learns the new mixture. Granularity changes likewise reuse the bank: costs in the $K = 20/50/100$ control show no consistent increase with adjustable shares (Table 3). Eq.(5) explains this behavior: each parameter gradient supplies all expert directions, so added shares do not require extra model forward/backward passes per update. Compared with regression, the four gradient searches use 26.6% fewer H100-equivalent GPU-hours than labeling the 4,096 mixtures needed for the strongest tested response-predictor fit. This is a search-cost comparison; shared construction and final training are accounted for separately in Sections C.7 and D.3.

Framework flexibility. CLIVO's modular design allows the encoder, target benchmark, and within-cluster selector to be chosen for the application. Allocation and ranking remain complementary: one determines how much of each content type to include, while the other chooses its representatives. The CLIVO + ICONS results demonstrate this compatibility (Section 4.2 and Table 1), showing how existing selectors can benefit from learned target-specific quotas.

6 Conclusion

We presented CLIVO, a framework for target-specific multimodal instruction selection that connects semantic allocation, automatically learned validation objectives, and differentiable mixture search. By relating sample-level proxy losses to downstream performance, CLIVO learns a target-relevant objective for optimizing fine-grained data composition. Our experiments show that learned allocation improves fixed-budget selection and complements sample ranking, highlighting the importance of designing the selection space and evaluation signal jointly. Ablations further support moderate-budget expert proxies and preserving individual validation responses when learning the objective. Beyond constructing a training subset, CLIVO provides an interpretable, reusable, and modular framework for inspecting data composition and adapting it to different capability needs. Shared experts support reuse across targets, while learned quotas accommodate existing sample selectors. We hope this work encourages the community to study instruction-data quality through both individual examples and their collective composition, providing a foundation for more transparent, target-aware data curation. Future work will explore reducing real-feedback requirements and transferring these selection strategies across models.

AI USE STATEMENT

Generative AI tools assisted with manuscript drafting and editing, literature search, experiment planning, interpretation of results, and theoretical exposition. GPT also provided the capability and format annotations used in the qualitative data analyses described in Section B.7. The authors are responsible for verifying the final wording, citations, derivations, implementation correspondence, and empirical claims.

ETHICS STATEMENT

This work uses publicly available datasets and derived instruction data in accordance with their applicable licenses and terms of use. We document the data sources and processing scope in Section E. Our experiments do not involve recruiting human participants or collecting new personal information. Data processing and reporting follow applicable privacy requirements, and qualitative examples are presented without personally identifying information. CLIVO is intended to support transparent, target-aware data selection for research.

REPRODUCIBILITY STATEMENT

Section 3 specifies the executed sample screening, nonnegative ridge, weight conversion, merge, and constrained search. Sections A and B give the implementation settings, baseline definitions, split discipline, and measurement procedures. Section C provides the derivations. Additional validation and search results appear in Section D, and example-level provenance is documented in Section E.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Ke qin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *ArXiv*, abs/2502.13923, 2025b. URL https://api.semanticscholar.org/CorpusID:276449796.

Lior Belenki, Alekh Agarwal, Tianze Shi, and Kristina Toutanova. Optimizing pre-training data mixtures with mixtures of data expert models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32570–32587, 2025.

Davide Berasi, Matteo Farina, Massimiliano Mancini, and Elisa Ricci. Linear model merging unlocks simple and scalable multimodal data mixture optimization. *arXiv preprint arXiv:2602.04937*, 2026.

Jinhe Bi, Yifan Wang, Danqi Yan, Wenke Huang, Zengjie Jin, Xiaowen Ma, Sikuan Yan, Artur Hecker, Mang Ye, Xun Xiao, et al. Prism: Self-pruning intrinsic selection method for training-free multimodal data selection. *arXiv preprint arXiv:2502.12119*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Qirun Dai, Dylan Zhang, Jiaqi W Ma, and Hao Peng. Improving influence-based instruction tuning data selection for balanced learning of diverse capabilities. In *EMNLP (Findings)*, pp. 7079–7102, 2025.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models

with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

Shizhe Diao, Yu Yang, Yonggan Fu, Xin Dong, Dan Su, Markus Kliegl, Zijia Chen, Peter Belcak, Yoshi Suhara, Hongxu Yin, et al. Climb: Clustering-based iterative data mixture bootstrapping for language model pre-training. *arXiv e-prints*, pp. arXiv–2504, 2025.

Mingkang Dong, Hongyi Cai, Xiwen Lei, Jie Li, Tao Zhang, and Muxin Pu. Once-for-all: A train-once and select-anytime framework for multimodal instruction tuning. *arXiv preprint arXiv:2605.26761*, 2026.

Yifan Du, Hangyu Guo, Kun Zhou, Wayne Xin Zhao, Jinpeng Wang, Chuyuan Wang, Mingchen Cai, Ruihua Song, and Ji-Rong Wen. What makes for good visual instructions? synthesizing complex visual reasoning instructions for visual instruction tuning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 8197–8214, 2025.

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

Yunhao Gou, Hansi Yang, Zhili Liu, Kai Chen, Yihan Zeng, Lanqing Hong, Zhenguo Li, Qun Liu, Bo Han, James Kwok, et al. Corrupted but not broken: Understanding and mitigating the negative impacts of corrupted data in visual instruction tuning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25945–25971, 2025.

Hongyi He, Xiao Liu, Zhenghao Lin, Mingni Tang, Yi Cheng, Jintao Wang, Wenjie Li, Peng Cheng, and Yeyun Gong. Learning from the best, differently: A diversity-driven rethinking on data selection. *arXiv preprint arXiv:2510.18909*, 2025.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7514–7528, 2021.

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

Ailin Huang, Chengyuan Yao, Chunrui Han, Fanqi Wan, Hangyu Guo, Haoran Lv, Hongyu Zhou, Jia Wang, Jian Zhou, Jianjian Sun, et al. Step3-vl-10b technical report. *arXiv preprint arXiv:2601.09668*, 2026.

Hongzhe Huang, Jiang Liu, Zhewen Yu, Li Cai, Dian Jiao, Wenqiao Zhang, Siliang Tang, Juncheng Li, Hao Jiang, Haoyuan Li, et al. Align2llava: Cascaded human and large language model preference alignment for multi-modal instruction curation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8759–8781, 2025.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Jaewoo Lee, Boyang Li, and Sung Ju Hwang. Concept-skill transferability-based data selection for large vision-language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5060–5080, 2024.

Jingwei Li, Xinran Gu, and Jingzhao Zhang. Capacity-aware mixture law enables efficient llm data optimization. *arXiv preprint arXiv:2603.08022*, 2026a.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, et al. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026b.

Shengrui Li, Fei Zhao, Kaiyan Zhao, Jieying Ye, Haifeng Liu, Fangcheng Shi, Zheyong Xie, Yao Hu, and Shaosheng Cao. Decouple searching from training: Scaling data mixing via model merging for large language model pre-training. *arXiv preprint arXiv:2602.00747*, 2026c.

Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26679–26689. IEEE, 2024.

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

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26286–26296. IEEE, 2024a.

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

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*, 2023b.

Zikang Liu, Kun Zhou, Wayne Xin Zhao, Dawei Gao, Yaliang Li, and Ji-Rong Wen. Less is more: High-value data selection for visual instruction tuning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 3712–3721, 2025.

Run Luo, Haonan Zhang, Longze Chen, Ting-En Lin, Xiong Liu, Yuchuan Wu, Min Yang, Yongbin Li, Minzheng Wang, Pengpeng Zeng, et al. Mmevol: Empowering multimodal large language models with evol-instruct. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19655–19682, 2025.

Yiwei Ma, Guohai Xu, Xiaoshuai Sun, Jiayi Ji, Jie Lou, Debing Zhang, and Rongrong Ji. Mllm-selector: Necessity and diversity-driven high-value data selection for enhanced visual instruction tuning. *arXiv preprint arXiv:2503.20502*, 2025.

James B McQueen. Some methods of classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symposium on Math. Stat. and Prob.*, pp. 281–297, 1967.

Shruthan Radhakrishna, Aman Tiwari, Aanjaneya Shukla, Masoud Hashemi, Rishabh Maheshwary, Shiva Krishna Reddy Malay, Jash Mehta, Pulkit Pattnaik, Saloni Mittal, Khalil Slimi, et al. Apriel-1.5-15b-thinker. *arXiv preprint arXiv:2510.01141*, 2025.

Bardia Safaei, Faizan Siddiqui, Jiacong Xu, Vishal M Patel, and Shao-Yuan Lo. Filter images first, generate instructions later: Pre-instruction data selection for visual instruction tuning. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14247–14256. IEEE, 2025.

James E Smith and Robert L Winkler. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025a.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Meituan LongCat Team, Bairui Wang, Bin Xiao, Bo Zhang, Bolin Rong, Borun Chen, Chang Wan, Chao Zhang, Chen Huang, Chen Chen, et al. Longcat-flash-omni technical report. *arXiv preprint arXiv:2511.00279*, 2025b.

Jiapeng Wang, Changxin Tian, Kunlong Chen, Ziqi Liu, Jiaxin Mao, Wayne Xin Zhao, Zhiqiang Zhang, and Jun Zhou. Mergemix: Optimizing mid-training data mixtures via learnable model merging. *arXiv preprint arXiv:2601.17858*, 2026.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

Xindi Wu, Mengzhou Xia, Rulin Shao, Zhiwei Deng, Pang Wei Koh, and Olga Russakovsky. Icons: Influence consensus for vision-language data selection. *arXiv preprint arXiv:2501.00654*, 2024.

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. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818, 2023a.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy S Liang. Data selection for language models via importance resampling. *Advances in Neural Information Processing Systems*, 36: 34201–34227, 2023b.

