QUALITY MEETS STRUCTURE: CONTEXT-AWARE TRAJECTORY SELECTION FOR AGENTIC SUPERVISED FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

The effectiveness of agentic supervised fine-tuning (SFT) depends on the quality of training trajectories and the diversity of their interaction patterns. However, existing data selection methods often overlook how decisions depend on earlier interactions and how execution patterns overlap across trajectories, resulting in redundant or less informative supervision. We introduce CATS, a graph-based selection method that estimates trajectory quality from context dependence and action complexity by comparing a frozen small language model's prediction losses across context variants. It constructs a similarity graph from ordered sequences of tool calls, natural-language replies, and tool-return statuses to capture overlap in execution patterns. Through message passing, CATS aggregates similarity-weighted trajectory scores, iteratively selects the highest-priority trajectory, and updates the priorities of unselected neighbors to discourage structural redundancy. Selecting 100K trajectories from a pool of over 600K, CATS outperforms nine baselines in average agentic and general performance across four benchmarks, improving average agentic performance by up to 2.6 percentage points over the strongest baseline. Ablations demonstrate the contributions of both quality scoring and structural selection, while subset analyses show broader coverage of execution patterns with high average trajectory quality. These findings highlight the value of jointly considering context-aware decisions and execution-pattern diversity for effective data selection in agentic SFT.

1 INTRODUCTION

Large language models (LLMs) are increasingly fine-tuned as agents that use external tools and respond to environment feedback to complete multi-step tasks (Chen et al., 2023; Zeng et al., 2024). Agentic supervised fine-tuning (SFT) develops these capabilities from multi-turn trajectories, supervising tool calls and natural-language replies conditioned on the preceding interaction history. Recent advances in synthetic data generation have made large training corpora available (Prabhakar et al., 2025), motivating the selection of informative subsets under limited training budgets. A useful subset should provide reliable, informative demonstrations while exposing the model to varied sequences of tool use, observations, and replies (Chen et al., 2026). Effective data selection for agentic SFT therefore requires considering both the quality of individual trajectories and the diversity of their interaction patterns (Wu et al., 2025; Yang et al., 2026).

Many existing data selection methods focus on instruction tuning, treating each training example as an instruction–response pair (Chen et al., 2024). They typically estimate quality through difficulty, learnability, or influence, and assess diversity using semantic representations or instruction coverage (Li et al., 2024b; Xia et al., 2024; Chen et al., 2025b; Lu et al., 2024). Their selection criteria therefore do not explicitly model multi-turn tool use or dependencies across interactions. For agentic SFT (Zeng et al., 2024), quality assessment should consider how later decisions use earlier interactions, since actions predictable from the initial request alone provide limited supervision for responding to evolving feedback. Meanwhile, trajectories with different requests and wording can still follow nearly identical sequences of tool calls, replies, and tool returns, making semantic diversity an incomplete measure of interaction-pattern coverage. Selecting these trajectories repeatedly can allocate a limited training budget to redundant supervision. This raises a central question: how

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.

can we jointly identify informative trajectories that teach context-dependent decisions and reduce redundancy in execution patterns under a fixed training budget?

To address this challenge, we introduce CATS, a graph-based selection method that couples context-aware quality assessment with structural diversity for agentic SFT. Our key insight is to assess the training value of each trajectory together with its execution-pattern overlap with other trajectories. We estimate trajectory quality from context dependence and action complexity using a frozen small model's full-context and history-masked prediction losses. To capture structural overlap, we represent each trajectory as an ordered sequence of tool names, natural-language reply labels, and tool-return statuses, and use sequence similarity to construct a graph over trajectories. Ranking trajectories by quality alone would leave this structural overlap unaccounted for, potentially allocating the budget to repeated execution patterns. We therefore use message passing (Maharana et al., 2024) over the graph to incorporate relationships among trajectories into their selection priorities. An initial aggregation combines each trajectory's quality score with similarity-weighted scores from its neighbors, favoring representatives of neighborhoods with high estimated training value. We then iteratively select the highest-priority trajectory and update the priorities of unselected neighbors to discourage repeated selection of similar execution patterns until the selection budget is filled.