Zhiyang Xu, Chao Feng, Rulin Shao, Trevor Ashby, Ying Shen, Di Jin, Yu Cheng, Qifan Wang, and Lifu Huang. Vision-flan: Scaling human-labeled tasks in visual instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15271–15342, 2024.

Yichen Yan, Ming Zhong, Qi Zhu, Xiaoling Gu, Jinpeng Chen, and Huan Li. Coido: Efficient data selection for visual instruction tuning via coupled importance-diversity optimization. *Advances in Neural Information Processing Systems*, 38:167045–167073, 2026.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations*, volume 2024, pp. 22743–22763, 2024.

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

Weijie Yin, Yongjie Ye, Fangxun Shu, Yue Liao, Zijian Kang, Hongyuan Dong, Haiyang Yu, Dingkang Yang, Jiacong Wang, Han Wang, et al. Sail-vl2 technical report. *arXiv preprint arXiv:2509.14033*, 2025.

Qifan Yu, Zhebei Shen, Zhongqi Yue, Yang Wu, Bosheng Qin, Wenqiao Zhang, Yunfei Li, Juncheng Li, Siliang Tang, and Yueting Zhuang. Mastering collaborative multi-modal data selection: A focus on informativeness, uniqueness, and representativeness. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 155–165, 2025.

Haotian Zhang, Mingfei Gao, Zhe Gan, Philipp Dufter, Nina Wenzel, Forrest Huang, Dhruti Shah, Xianzhi Du, Bowen Zhang, Yanghao Li, et al. Mm1. 5: Methods, analysis & insights from multimodal llm fine-tuning. In *International Conference on Learning Representations*, volume 2025, pp. 64669–64714, 2025.

A Implementation and Reproducibility Record

In this section, we detail the validation objective (Section A.1), executed settings (Table 6), benchmark definitions (Table 7), and evaluation scope (Section A.2), supplementing the procedure in Algorithm 1.

Common base and data. The 7B base θ_0 is obtained by multimodal representation pretraining followed by mid-training, as described in Section 4.1. Experts, real probes, and final selected-data models share this initialization. The upstream collection comprises approximately 40M instructions from publicly available datasets. The active selection pool contains 8,007,947 candidate IDs, partitioned into 100 semantic clusters; its provenance records are described in Section E. The implementation settings below apply to the sample-weighted gradient-search configuration in Section 3.

A.1 Validation Objective Construction

Each real probe uses uniform within-cluster selection under its assigned quotas and is trained from the common base. The fixed validation candidate panel contains 100 uniformly sampled examples per cluster. Proxies use the same 128 mixture vectors as the real probes, linking their per-example loss responses to the corresponding target scores.

For $T_x > 0$ valid answer tokens, $\ell(\theta; x) = -T_x^{-1} \sum_{t \in \text{answer}(x)} \log p_\theta(a_t \mid \text{context}(x), a_{<t})$. The matrix $X_{ij} = -\ell(\widetilde{\theta}(\mathbf{w}_i); v_j)$ contains $128 \times 10{,}000$ responses. Each example is normalized by its own answer length before aggregation, so long responses do not receive extra weight through token pooling.

For fitting-probe indices $\mathcal{I}$, candidate example j receives

$$q_{d,j} = \tfrac{1}{2}\rho_{\text{Sp}}\left(X_{\mathcal{I},j}, \mathbf{y}_{\mathcal{I}}^{(d)}\right) + \tfrac{1}{2}\rho_{\text{Sp}}\left(X_{\mathcal{I}_d^+,j}, \mathbf{y}_{\mathcal{I}_d^+}^{(d)}\right), \tag{6}$$

where $\mathcal{I}_d^+$ contains the higher-scoring half of the fitting probes for target d: 48 of 96 in each outer-fold fit, or 64 of 128 for the deployed objective. The top $m = 64$ candidates form S_d; weight fitting subsequently uses all indices in $\mathcal{I}$. This selects validation examples, not training clusters.

Selected features are standardized and fitted using Eq.(3) with $\lambda = 10$. All means and standard deviations use fitting probes only, and λ multiplies summed squared residuals. The coefficients return to raw-loss units via

$$c_{d,j} = \widehat{\beta}_{d,j}/s_j, \qquad C_d = \sum_{h \in S_d} c_{d,h}, \qquad \alpha_{d,j} = c_{d,j}/C_d. \tag{7}$$

For $C_d > 0$, the weights sum to one and weights outside S_d are zero. The conversion, including its positive-sum condition, yields the affine score–loss relation derived in Section C.4. All screening, centering, scaling, and fitting are repeated within each training fold. Final search uses weights refitted on all development probes, with no subsequent joint optimization of $\boldsymbol{\alpha}_d$ and $\mathbf{w}$.

Historical probes outside the current capacity constraints may supply objective-learning feedback, but only feasible probes can initialize search. Complete-objective evaluations select the returned iterate. Algorithm 1 gives the full sequence.

Table 5: **Deployed validation objectives and initialization.** Recipe identifiers refer to historical mixtures, not semantic clusters. All targets use the nominal step-10 bank, screen 64 examples, and run 120 projected Adam updates.

Target	Screened	Positive weights	Start recipe
General	64	15	101
Parsing	64	6	119
Knowledge	64	14	27
Overall	64	14	101

Repeat-specific settings are reported in Section B.2.

Table 6: **Executed CLIVO settings.** These settings expand Section 4.1. Expert checkpoint conventions are detailed in Section B.4.

Component	Setting
Representation and units	Qwen3-VL-Embedding-8B; standard K-means, $K = 100$
Expert bank	Common-base cluster experts; nominal step-10 configuration
Real feedback	$Y = 128$ mixture-trained probes
Validation candidates	100 examples per cluster; $H = 10{,}000$
Answer loss	Per-example mean cross-entropy over answer tokens
Screening	Equal full-range/upper-half Spearman average; retain $m = 64$
Weight fitting	Nonnegative ridge, $\lambda = 10$; train-only standardization
Probe holdout	Five repetitions of four folds; 96 fit / 32 held out
Deployed objective	Refit on all 128 probes; frozen during search
Search initialization	Lowest-objective recipe among 115 feasible probes
Search variables	FP32 mixture only; fixed base, experts, and validation weights
Optimizer	Adam; learning rate 10^{-3}; zero weight decay; 120 updates
Objective per update	All positive-weight examples; no minibatch sampling
Feasibility	Euclidean projection to $\sum_k w_k = 1$, $0 \le w_k \le \min(1, M_k/N)$
Checkpoint selection	Complete BF16 proxy objective at steps $0, 20, \ldots, 120$
Quota construction	Largest remainder; exactly $N = 100{,}000$ examples
Within-cluster selection	Uniform without replacement; ICONS only for the named variant
Final instruction tuning	Original base; 100 optimizer steps

Table 7: **Benchmark-to-target mapping.** Domain targets average their constituent benchmark scores equally. Overall averages all seven scores, not the three domain means.

Benchmark	Domain target	Domain weight	Overall weight
MMBench-EN	General	$1/3$	$1/7$
MMStar	General	$1/3$	$1/7$
GQA	General	$1/3$	$1/7$
OCRBench	Parsing	$1/2$	$1/7$
ChartQA	Parsing	$1/2$	$1/7$
MMLU	Knowledge	$1/2$	$1/7$
MMLU-Pro	Knowledge	$1/2$	$1/7$

For benchmark set $\mathcal{B}_d$, the probe target is $y_i^{(d)} = |\mathcal{B}_d|^{-1} \sum_{b \in \mathcal{B}_d} s_{i,b}$. Each $s_{i,b}$ uses the benchmark score convention of the archived evaluation. Recipe-level holdout separates probe mixtures; it does not establish disjoint benchmark questions or training examples.

Numerical conditions. The loss requires nonempty answers; standardization requires positive feature variance; converting ridge coefficients requires a positive coefficient sum. Undefined correlations and all-zero fits do not define valid objectives under the stated formulas. The derivations in Section C state the conditions under which the objective and its gradients are well defined.

Proxy construction and final sampling. Proxy parameters are formed blockwise by linear addition of frozen experts, without training or saving a full checkpoint for every candidate. Gradients pass through this construction to the mixture coefficients. Projection enforces the cluster capacities, and largest-remainder rounding preserves integer quotas under those capacities (Section C.8). Final subsets contain exactly 100,000 distinct IDs. The searched proxy is not the initialization for final instruction tuning.

A.2 Limitations and Evaluation Scope

CLIVO depends on finite feedback and merged-proxy fidelity at searched mixtures. Screening can overfit despite a small final support, and local gradient search approximates discrete subset selection and training. The current configuration establishes neither an optimal cluster count nor cross-architecture transfer. Experts and probes incur shared upfront costs. Recipe holdout measures

generalization across mixtures; it does not by itself establish generalization to unseen benchmark questions or disjoint training examples.

B Experimental Protocols

In this section, we supplement the main selection results with baseline and repeat details (Sections B.1 and B.2), specify the probe-held-out evaluation (Section B.3), and detail the allocation, expert-budget, and validation-aggregation experiments (Sections B.4 to B.6). Section B.7 documents the training and validation cases in Section 5.1.

B.1 Baseline Selection and Reporting

The comparison in Section 4.2 uses 100,000 selected examples per method, with Full as a separate 8M reference. Final training starts from the common base. Pool Random samples uniformly without replacement. Sample-level baselines are PPL, CLIP-Score, MM-Selector, PRISM, ICONS, DSIR, and DataTailor; the group-level baselines are SemDeDup and COINCIDE. ICONS uses influence-consensus scoring, DSIR uses a reference distribution, and COINCIDE uses concept–skill grouping with density and transferability. For ICONS and DSIR, reference examples are the same target-specific examples selected by CLIVO. Sharing these IDs aligns reference content; each method retains its own scoring rule.

Baseline configurations and provenance. The ICONS and DSIR rows in Table 1 are the global-selection runs using CLIVO's target-specific validation examples. CLIVO + ICONS instead preserves the learned quotas and ranks within each cluster. Random reports three runs; other baselines report single runs. Table 8 preserves the supplied Random uncertainty convention.

Table 8: **Baseline repeat statistics.** Entries reproduce mean ± population standard deviation over three sampling seeds. CLIVO's repeat statistics instead use sample variance with an $n-1$ denominator.

Method	Overall	General	Parsing	Knowledge
Random	66.61 ± 0.23	63.57 ± 0.28	77.73 ± 0.09	60.05 ± 0.44

B.2 Reported Repeats and Selection Variants

The repeated-training results use a fixed searched mixture and within-cluster random sampling for each target. The reporting convention is $\bar{y} = \frac{1}{3}\sum_{s=0}^{2} y_s$, $s_y^2 = \frac{1}{2}\sum_{s=0}^{2}(y_s - \bar{y})^2$, and $s_y = \sqrt{s_y^2}$. The means, standard deviations, and variances in Table 9 come from the supplied final result summary. Tables display scores and uncertainty to two decimal places; calculations use the supplied precision before rounding. Random and CLIVO show three-run means in the main table. These statistics describe final training at fixed searched mixtures, without repeating objective learning or search. Variances are computed before rounding, rather than by squaring the displayed standard deviations.

For method m, the additional summary is $\mathrm{Average}_m = \frac{1}{4}(s_{m,\mathrm{General}} + s_{m,\mathrm{Parsing}} + s_{m,\mathrm{Knowledge}} + s_{m,\mathrm{Overall}})$. It equally weights the four target columns, whereas Overall averages seven benchmarks. For target-aware methods, Average summarizes separately selected target models. Scores and derived differences are rounded only for display, with best and second-best entries determined before rounding.

The run record identifies General, Parsing, and Overall seeds 1/2 as training runs on 4×8 H100 GPUs with gradient accumulation 1 and 100 optimizer steps. The registered s0 allocation experiments use effective global batch 32 and data/training seeds 42/52.

CLIVO + ICONS reports one s0 real-training run for each of the four targets; Random and CLIVO report three-run means. The Knowledge result uses the same 100K selected examples, 32 H100 GPUs, gradient accumulation 1, and 100 optimizer steps. Evaluation gives MMLU 72.25 and MMLU-Pro 51.93, whose mean is 62.09. Global selection variants also report single-run real-training results. Differences from three-run means describe the reported scores, not paired seed effects or statistical

significance. Quota-constrained ICONS preserves CLIVO's integer allocation and changes the within-cluster selector; global ICONS and global DSIR do not preserve that allocation. The global selectors use the target-specific validation examples shared with CLIVO. The ICONS and DSIR entries in Table 1 report these global selection runs; columns are target summaries, not necessarily one model's profile.

Table 9: **CLIVO final-training repeats.** Three runs at fixed searched shares with uniform within-cluster sampling. Variance uses the $n - 1$ denominator. These are not repeats of objective learning and search.

Target	Mean	Standard deviation	Sample variance
General	67.28	0.53	0.28
Parsing	79.64	0.59	0.34
Knowledge	61.76	0.11	0.01
Overall	68.13	0.13	0.02

B.3 Probe Holdout and Candidate-Selection Metrics

This protocol specifies the probe holdout used by the matched 64-example comparison in Table 4. Its four aggregation rules are defined in Section B.6. The expert-budget experiment uses the same outer split sizes but a separate cluster-selection rule, described in Section B.4.

Each of five repetitions partitions the 128 probe identifiers into four folds of 32. The split identifiers are fixed for the paired comparisons. For each fold, let $\mathcal{I}$ be its 96 fitting indices and $\mathcal{T}$ its 32 held-out indices. All sample ranking, centering, standardization, and fitting use $\mathcal{I}$ only. The candidate samples are fixed independently of this splitting. Screening is repeated within each fitting fold; the all-probe top-64 panel used for final search is not reused in outer evaluation.

For evaluator e, let $z_{i,e} = -J_e(\mathbf{w}_i)$. Compute Spearman correlation between $z_{i,e}$ and $y_i^{(d)}$ on $\mathcal{T}$, using average ranks for ties. For selection, the candidate set is $\mathcal{C} = \{i \in \mathcal{T} : \mathbf{w}_i \in \mathcal{W}\}$, and $\widehat{i}_e$ maximizes $z_{i,e}$ within this feasible subset. Define

$$R_e = \max_{i \in \mathcal{C}} y_i^{(d)} - y_{\widehat{i}_e}^{(d)}. \tag{8}$$

This regret compares the selected recipe with the best observed feasible recipe in the same held-out fold, not with an unknown global optimum. Table 14 reports its repeated-fold mean.

Interpretation of repeated holdout. Metrics are averaged over 20 outer splits. These splits reuse the same 128 probe models and are not independent training replications. Method choices were developed using these labels, so the comparison is exploratory rather than a fresh external test. All four readers in Table 4 share each split and its 64 fitting-selected examples.

B.4 Expert Training Amount and Proxy Agreement

The completed comparison uses four banks: nominal step-1, step-5, and step-10 short-budget experts and a full bank trained under a separate variable-step regime. Every bank is evaluated on the same 128 proxy mixtures and seed-1 panel of 100 examples from each of 100 clusters, giving 1,280,000 per-example losses per bank. Real-model losses use these same mixtures and example IDs. The comparison uses matched example IDs for proxy and real losses.

Mixture-response fidelity. For each mixture, average losses over the same 10,000 examples for the proxy and real model. The fidelity column in Table 10 is Spearman between their negative mean losses over all 128 mixtures. Negating both losses leaves this correlation unchanged. It tests preservation of loss ordering, not prediction of benchmark accuracy. Figure 2 plots all paired observations without fitting a response surface. Although the full bank has the lowest absolute loss, nominal step 10 has the highest fidelity to real-model loss responses.

Table 10: **Expert budget and two kinds of agreement.** Fidelity compares proxy with real-model loss over 128 mixtures. G/P/K/O are held-out correlations with real benchmark scores after budget-specific validation-cluster selection. Bold: highest in each correlation column.

Bank	Mean loss	Fidelity ρ	G ρ	P ρ	K ρ	O ρ
Step 1	1.283	0.485	0.383	0.435	0.523	0.367
Step 5	1.007	0.647	0.409	**0.462**	0.524	0.387
Step 10	0.972	**0.730**	0.518	0.448	**0.527**	**0.460**
Full schedule	0.940	0.644	**0.577**	0.382	0.440	0.457

Downstream validation selection. Each of five repeated four-fold outer splits uses 96 fitting and 32 held-out mixtures. For each target and budget, rank validation clusters by the fitting-probe Spearman correlation between negative cluster-mean loss and archived real-training scores. Inner three-fold validation selects the number of clusters from $\{1, 2, 4, 8, 16, 32\}$. Freeze the selected cluster set before scoring the outer held-out mixtures.

Let $\bar{\ell}_{i,k}^{(b)}$ be proxy i's mean loss on cluster k at budget b, and let $A_{d,b,\mathcal{I}}$ be the cluster set selected on outer fitting indices $\mathcal{I}$. The held-out statistic is

$$\rho_{d,\mathcal{I}}^{(b)} = \rho_{\mathrm{Sp}}\left(\left\{-\frac{1}{|A_{d,b,\mathcal{I}}|}\sum_{k\in A_{d,b,\mathcal{I}}}\bar{\ell}_{i,k}^{(b)}\right\}_{i\in\mathcal{T}}, \{y_i^{(d)}\}_{i\in\mathcal{T}}\right). \tag{9}$$

Table 10 averages this correlation over the 20 outer splits. Because clusters are reselected separately for each bank, this evaluates the bank and cluster-selection procedure jointly. It differs both from the all-panel proxy–real loss correlation and from the matched 64-example aggregation ablation. The nominal step-10 bank is the shared short-budget configuration used in CLIVO. The full bank has the highest General correlation and step 5 has the highest Parsing correlation; step 10 has the highest fidelity, Knowledge correlation, and Overall correlation.

Meaning of the budget labels. The main text and figure use nominal checkpoint indices $b = 1, 5, 10$. These configurations have archived aliases 1K, 5K, and 10K; they identify budget configurations, not counts of unique training examples. The full schedule is a separate variable-step regime, not a later point on the same training trajectory.

B.5 Tied Allocation with a Fixed Expert Bank

The completed comparison uses effective group counts $G \in \{20, 50, 100\}$ for all four targets, reported as adjustable shares K in Table 3. We use G here to distinguish the parent dimension from the fixed 100-expert bank. A target-independent hierarchy ties the same 100 nominal step-10 experts into 20 or 50 parents; no new expert bank is trained. The base, target-specific 64-example weighted objective, feasible initial 100-dimensional mixture, 120-step search rule, and capacity constraints are shared.

For initial mixture $\mathbf{w}^0$, let $\mathcal{G}_g$ contain parent g's children and $z_g^0 = \sum_{k\in\mathcal{G}_g} w_k^0$. For $z_g^0 > 0$, define

$$a_{k|g} = w_k^0/z_g^0, \qquad B_{kg} = a_{k|g}\ \ (k \in \mathcal{G}_g), \qquad \mathbf{w} = B\mathbf{z}. \tag{10}$$

Entries outside each parent's children are zero. For a zero-mass parent, the parameterization requires a fixed normalized vector over its children. With this convention, $B\mathbf{z}^0 = \mathbf{w}^0$ and each column of B sums to one. Zero child shares within a positive-mass parent remain tied to zero.