We evaluate CATS against nine baselines using 100K trajectories selected from a pool of over 600K synthetic agentic trajectories. Our evaluation covers two agentic benchmarks, τ^3-bench (Barres et al., 2025) and VitaBench (He et al., 2025), and two general benchmarks, MMLU-Pro (Wang et al., 2024) and SuperGPQA (Team et al., 2025). CATS achieves the highest average performance on both the agentic and general benchmarks, improving average agentic performance by up to 2.6 percentage points over the strongest baseline. Ablation studies show that removing either the quality signal or graph-based selection reduces agentic performance, supporting the contribution of both components. Subset analyses further show that CATS covers the most distinct execution patterns among the compared methods while maintaining high average trajectory quality. Across selection budgets from 100K to 250K trajectories, CATS continues to improve and outperforms the baselines included in the budget comparison.

2 RELATED WORK

Data Selection for LLM Fine-tuning Data selection aims to identify useful training examples, as a small, carefully selected subset can match the performance of training on the full corpus (Zhou et al., 2023; Chen et al., 2024; Luo et al., 2025). To identify such examples, quality-focused methods use response quality, difficulty, or expected training benefits to score individual examples (Li et al., 2024a; Xia et al., 2024; Agarwal et al., 2025). Beyond individual example quality, diversity-focused approaches seek broader coverage by grouping or tagging examples by their content (Arthur & Vassilvitskii, 2007; Lu et al., 2024; Bai et al., 2026), or selecting examples that collectively represent the full dataset (Wei et al., 2015; Mirzasoleiman et al., 2020). To account for both considerations, recent methods combine quality and diversity through joint selection criteria (Liu et al., 2024; Chen et al., 2025b; Wu et al., 2025) and graph-based selection that considers both example difficulty and similarity (Maharana et al., 2024). These efforts emphasize the importance of selecting high-quality data with broad coverage of instructions and topics for effective LLM fine-tuning.

Agentic Fine-tuning and Trajectory Corpora Agentic fine-tuning trains LLMs to use tools and interact with environments through multi-turn trajectories that interleave tool calls, observations, and user interactions (Chen et al., 2023; Zeng et al., 2024; Qin et al., 2024; Zhang et al., 2025a). To support this training, recent pipelines generate and validate trajectories in simulated or executable environments (Prabhakar et al., 2025; Chen et al., 2025a), while related efforts assemble reusable trajectory corpora (Song et al., 2025; 2024). To select useful training data from such corpora, quality-focused methods score individual trajectories (Zhang et al., 2025b; Chen et al., 2025c; Zheng et al., 2026), but selecting solely by these scores may neglect the coverage of tool combinations and interaction patterns needed across diverse agent tasks (Barres et al., 2025; He et al., 2025). To capture diversity in these execution patterns, other methods compare alternative paths across repeated rollouts of the same task (Yang et al., 2026), but require multiple trajectories per task, limiting their use when each task has only one recorded trajectory.