Parent capacities follow from the original child capacities:

$$c_g = \min\left(1, \min_{\substack{k\in\mathcal{G}_g \\ a_{k|g}>0}} \frac{u_k}{a_{k|g}}\right). \tag{11}$$

Search optimizes only parent shares, subject to $\sum_g z_g = 1$ and $0 \le z_g \le c_g$, and expands them to the 100-dimensional mixture. The parent gradient is $B^\top \nabla_{\mathbf{w}} J_d$. At $G = 100$, shares are untied. This

comparison changes the search coordinates as well as the allocation constraints; it does not maintain a common 100-dimensional Adam state.

Each returned mixture supplies 100,000 distinct training examples by random within-cluster selection. All real runs use data/training seeds 42/52, effective global batch 32, and 100 optimizer steps, followed by target evaluation. Table 3 reports one-seed scores; the main table reports three-run means. This comparison measures allocation quality rather than search time-to-threshold.

Proxy loss and real target scores need not vary monotonically together: Knowledge has its lowest selected proxy loss at 20 groups but its highest real score at 100. Several real-score differences are small relative to observed training variation, so the single-seed allocation comparison supports target-dependent effects rather than a universal optimal granularity.

What the tied comparison isolates. Substitution gives the exact identity $\widetilde{\theta}(B\mathbf{z}) = \theta_0 + \sum_g z_g \sum_k B_{kg}\Delta_k$: a parent expert difference is a fixed weighted average of existing child differences. This establishes parameter-level equivalence to merging preaggregated differences, before floating-point effects. The executed implementation instead expands parent weights and merges all 100 experts; it also computes all 100 expert-direction gradients before aggregating them by parent. No preaggregated 20- or 50-expert bank was materialized. This control isolates adjustable allocation shares on one proxy basis, while also changing the coordinates used by Adam. It does not test independently trained expert banks at different clustering resolutions. The fixed validation objective has one probe-held-out evaluation; changing G does not produce a new correlation estimate.

Full-framework summary and historical probe references. The CV-Spearman column in Table 2 reports the sample-weighted rule's global held-out correlations from Table 4. Objective learning and search are performed once per target; the final score $\bar{F}$ averages three training runs at the fixed selected mixture, not three complete framework runs. Standard deviations remain in Table 9. The best-of-128 historical probe scores B are 66.600, 79.310, 61.465, and 67.9185714286 for General, Parsing, Knowledge, and Overall, respectively (recipe IDs 101, 29, 32, and 101). We report $\Delta F = \bar{F} - B$ in percentage points, computing differences before rounding. Historical probes use a different training budget from the final 100K/100-step models, so these are descriptive cross-protocol differences, not controlled optimization gains or statistical-significance tests. The positive differences across targets describe the framework's final results; isolating the downstream effect of gradient updates requires matched-budget training of starting mixtures.

Within-run search progress. Table 2 uses $D_J = 100(J_0 - J^\star)/J_0$, where J_0 is the complete-objective evaluation at step zero of the same run. This avoids cached-versus-remerged evaluation differences. For $K = 100$, the inputs are in Table 16. The granularity comparison in Table 3 reports the lowest proxy loss, its first recorded step, cumulative search cost to that step, and the corresponding real-training score for each K. Parsing at $K = 20$ retains its starting mixture at step zero. The first best recorded step $t^\star$ is selected from $0, 20, \ldots, 120$; all searches execute 120 updates. It is a checkpoint-selection index, not a convergence or runtime measurement. Table 3 reports matched s0 real-training scores, with bold marking the highest score across group counts. The fixed validation rule has no new CV-Spearman estimate when only K changes.

B.6 Matched Validation Aggregation

The four-way comparison uses the nominal step-10 $128 \times 10{,}000$ per-example loss matrix. In each of five repeated four-fold outer splits, only the 96 fitting probes screen 64 examples using Eq.(6), with 48 probes in its higher-scoring half. All four readers receive these identical example IDs. The 32 held-out probes participate in neither screening nor fitting. This ablation was recomputed from existing losses without additional model forward passes or training.

Separating response granularity from weighting. Sample readers retain each of the 64 negative losses as a feature. Cluster readers first average these features within each represented cluster, giving about 23–30 features per fold. Equal readers average their features uniformly; consequently, equal cluster weights and equal sample weights need not induce the same mass per example. Weighted readers apply nonnegative ridge with $\lambda = 10$, fitting-only feature standardization, conversion back to raw-loss units, and simplex normalization. All readers share sample-level screening; this is an

aggregation ablation, not a comparison to the original top-k-cluster selection method. Table 4 reports mean held-out global Spearman, and Table 13 reports correlation restricted to the 16 highest-predicted held-out mixtures.

Feasible top-1 decisions. Only 115 of the 128 probes meet the capacities for selecting 100K distinct examples. With fitting folds unchanged, restrict each test fold to its feasible subset $\mathcal{T}_{\text{feas}}$ and compute

$$R = \max_{i \in \mathcal{T}_{\text{feas}}} y_i^{(d)} - y^{(d)}_{\arg\max_{i \in \mathcal{T}_{\text{feas}}} z_i}, \tag{12}$$

where z_i is the reader's predicted score. Sample weighting has the highest feasible global correlation as well, with 0.704/0.475/0.615/0.589 for General/Parsing/Knowledge/Overall. However, cluster weighting has lower feasible top-1 regret on Parsing, Knowledge, and Overall (Table 14). Better ordering across candidates does not establish a better single selected mixture.

Concentration and evaluation scope. The sample-weighted fits place positive mass on a mean of 15.3/12.2/13.6/16.1 examples; the mean largest weights are 0.347/0.374/0.250/0.416 (Table 15). This describes how the fit uses individual responses and motivates weight-robustness checks; it does not prove selection quality. The 20 overlapping folds reuse 128 development models, and method choices were developed using these labels. The results are exploratory held-out comparisons, not a fresh independent test set or evidence of a downstream search gain.

B.7 Training-Data Composition and Validation Cases

Capability annotation hierarchy. We use GPT-based task annotations and cluster summaries to describe the capabilities represented in each semantic region. The summaries retain ranked task types, a short definition, and a detailed content description. For the coarse visualization labels, we take the first entry in each cluster's `top_task_types` field as its primary task and apply the mapping in Table 11. Fine-grained task descriptions and the displayed examples support comparisons within each coarse class. These labels describe the data for analysis; clustering uses embeddings, and training shares are learned from the target objective. The complete 100-cluster map, including primary tasks and descriptions, is provided in `cluster_capability_map.csv` under `data/selected_cases/`.

Table 11: **Coarse capability labels for visualization.** Each cluster's primary annotated task determines its coarse label. The original task descriptions are retained for finer interpretation.

Coarse capability	Primary task types
Visual QA & Captioning	Image Captioning
Grounding & Detection	Object Detection; Visual Grounding; Video Tracking
OCR & Documents	OCR; Document Understanding
Charts & Data	Chart/Data Analysis
Math & Logic	Math Solving; Logic/Puzzle
Knowledge & Science	Scientific Analysis; Concept Explanation; Fact Lookup
Generation & Other	Remaining task types, including code and creative generation

Within-capability training cases. The case panel in Figure 3(b) uses the cluster-to-capability map to pair c60/c38 (Grounding & Detection), c27/c14 (Charts & Data), and c36/c20 (Knowledge & Science) for General, Parsing, and Knowledge. The bundled case records also contain the Overall pair c26/c25 (Math & Logic). The displayed percentages are final continuous training shares over all 100 clusters; they are neither within-category normalized shares nor validation weights. Pool shares use the 8,007,947 candidate IDs. Higher-share examples appear in the corresponding s1 training manifest; comparison examples come from the candidate pool. The cases illustrate content differences within an existing category and do not establish that all examples in a cluster share one format or that a particular format causes the performance gain. The cluster shares and capability labels define the quantitative summaries below; example sources are listed in Section E.

Allocation concentration. For normalized training shares w_k, the entropy-based effective number of clusters is $K_{\text{eff}} = \exp(-\sum_{k:w_k>0} w_k \log w_k)$. The final General, Parsing, Knowledge, and

Overall mixtures have $K_{\text{eff}} = 3.024, 10.134, 3.923, 4.362$, respectively. These values are computed from the 100-cluster vectors in the case bundle and describe allocation concentration, not the number of optimized coefficients.

Uniform-cluster reference. The uniform reference assigns $1/K = 1\%$ of the training budget to each cluster. For a coarse capability containing a cluster set A, its reference share is $|A|/K$, its learned share is $\sum_{k \in A} w_k$, and its enrichment is $(\sum_{k \in A} w_k)/(|A|/K)$. For example, a category with ten clusters has a 10% reference share; a learned share of 30% corresponds to $3\times$ uniform. The percentages beside individual clusters use the full training budget as their denominator.

Distances between target mixtures. The distances in Table 17 use the continuous final vectors stored in the case bundle's `cluster_weights.csv`. For distributions p and q over the 100 clusters, total variation is $\frac{1}{2}\sum_k |p_k - q_k|$. Jensen–Shannon divergence uses $\frac{1}{2}\mathrm{KL}_2(p\|m) + \frac{1}{2}\mathrm{KL}_2(q\|m)$ with $m = (p+q)/2$ and base-2 logarithms. These statistics compare final training shares, not validation weights or coarse-category counts. They describe composition; the controlled granularity experiment evaluates the effect of restricting allocation freedom.

Complete fixed-fold validation check. Table 12 reports the five highest-weight items per target and the deterministic hash-selected control in the first persisted 96/32 split (seed 2026092200). All weights are fitted on the 96 development probes; each individual correlation uses the 32 held-out probes. The highest Parsing item asks for a map-based comparison and has correlation -0.009; the highest Overall item asks for a map range and has correlation -0.085. Consequently, a large fitted coefficient is not a guarantee of a positive marginal held-out response. The combined first-fold correlations are 0.8585, 0.1602, 0.4795, and 0.5722 for General, Parsing, Knowledge, and Overall; the repeated-fold results in Table 4 assess the complete selection-and-weighting rule. None of these fixed-fold weights is substituted for the all-development objective used in final search.