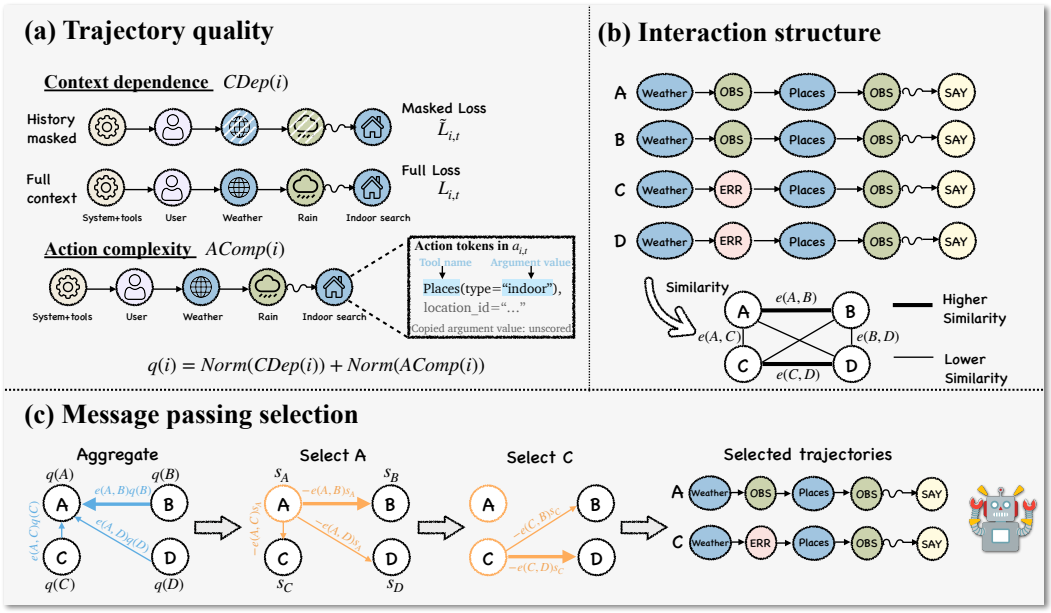


Figure 1: Overview of CATS. (a) Trajectory quality scoring using context dependence and action complexity. (b) Trajectory similarity based on ordered interaction label sequences. (c) Message passing for trajectory selection through score aggregation and iterative priority updates.

Ours. To better account for cross-turn dependencies and structural redundancy in agentic data selection, we propose CATS, a graph-based approach that connects trajectories according to the similarity of their interaction structures. By integrating quality scores derived from context dependence and action complexity with structural similarity through message passing, CATS balances trajectory quality and structural diversity, improving performance across agentic and general benchmarks.

3 METHOD

In this section, we first formulate the trajectory selection problem for agentic supervised fine-tuning (SFT) (Section 3.1). We then introduce a message-passing framework that jointly considers trajectory quality and interaction structure during selection (Section 3.2). Finally, we describe how we estimate trajectory quality through context dependence and action complexity, and measure similarity between interaction structures using ordered label sequences (Section 3.3).

3.1 AGENTIC DATA SELECTION PROBLEM

Let $\mathcal{D} = \{\tau_i\}_{i=1}^{n}$ be a deduplicated pool of n trajectories. Each trajectory $\tau_i = (x_i, \{(a_{i,t}, o_{i,t})\}_{t=1}^{T_i})$ contains an initial context x_i (system prompts, tool definitions, and the user request) and T_i assistant turns, with assistant messages $a_{i,t}$ (tool calls, text, or both) and intervening tool, environment, or user messages $o_{i,t}$. Given an integer selection budget $1 \le B \le n$, we seek an optimal subset S^* of B trajectories:

$$S^* \in \underset{S \subseteq \mathcal{D},\, |S|=B}{\arg\max}\ \mathcal{R}\big(f_{\mathrm{SFT}(\theta_0, S)}\big)\,, \tag{1}$$

where f_θ is the model with parameters θ, and $\mathrm{SFT}(\theta_0, S)$ returns parameters by fine-tuning initial parameters θ_0 on subset S. The function $\mathcal{R}$ measures expected performance on the target tasks. SFT conditions on the initial and preceding interaction context and supervises only assistant tokens.

3.2 DATA SELECTION WITH MESSAGE PASSING

Quality-based selection prioritizes trajectories with high estimated training value, but trajectories with similar interaction structures (e.g., sequences of tool calls, tool outputs, and natural-language

replies) can all receive high quality scores and be selected together, introducing redundant training information under a limited budget. To balance training value and structural diversity, we build a similarity graph from these structures and update selection priorities through the message-passing procedure illustrated in Figure 1(c). We describe graph initialization, followed by trajectory selection and priority updates, using the quality and similarity metrics defined in Section 3.3.

Message Passing. Message passing (Maharana et al., 2024) updates node states by aggregating information from neighboring nodes. In our approach, nodes represent trajectories, edges encode their similarity, and messages carry similarity-weighted scores. Before selection, forward aggregation combines each trajectory's score with those of its neighbors to prioritize high-value neighborhoods. At each iteration, we select the remaining trajectory with the highest priority and apply a reverse update to its unselected neighbors' priorities. Similarity determines the proposed reductions. This greedy procedure heuristically balances estimated training value and structural diversity.