Table 12: **Fixed-fold validation cases.** Five leading positive-weight examples and one hash-selected control per target. ρ is the individual negative-loss Spearman on 32 held-out probes. Rows are ordered by fitting weight, retaining weak and negative held-out responses.

Target	Case	Cluster	α	ρ
General	Rank 1	60	0.402	0.652
	Rank 2	97	0.178	0.688
	Rank 3	69	0.103	0.704
	Rank 4	60	0.098	0.828
	Rank 5	23	0.083	0.591
	Control	12	0.000	0.301
Parsing	Rank 1	69	0.329	-0.009
	Rank 2	44	0.190	0.346
	Rank 3	25	0.148	0.152
	Rank 4	99	0.125	-0.173
	Rank 5	79	0.038	0.226
	Control	12	0.000	0.303
Knowledge	Rank 1	66	0.207	0.450
	Rank 2	81	0.136	0.579
	Rank 3	61	0.122	0.469
	Rank 4	36	0.115	0.263
	Rank 5	20	0.086	0.377
	Control	12	0.000	-0.086
Overall	Rank 1	69	0.484	-0.085
	Rank 2	38	0.123	0.328
	Rank 3	69	0.097	0.038
	Rank 4	38	0.084	0.578
	Rank 5	69	0.083	0.123
	Control	12	0.000	0.282

C Derivations and Scope of the Analysis

In this section, we supplement Section 3 with the mixture interpretation of uniform sampling (Section C.1), allocation and aggregation constraints (Sections C.2 and C.3), validation-weight conversion and conditional selection bounds (Sections C.4 and C.5), and the gradient, search-work, projection, and quota calculations (Sections C.6 to C.8). Section C.9 explains the quota-preserving ranking variant.

Write $F_d(\mathbf{w}) = F_d(\mathbf{w}; \pi_{\mathrm{U}})$ for the expected score in Eq.(1) with uniform within-cluster selection, a fixed base, and a fixed training recipe. The training recipe is held fixed; the expectation is over uniform within-cluster subset selection. The feasible-set and sampling arguments do not require differentiability of the real-training response. Gradients below apply to the merged proxy, not to real training.

C.1 Why Random-Within-Cluster Probes Represent a Mixture

Uniform within-cluster selection fixes the conditional sampling rule, so changing the quotas changes the mixture without adding a ranking preference inside any cluster. The following finite-population argument states precisely what this represents.

Proposition C.1 (Mixture representation by stratified sampling)**.** *Let the clusters be disjoint, with $M_k = |\mathcal{D}_k| > 0$, and fix integer quotas $0 \le n_k \le M_k$ with $\sum_k n_k = N$. Independently sample n_k examples uniformly without replacement from each cluster, giving $\mathcal{S} = \bigcup_k \mathcal{S}_k$, and set $r_k = n_k/N$. For any fixed statistic $f : \mathcal{D} \to \mathbb{R}$, define $\bar{f}_k = M_k^{-1} \sum_{x \in \mathcal{D}_k} f(x)$ and $\widehat{\mu}_f = N^{-1} \sum_{x \in \mathcal{S}} f(x)$. Then*

$$\mathbb{E}[\widehat{\mu}_f] = \sum_k r_k \bar{f}_k, \qquad \Pr(Z = x) = \frac{r_k}{M_k} \quad (x \in \mathcal{D}_k), \tag{13}$$

where Z is a uniformly chosen member of the random subset. For $M_k > 1$, let $s_{f,k}^2 = (M_k - 1)^{-1} \sum_{x \in \mathcal{D}_k} (f(x) - \bar{f}_k)^2$. The sampling variance is

$$\mathrm{Var}(\widehat{\mu}_f) = \sum_{k : n_k > 0} r_k^2 \left(1 - \frac{n_k}{M_k}\right) \frac{s_{f,k}^2}{n_k}, \tag{14}$$

with singleton-cluster variance set to zero.

Proof. For $x \in \mathcal{D}_k$, its inclusion indicator I_x satisfies $\mathbb{E}[I_x] = n_k/M_k$. Linearity of expectation gives $\mathbb{E}[\widehat{\mu}_f] = N^{-1} \sum_k (n_k/M_k) \sum_{x \in \mathcal{D}_k} f(x)$. Conditional on inclusion, x is chosen as Z with probability $1/N$, which gives its stated marginal probability. For two distinct examples in cluster k, $\mathbb{E}[I_x I_{x'}] = n_k(n_k - 1)/(M_k(M_k - 1))$. Expanding the variance with these pairwise inclusion probabilities gives the finite-population correction in Eq.(14); independence across clusters allows their contributions to be added. □

Thus the expected empirical distribution is the mixture of uniform cluster distributions with shares $\mathbf{r}$, not necessarily the original pool distribution. If Nw_k are integers, $\mathbf{r} = \mathbf{w}$. Otherwise largest-remainder rounding gives $|r_k - w_k| < 1/N$ and $\|\mathbf{r} - \mathbf{w}\|_1 < K/N$. For any statistic in $[0, B]$, its expected subset mean differs from $\sum_k w_k \bar{f}_k$ by at most $\frac{B}{2} \|\mathbf{r} - \mathbf{w}\|_1$. The fixed-statistic result applies to example loss, or componentwise to its gradient, at parameters fixed independently of the sampled subset. It does not assume that cross-entropy is bounded; the bounded-statistic inequality applies only when the stated bound holds.

From sampled data to real-training feedback. With the base, training recipe, selection rule, and target evaluation fixed, a genuinely trained probe score Y_d is a draw from the response distribution with mean $F_d(\mathbf{w}; \pi_{\mathrm{U}})$ as defined in Eq.(1). For R independent repetitions, $\widehat{F}_{d,R} = R^{-1} \sum_{r=1}^{R} Y_d^{(r)}$ satisfies $\mathbb{E}[\widehat{F}_{d,R}] = F_d(\mathbf{w}; \pi_{\mathrm{U}})$ and $\mathrm{Var}(\widehat{F}_{d,R}) = \mathrm{Var}(Y_d)/R$. This interpretation requires the same declared training protocol and no selection of repetitions based on their outcomes. It does not make a single probe noise-free, or equate training on a sampled subset with training on the full weighted pool: the training procedure is nonlinear. Nor does it identify the response under another within-cluster rule π. The advantage of π_{U} is that it introduces no additional sample-ranking policy when measuring mixture effects.

C.2 Allocation Constraints and Nested Families

The capacity-constrained allocation set used throughout the paper is

$$\mathcal{W} = \left\{\mathbf{w} \in \mathbb{R}^K : \mathbf{1}^\top \mathbf{w} = 1,\ 0 \le w_k \le u_k\right\}, \qquad u_k = \min\{1, M_k/N\}. \tag{15}$$

It is nonempty when the eligible cluster capacities can supply the budget.

Proposition C.2 (Feasible-set inclusion). *Let the columns of $B \in \mathbb{R}^{K\times G}$ be nonnegative and sum to one. Each column is supported within one of G disjoint groups that partition the fine clusters; some child coefficients may be zero. Fix the within-group proportions encoded by B, and let $\mathcal{Z} = \{\mathbf{z} \ge 0 : \mathbf{1}^\top \mathbf{z} = 1,\ B\mathbf{z} \le \mathbf{u}\}$ be nonempty. Then $B\mathcal{Z} \subseteq \mathcal{W}$ and $\sup_{\mathbf{z}\in\mathcal{Z}} F_d(B\mathbf{z}) \le \sup_{\mathbf{w}\in\mathcal{W}} F_d(\mathbf{w})$.*

Proof. For $\mathbf{z} \in \mathcal{Z}$, nonnegativity of B gives $B\mathbf{z} \ge 0$. Since its columns sum to one, $\mathbf{1}^\top B\mathbf{z} = \mathbf{1}^\top \mathbf{z} = 1$. The definition of $\mathcal{Z}$ enforces $B\mathbf{z} \le \mathbf{u}$. Thus every tied mixture is a feasible fine mixture, and taking a supremum over the larger set cannot reduce its value. □

The inequality can be strict when the fixed within-group proportions exclude a better composition, but strictness is not guaranteed. With nondegenerate capacities and nonempty relative interiors, the untied simplex has dimension $K - 1$ and the tied family at most $G - 1$. Active caps and zero proportions can reduce these dimensions. The proposition supplies neither a finite-feedback sample complexity nor an optimal K. Independent source and semantic partitions are generally not nested, so it cannot be used to rank them without an explicit common refinement.

For the control in Section B.5, $B\mathbf{z}^0 = \mathbf{w}^0$ establishes nonemptiness directly. The equivalent group caps follow from $a_{k|g} z_g \le u_k$ for every child with $a_{k|g} > 0$, which yields Eq.(11).

C.3 Information Removed by Fixed Aggregation

Proposition C.3 (Aggregation ambiguity). *Let $A : \mathbb{R}^m \to \mathbb{R}^q$ be a fixed linear aggregation and suppose two candidate loss-feature vectors satisfy $A\mathbf{x} = A\mathbf{x}'$. For any predictor h of the aggregated features and associated real scores y, y',*

$$\max\{|h(A\mathbf{x}) - y|,\ |h(A\mathbf{x}') - y'|\} \ge \tfrac{1}{2}|y - y'|.$$

Proof. Both predictions equal a common number q. The triangle inequality gives $|y - y'| \le |y - q| + |q - y'|$. At least one summand is at least half the left-hand side. □

Differences in $\ker A$ are invisible to every predictor operating only on $A\mathbf{x}$, even if that predictor is nonlinear. A sample-weighted linear score $\boldsymbol{\alpha}^\top \mathbf{x}$ can distinguish some such pairs when its weights are not restricted to the row space of A. This establishes representational possibility, not better generalization from finite probes.

In the matched ablation of Table 4, $m = 64$ and each row of A averages the selected examples from one represented cluster. A nonnegative unit-sum combination of cluster means has sample weights $\boldsymbol{\alpha} = A^\top \boldsymbol{\gamma}$: examples in one cluster receive identical weights. Sample-level weighting releases these within-cluster equality constraints on the same screened examples. This larger weighting family can retain distinctions removed by cluster averaging; its empirical ranking quality is measured by probe holdout, not guaranteed by the inclusion argument or ridge regularization.

C.4 Validation Weights on the Original Loss Scale

On fitting probes, write $X_{ij} = -\ell_{ij}$ and $\widetilde{X}_{ij} = (X_{ij} - \mu_j)/s_j$ for retained features with $s_j > 0$. The fitted affine prediction is

$$\begin{aligned}\widehat{y}_i &= \bar{y}_d + \sum_j \widehat{\beta}_j (X_{ij} - \mu_j)/s_j \\ &= \underbrace{\bar{y}_d - \sum_j c_j \mu_j}_{b_d} + \sum_j c_j X_{ij} = b_d - C_d J_d(\mathbf{w}_i), \end{aligned} \tag{16}$$

$$c_j = \widehat{\beta}_j / s_j, \qquad C_d = \sum_j c_j > 0, \qquad \alpha_j = c_j / C_d.$$

Thus maximizing the fitted score and minimizing the raw weighted loss give identical rankings. The positive rescaling does not change exact minimizers, although changing the numerical scale could affect a finite optimization algorithm. The deployed implementation uses the normalized α in Eq.(7). This derivation relies on nonnegative coefficients and a positive sum; it does not define an evaluator when every coefficient is zero.

For a fixed screened support, the ridge objective has Hessian $2\widetilde{X}^\top \widetilde{X} + 2\lambda I \succ 0$ for $\lambda > 0$. It is coercive and strictly convex, so its minimization over the closed nonnegative orthant has a unique solution, even when there are fewer fitting probes than retained features. This addresses numerical identifiability of the regularized fit, not prediction accuracy: outcome-dependent screening and generalization still require the complete outer holdout procedure.

C.5 Selection Error and an Additional Response Fit

Merged proxies and learned validation weights approximate the real-training response, so proxy mismatch and finite-feedback error can remain even without a response regressor. The following conditional bound separates these errors from the additional error introduced by fitting another mixture-response function.

Proposition C.4 (Conditional decision bound)**.** *Let $\mathcal{C}$ be a fixed nonempty finite set of feasible candidate mixtures, and suppose for some $a_d \in \mathbb{R}$, $c_d > 0$,*

$$\sup_{\mathbf{w}\in\mathcal{C}} |F_d(\mathbf{w}) - (a_d - c_d J_d(\mathbf{w}))| \le \epsilon_d.$$

If $\widehat{\mathbf{w}} \in \mathcal{C}$ satisfies $J_d(\widehat{\mathbf{w}}) \le \min_{\mathbf{w}\in\mathcal{C}} J_d(\mathbf{w}) + \eta$, then

$$\max_{\mathbf{w}\in\mathcal{C}} F_d(\mathbf{w}) - F_d(\widehat{\mathbf{w}}) \le 2\epsilon_d + c_d\eta. \tag{17}$$

If instead a response predictor h satisfies $\sup_{\mathcal{C}} |h - J_d| \le \xi$ and the returned candidate is η-optimal for h on $\mathcal{C}$, the bound becomes $2\epsilon_d + c_d(\eta + 2\xi)$.

Proof. Let $\mathbf{w}^\star$ maximize F_d over $\mathcal{C}$. The assumed uniform score-approximation bound gives

$$F_d(\mathbf{w}^\star) - F_d(\widehat{\mathbf{w}}) \le c_d\big[J_d(\widehat{\mathbf{w}}) - J_d(\mathbf{w}^\star)\big] + 2\epsilon_d.$$

Since $J_d(\mathbf{w}^\star) \ge \min_{\mathcal{C}} J_d$, approximate minimization bounds the bracket by η. For the response predictor, if $\mathbf{w}_J$ minimizes J_d,

$$J_d(\widehat{\mathbf{w}}) \le h(\widehat{\mathbf{w}}) + \xi \le h(\mathbf{w}_J) + \eta + \xi \le J_d(\mathbf{w}_J) + \eta + 2\xi.$$

Apply the first bound. □

The same algebra extends to attained minima on compact continuous sets. It does not assert that a finite optimizer attains a small η. The score-approximation error ϵ_d includes merged-proxy mismatch, validation-objective mismatch, and any failure to transfer from historical to newly searched mixtures. The proof does not claim these errors are independently estimable or additive. For an adaptively searched family, the uniform assumption must hold on the searched candidates, not merely on the original 128 probes. Higher empirical Spearman correlation does not establish this assumption. Actual proxy verification can correct the ranking of proposed candidates, although it does not establish the assumed relationship between proxy loss and real-training score.

C.6 Gradients Along Feasible Mixture Directions

Vectorize all parameters and place the expert differences in $D = [\Delta_1, \ldots, \Delta_K]$. The affine merge is $\widetilde{\theta} = \theta_0 + D\mathbf{w}$. For the frozen sample objective $\mathcal{L}_d(\theta)$, the chain rule gives

$$\nabla_{\mathbf{w}} J_d = D^\top \nabla_\theta \mathcal{L}_d(\widetilde{\theta}),$$

which is Eq.(5). For any feasible displacement $\mathbf{s}$, $\mathbf{1}^\top \mathbf{s} = 0$, hence $D\mathbf{s} = \sum_k s_k \theta_k$. Using expert parameters rather than differences changes coordinate gradients by a shared constant, but not their directional derivative on the simplex. The implementation should nonetheless retain its specified gradient convention; Adam's coordinatewise state is not invariant to arbitrary gradient shifts.

C.7 Shared Feedback and Search-Work Decomposition

The following accounting explains the reuse mechanism in Section 5.2. Fix the base, expert bank, candidate pool, and probe mixtures. For a new target d, acquiring the probe scores costs $C_{\mathrm{score}}(d)$; it uses existing benchmark outputs when available and benchmark evaluation otherwise. Screening and fitting the validation objective use the existing $Y \times H$ loss matrix. The incremental selection cost decomposes as

$$\Delta C_d = C_{\mathrm{score}}(d) + C_{\mathrm{screen}}(Y, H) + C_{\mathrm{fit}}(Y, m) + C_{\mathrm{search}}(d). \tag{18}$$

The middle terms are CPU computations. Encoding, clustering, expert and probe training, and construction of the fixed loss matrix are shared setup costs. Final selected-data training is a separate stage. Changing the base or semantic partition changes these reusable assets.

Allocation dimension versus bank size. Let E be the number of stored experts, P the number of merged parameters, and G the number of adjustable parent shares. The columns of $D \in \mathbb{R}^{P \times E}$ are expert differences and $B_G \in \mathbb{R}^{E \times G}$ expands parent shares. At most one entry per row of B_G is nonzero. For the accumulated parameter gradient $q = \nabla_\theta \mathcal{L}_d$, the executed order is

$$\mathbf{w} = B_G \mathbf{z}, \qquad \widetilde{\theta} = \theta_0 + D\mathbf{w}, \qquad \nabla_{\mathbf{z}} J_d = B_G^\top (D^\top q). \tag{19}$$

Expansion and parent aggregation take $O(E)$ arithmetic. Dense parameter merging and expert-gradient contraction each take $O(PE)$, while the validation forward/backward work $C_{\mathrm{FB}}(S_d)$ depends on the fixed active examples and their sequence lengths. Adam costs $O(G)$; bisection projection with L_τ iterations costs $O(GL_\tau)$. Thus the per-step arithmetic has the form

$$C_{\mathrm{step}}(G; E) = C_{\mathrm{FB}}(S_d) + O(PE) + O(E + GL_\tau). \tag{20}$$

At fixed $E = 100$, increasing G retains the bank-wide work and the number of model passes. Every validation-example backward pass contributes to all directions through Eq.(19); it is not repeated once per direction. Real runtime also includes transfers, collectives, and implementation-dependent handling of zero weights. These terms are measured by the active-loop accounting in Section D.3.

Algebraically, one could materialize DB_G and then operate on G preaggregated directions. The reported tied runs do not use this implementation. Enlarging the expert bank itself increases the PE terms and its construction and storage requirements. The fixed-bank argument therefore concerns allocation freedom, not unrestricted bank scaling or the number of iterations needed to reach a specified loss.

Response-regression accounting. If M new mixtures need loss labels, a regression route incurs candidate merging and panel forwards, followed by fitting, proposal enumeration, and any verification passes. Cached per-example losses covering a new objective's support can be reweighted without further model evaluation; cached scalar objectives alone need not provide this coverage. This reuse applies equally to gradient and regression objectives. No universal relation between M and G is assumed. For illustration, a complete quadratic response on the $G - 1$ simplex coordinates has $G(G+1)/2$ coefficients, whereas a particular tree regressor need not fit that family. The quadratic count is not a sample-complexity or runtime bound for the recorded regression implementation.

C.8 Projection and Integer Quotas

The capped simplex is nonempty precisely when $\sum_k u_k \geq 1$. For its Euclidean projection, the KKT conditions of $\min_{\mathbf{w}\in\mathcal{W}} \frac{1}{2}\|\mathbf{w}-\mathbf{v}\|^2$ give

$$w_k = \mathrm{clip}(v_k - \tau, 0, u_k), \qquad \sum_k w_k = 1. \tag{21}$$

The sum of these coordinates is continuous and nonincreasing in τ. A bracketing interval is $[\min_k(v_k - u_k), \max_k v_k]$; bisection finds a threshold within the declared tolerance. The projected vector is unique, though the threshold need not be unique when every coordinate is saturated.

For integer capacities M_k and exact feasible $\mathbf{w}$, write $q_k = Nw_k$ and $n_k^0 = \lfloor q_k \rfloor$. Let $R = N - \sum_k n_k^0$ and add one to the R largest fractional parts, breaking ties by cluster ID. The quotas sum to N and each differs from q_k by less than one. Since $q_k \leq M_k$ and M_k is integer, every positive-fraction coordinate satisfies $\lceil q_k \rceil \leq M_k$; rounding therefore preserves capacities. In floating-point code, repair feasibility within tolerance before rounding and verify the integer invariants explicitly. This arithmetic guarantee does not imply that rounding leaves model performance unchanged.

C.9 Allocation and Within-Cluster Selection

At the mixture level, the selection rule in Section 3.1 induces $p_{\mathbf{w},\pi}(x) = \sum_k w_k p_\pi(x \mid k)$ for a disjoint partition. For a finite selected subset, the group masses are the realized shares n_k/N; the within-group distribution can also depend on the quota when π ranks examples. Changing quotas changes group mass, while changing π changes which examples fill each quota. These are compatible decisions but their training effects can interact, so optimizing one does not guarantee optimality of the other. The default rule π_{U} samples uniformly within each cluster, as analyzed in Section C.1. No additive-gain claim for combining CLIVO with a ranking method follows from this factorization alone. The quota-preserving ICONS experiment in Table 1 measures such a combination. Relative to CLIVO's three-run means with random within-cluster selection, the single-run ICONS results are higher for General, Parsing, and Knowledge and slightly lower for Overall. This comparison describes target-dependent outcomes rather than paired effects.

D Additional Results and Diagnostics

In this section, we supplement the main results with validation diagnostics (Section D.1), gradient-search trajectories (Section D.2), and incremental costs and endpoints (Section D.3). The target-mixture distances used in Section 5.1 are given in Table 17.

D.1 Additional Validation Diagnostics

The completed nominal step-10 aggregation comparison is reported in Table 4. Here we separate ranking within the predicted head, feasible top-1 regret, and weight concentration. All four readers share the same 64 screened IDs within each fitting fold and the same nominal step-10 loss matrix. These are complementary diagnostics of the completed experiment in Table 4.

Table 13: **Ranking within the predicted head.** Mean held-out Spearman restricted to each reader's 16 highest-predicted candidates. C/S: cluster/sample; E/W: equal/weighted. Bold: highest per target. The selected head can differ between readers.

Target	C–E	C–W	S–E	S–W
General	0.511	0.571	0.471	**0.575**
Parsing	0.234	0.251	0.202	**0.310**
Knowledge	0.312	**0.368**	0.356	0.360
Overall	0.254	**0.359**	0.244	0.357

Table 14: **Feasible top-1 regret.** Mean score shortfall against the best capacity-feasible held-out probe. Smaller is better; bold marks the lowest per target. Fitting and screening folds are unchanged.

Target	C–E	C–W	S–E	S–W
General	0.773	0.534	0.962	**0.331**
Parsing	1.240	**0.617**	1.362	0.987
Knowledge	0.699	**0.356**	0.449	0.378
Overall	0.727	**0.625**	0.773	0.660

Table 15: **Weight concentration across fitting folds.** Mean support and maximum sample weight across 20 fits, each screening 64 examples. These are fold-level fits, distinct from the deployed all-probe objective in Table 5.

Target	Positive weights	Largest weight	Feasible global ρ
General	15.3	0.347	0.704
Parsing	12.2	0.374	0.475
Knowledge	13.6	0.250	0.615
Overall	16.1	0.416	0.589

D.2 Recorded Gradient Search

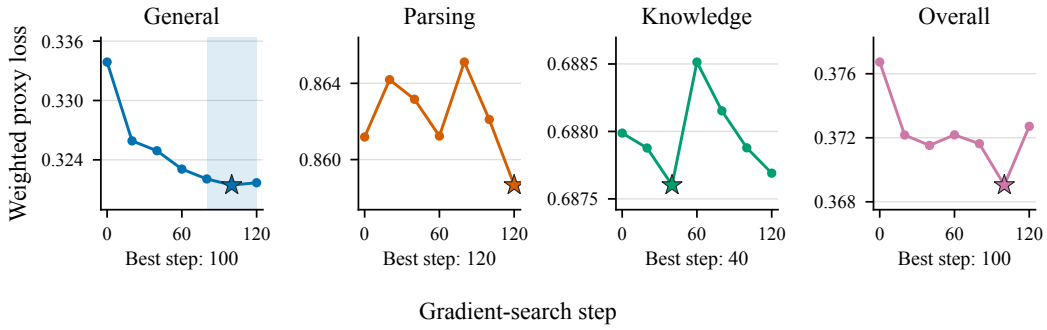

Figure 4: **Gradient search on frozen validation objectives.** Full-panel evaluations every 20 steps for $K = 100$; stars mark selected minima. Remerged step-zero losses also define D_J in Table 2. General's shaded region retains at least 95% of its observed loss decrease with two follow-up checks; the other targets show no sustained plateau by this criterion. This describes observed stability, not guaranteed convergence.

The trajectories supplement the search procedure in Section 3.4. Table 16 gives the remerged step-zero losses used by both the traces and the within-run decreases in Table 2. Final-training means and uncertainty are in Tables 1 and 9; no gain over a real-trained starting model is inferred from these proxy traces.

Observed stability, not a convergence certificate. The trajectories contain seven full-panel BF16 evaluations, at steps $0, 20, \ldots, 120$; lines connect recorded points and do not supply intermediate losses. Each target keeps its expert bank and validation weights fixed. The best recorded steps are 100, 120, 40, and 100 for General, Parsing, Knowledge, and Overall. For a descriptive stability check, set $\epsilon = 0.05(J_0 - J_{\min})$ using the remerged initial loss. A near-best plateau begins at the earliest recorded step whose loss and all subsequent recorded losses are at most $J_{\min} + \epsilon$, with at least two subsequent evaluations. Only General satisfies this retrospective rule, beginning at step 80. This is not an online stopping rule or evidence of stationarity. The other targets' fluctuations support checkpoint selection by full-panel loss rather than an assumption that the final iterate is best.

Table 16: **Executed search records.** $r^\star$ is selected by complete BF16 objective evaluation. J_0 is remerged at step zero and defines the within-run decrease D_J. These are proxy losses; real-training repeat statistics are reported in Table 9.

Target	Start recipe	$r^\star$	J_0	$J_\star$
General	101	100	0.333885	0.321450
Parsing	119	120	0.861185	0.858662
Knowledge	27	40	0.687988	0.687602
Overall	101	100	0.376730	0.369045

Table 17: **Target-dependent training distributions.** TV and JSD use the final 100-dimensional continuous training-share vectors, not validation weights or integerized sample counts.

Target pair	TV	JSD (bits)
General / Parsing	0.9783	0.9481
General / Knowledge	0.9999	0.9995
General / Overall	0.2610	0.2126
Parsing / Knowledge	0.9895	0.9789
Parsing / Overall	0.8823	0.8338
Knowledge / Overall	0.9765	0.9235

D.3 Incremental Search Costs and Endpoint Records

This section supplies the accounting and endpoint measurements used in Section 5.2. The fixed setup comprises the base, embedding and clustering outputs, 100 nominal step-10 experts, 128 real probes, and the fixed candidate-pool loss matrix. All costs below concern search after this setup. Final subset construction, real-model training, and downstream evaluation are separate stages.

New-target work. Table 18 separates reusable assets from target-dependent operations. The existing loss matrix covers all 10,000 candidate validation examples. A newly selected support can therefore be read without forwarding the historical mixtures again. This property does not require the new target to choose the same validation examples. New benchmark questions require evaluating the saved real probes; targets defined from already recorded benchmark scores require only score aggregation. This reuse assumes a fixed partition. Replacing the semantic encoder and reclustering requires rebuilding the affected experts and feedback; component flexibility does not imply cost-free reconstruction.

Table 18: **Reuse for a new target at fixed base and partition.** Validation selection is CPU work on cached losses, with no additional model forward or training. Final training is outside selection accounting.

Stage or artifact	New-target action
Embeddings, clusters, expert bank	Reuse
Real probe models	Reuse; no probe retraining
Fixed candidate-pool proxy losses	Reuse all 128 mixture rows
Target scores	Aggregate existing scores or evaluate new benchmarks
Validation support and weights	Screen and fit on CPU
Mixture search	Run with the new frozen objective
Selected-data model	Train from the common base after selection

Regression fitting budget. The regression comparison fits LightGBM predictors of the frozen proxy-loss objectives on nested prefixes of 256, 512, 1,024, 2,048, and 4,096 candidate mixtures. Labeling requires merged-proxy forwards, not real-model training. Five-fold CV measures predicted versus evaluated proxy loss, with splits formed separately within each prefix. In General/Parsing/Knowledge/Overall order, mean correlations rise from 0.571/0.625/0.539/0.592 at 256

candidates to 0.935/0.909/0.898/0.885 at 4,096, the highest observed values for all four targets. This latter budget supplies the full-cost comparison; it does not establish a minimum sufficient budget or regressor saturation.

Device-normalized accounting. Active GPU-hours equal allocated device count times active elapsed seconds divided by 3,600. We use H100 SXM peak BF16 capacity as the reference unit. NVIDIA lists A100 BF16 throughput as 312 TFLOPS dense (624 with sparsity) and H100 SXM as 1,979 TFLOPS with sparsity, equivalent to 989.5 dense. Thus

$$C_{\text{H100-eq}} = C_{\text{A100}} \frac{312}{989.5} \simeq 0.31531\, C_{\text{A100}}. \tag{22}$$

Recorded H100-hours carry unit weight. This metric measures allocated accelerator capacity weighted by peak BF16 throughput; runtime on a different device also depends on memory movement and communication. We apply the same precision and sparsity convention to both devices. The four gradient loops sum to 258.95 H100-hours; regression label construction takes 1,118.25 A100-hours, or 352.60 H100-equivalent hours. The normalized difference is 26.6% at these full search budgets.

Table 19: **Active incremental search cost.** Each gradient loop uses 48 H100 GPUs; regression labels use 24 A100 GPUs and serve all four targets together. Normalization follows Eq.(22). The total is computed before rounding.

Workload	Active seconds	Native GPU-h	H100-eq. h
Gradient: General	4,605.974	61.41	61.41
Gradient: Parsing	4,764.591	63.53	63.53
Gradient: Knowledge	5,390.556	71.87	71.87
Gradient: Overall	4,660.288	62.14	62.14
Gradient: four-target total	19,421.409	258.95	258.95
Regression: 4,096 shared labels	–	1,118.25	352.60

The gradient totals cover the complete 120-step active search loops. Regression totals sum allocated-device time over all 4,096 candidate merge/forward operations, including concurrently executed shards; their sum is work, not calendar latency. The label construction is charged once across four targets. CPU fitting and proposal enumeration, queueing and environment setup, shared asset construction, and final training/evaluation are excluded from this GPU-search account. The regression arm selects by predicted loss. This account compares full search budgets, not time to a common objective threshold.

Model-evaluation workload. The four gradient objectives have 15, 6, 14, and 14 positive-weight examples. Over 120 updates they require $120(15+6+14+14) = 5{,}880$ example forward/backward evaluations and $6(15+6+14+14) = 294$ periodic full-panel forwards after initialization. The regression objectives share 44 distinct active examples, giving $4{,}096 \times 44 = 180{,}224$ sample forwards for labeling. These are operation counts; answer lengths and backward work are not equalized by counting examples. Merge and expert-gradient operations are included in the active-loop times, not in the example counts.

Table 20: **Selected search checkpoints.** First best full-panel step within 120 updates. G counts adjustable shares on the same 100-expert bank; step zero means the start was retained.

Target	$G = 20$	$G = 50$	$G = 100$
General	120	80	100
Parsing	0	100	120
Knowledge	40	120	40
Overall	40	100	100

Effective granularity records. The $G = 20/50$ runs expand parent shares to the 100-dimensional mixture and compute all 100 expert-direction gradients before parent aggregation. Hence the bank-wide work is not reduced in proportion to G. Table 20 gives the first recorded minimum among

steps $0, 20, \ldots, 120$. Parsing at $G = 20$ retains step zero because none of its later evaluations improves the initial full-panel loss. Each run attains a different objective minimum, so these steps measure checkpoint selection rather than equal-quality hitting time. Cumulative costs to the selected checkpoints are reported in Table 3; the $G = 100$ loss traces remain in Figure 4.

Cumulative cost to the selected checkpoint. $C^\star$ accumulates active search work through the first recorded minimum, including iterative merging and forward/backward computation. It excludes initial loading, step-zero evaluation, queueing, and final training; A100 segments follow Eq.(22). Where evaluation-merge, synchronization, or checkpoint-saving time is recorded only for the complete loop, we report the upper endpoint of the reconstructed cost bound as a single conservative value, without proportional interpolation by step count. These accounting bounds are not statistical confidence intervals. General $K = 50$ and Knowledge $K = 20$ have incomplete interrupted-segment timing, so their accumulated costs remain lower bounds, marked $\geq$. Parsing $K = 20$ retains step zero: its incremental cost to the selected candidate is 0.00, while the full executed search consumed 19.94 H100-equivalent GPU-hours. These selected-checkpoint costs are distinct from the complete 120-update costs in Table 19 and do not measure time to a common loss threshold across K.

Table 21: **Search endpoints under the shared final-training recipe.** All scores use 100K examples, 100 optimizer steps, within-cluster Random, and the same data/training seeds. GD varies effective group count on one 100-expert bank. Regression selects using a predictor fitted on 4,096 candidate mixtures.

Target	GD 20	GD 50	GD 100	Regression
General	67.20	66.83	67.02	67.09
Parsing	79.81	80.12	80.26	80.10
Knowledge	61.42	61.57	61.83	62.17
Overall	67.99	67.85	68.02	68.22

Regression selection protocol. The regression arm uses 4,096 labels and 10,000,000 new enumeration points, choosing by predicted log loss without an exact-loss gate.

The shared labeling pool contains 115 feasible historical probes, four gradient endpoints, 994 perturbations around gradient incumbents, and 2,983 Dirichlet samples. The direct arm excludes exact gradient endpoints from final selection but retains their labels and surrounding perturbations; its General candidate is such a perturbation. This is a gradient-informed candidate pool, not an independent solver arm. The table records one real-training seed per endpoint.

E Data Pool and Example Sources

In this section, we detail the instruction collection, candidate-pool construction, and example sources underlying Section 4.1.

Collection and coverage. The upstream collection contains approximately 40.03M processed records across 30 data branches, listed in Table 22. These branches cover general visual instruction following, grounding, charts, documents, mathematical reasoning, long-context text, and video. A branch denotes a processed release, configuration, or derived variant, rather than necessarily a distinct dataset: the collection includes two MM-IFInstruct branches, several LLaVA-OneVision modalities, and a text-only variant of VisualWebInstruct. The COYO-derived branch contains synthesized QA rather than an independently released public instruction dataset.

From the collection to the selection pool. We retain 20% of each semantic region from the upstream collection, producing an active pool of 8,007,947 candidate IDs according to the cluster-capacity record. Region-wise subsampling reduces processing costs while retaining coverage of the semantic partition. All selection methods operate on this fixed pool, and the Full reference trains on the same pool. CLIVO allocates a budget of 100,000 distinct examples across its 100 semantic clusters, then samples without replacement within each cluster in the default configuration. The collection, candidate pool, and final subset therefore refer to different stages and should not be treated as interchangeable data budgets. The collection's source counts do not specify the exact source proportions of the subsampled pool or selected subsets.

Counting units and processing. The counts in Table 22 describe processed rows, not unique images or individual question–answer turns. A row may contain a single QA example, a multi-turn conversation, or a video instruction. For example, MTVQA contributes 6,678 image-level conversations covering multiple QA pairs, while LLaVAR-2 contributes 43,660 conversations rather than the larger number of instruction pairs in its public release. The LLaVA-OneVision video count is estimated from a deterministic sample of 2,048 Parquet file footers; the remaining counts are exact row totals. Consequently, the total is approximate and is not a globally deduplicated count of independent examples.

Release attribution. Counts refer to the particular components used: the Nemotron branch is VQA-9 of version 1, the mathematical branch is MM-MathInstruct-3M, and Tulu contributes its Persona-IF component. MMPR counts unweighted records rather than repeat-weighted training pairs. The processed Magpie metadata identifies the dataset family but does not retain an exact release identifier. The WildChat branch is consistent with the non-toxic WildChat-4.8M release, without an exact conversation-key join. We therefore report these two branches at the family level. Processing and conversion reduce some branch counts, but the available records do not identify an exact filtering predicate for every reduction; no uniform retention rate or additional quality-filtering procedure is assumed.

Example-level provenance. Table 23 lists the source keys for the six displayed examples and the additional Overall pair. These are example-level provenance records, not a complete source inventory or labels for every member of a cluster.

Higher-share examples occur in the corresponding s1 random-within-quota training manifests; lower-share examples are candidate-pool exemplars. Their displayed shares refer to the final continuous mixtures. The case bundle also stores the fixed-fold validation records separately, including their fitting weights and held-out correlations.

Table 22: **Instruction collection before candidate-pool subsampling.** Counts refer to processed records. Separate configurations and derived variants are listed as branches; only the video-branch count is estimated.

Branch	Dataset or configuration	Processed records
1	Effective Chart Dataset (ECD-10K Images)	321,544
2	Honey-Data-1M	1,070,304
3	Llama-Nemotron-VLM v1, VQA-9	46,745
4	MGrounding-630K	623,979
5	MM-IFInstruct-23K	22,555
6	MM-IFInstruct-23K, second branch	22,551
7	SPIQA, training split	262,524
8	CMM-Math, image-bearing training subset	5,004
9	DocReason25K	25,741
10	MP-DocReason51K	49,959
11	Enigmata-Data, generated snapshot	4,758
12	IQBench	500
13	LLaVA-OneVision Stage-2, video	$\sim$1,090,000
14	LLaVA-OneVision Stage-2, single image	2,633,205
15	LLaVA-OneVision Stage-2, text	522,701
16	LLaVA-OneVision-1.5 Instruct	21,592,501
17	LLaVAR-2 Hybrid-Instruct	43,660
18	LongAlign-10K	9,861
19	LongWriter-6K	5,926
20	Magpie-Align branch	479,743
21	MMDU-45K	28,979
22	MM-MathInstruct-3M	2,871,988
23	MTVQA, training split	6,678
24	SmolTalk2, SFT configuration	3,382,846
25	MMPR-v1.2	814,687
26	Tulu-3, Persona-IF component	29,980
27	VisualWebInstruct, conversation configuration	1,004,068
28	VisualWebInstruct, text-only derivative	558,467
29	COYO-derived synthesized QA	99,965
30	WildChat branch	2,398,331

Table 23: **Sources of the displayed training examples.** Source keys are taken from the supplied example records. Each pair shares a coarse capability label; its source keys need not match.

Target	Cluster	Share	Recorded source key
General	c60	Higher	`llava_next_raw_format`
	c38	Lower	`svit`
Parsing	c27	Higher	`ureader_chart`
	c14	Lower	`llava_cot_100k`
Knowledge	c36	Higher	`orca_agentinstruct`
	c20	Lower	`visualwebinstruct`
Overall	c26	Higher	`hme100k`
	c25	Lower	`visualwebinstruct`