Initialization. We represent the trajectory pool $\mathcal{D}$ as a weighted, undirected graph, with node i representing trajectory τ_i. The trajectory quality score $q(i)$ sums separately normalized context dependence and action complexity scores. Context dependence measures how strongly action-token predictions rely on preceding interactions; action complexity measures a frozen probe's mean full-context loss over action tokens after excluding copied argument-value spans. We encode each trajectory as a chronological sequence of labels for tool names, natural-language replies, and tool-return status. Similarity $e(i,j) \in [0,1]$ between trajectories i and j is one minus the normalized edit distance between these sequences. Let $\mathcal{K}(i) = \mathrm{TopK}_{j \neq i}\, e(i,j)$ contain the indices of the K most similar other trajectories, where K is the neighbor count before symmetrization. The undirected neighbor set is $N(i) = \mathcal{K}(i) \cup \{j : i \in \mathcal{K}(j)\}$. Each edge (i,j) has weight $e(i,j)$. We initialize selection priorities as $s_i = q(i)$; message passing updates s_i while $q(i)$ remains fixed.

Selection and update. Following Maharana et al. (2024), we aggregate similarity-weighted scores from neighboring trajectories to prioritize representatives of high-value neighborhoods:

$$s_i \leftarrow q(i) + \sum_{j \in N(i)} e(i,j)\, q(j). \tag{2}$$

Forward aggregation is performed once before selection. Starting from $S = \varnothing$, each iteration selects a trajectory with index k:

$$k \in \arg\max_{i:\tau_i \notin S} s_i, \qquad S \leftarrow S \cup \{\tau_k\}, \tag{3}$$

and applies a reverse update to its unselected neighbors:

$$s_j \leftarrow \begin{cases} s_j - e(k,j)\, s_k, & \text{if } s_j - e(k,j)\, s_k > 0, \\ 0, & \text{otherwise,} \end{cases} \qquad j \in N(k), \quad \tau_j \notin S, \tag{4}$$

where s_k is the priority of τ_k when selected. Similarity-weighted reductions penalize structural redundancy, while clipping at zero keeps priorities nonnegative. We repeat selection and updates until $|S| = B$.

Implementation. We precompute and cache trajectory quality scores $q(i)$ and interaction label sequences. To avoid storing the full $O(n^2)$ similarity matrix, we compute pairwise similarities in blocks, retaining each trajectory's K most similar neighbors across blocks. We symmetrize these neighbor relations and store the graph as adjacency lists with fixed edge weights. A max-priority queue, initialized with scores from forward aggregation, retrieves the highest-priority unselected trajectory at each iteration. After each reverse update, we update only the affected queue entries, avoiding a scan of all trajectories. Ties in neighbor search and selection are broken by fixed trajectory indices. The experimental setup specifies K and the hardware configuration.

3.3 Trajectory Scoring and Similarity

Scoring setup. We use a small frozen probe from the same model family as the backbone to score action tokens, defined as tokens encoding tool names and argument values in assistant messages. Let $M_{i,t}$ denote their positions in $a_{i,t}$, p_ϕ the probe token distribution with frozen parameters ϕ, and

$C_{i,t} = (x_i, a_{i,1}, o_{i,1}, \ldots, a_{i,t-1}, o_{i,t-1})$ the full context. For context C and a set of token positions M, define the summed negative log-likelihood $\mathcal{L}_{i,t}(C, M) = -\sum_{u \in M} \log p_\phi(a_{i,t,u} \mid C, a_{i,t,<u})$, where $a_{i,t,u}$ is token u in $a_{i,t}$ and $a_{i,t,<u}$ is its full token prefix. The full-context loss is $L_{i,t} = \mathcal{L}_{i,t}(C_{i,t}, M_{i,t})$. Natural-language text outside tool calls is excluded from scoring. This restriction applies only to trajectory scoring; SFT supervises all assistant tokens in the original trajectories.

Trajectory score. As illustrated in Figure 1(a), we combine context dependence and action complexity to estimate each trajectory's training value. Context dependence measures the increase in action-token prediction loss after masking preceding interaction context. At turn t, we retain x_i and all user messages and replace earlier assistant messages and tool outputs with neutral placeholders, preserving roles and order. This gives masked context $\widetilde{C}_{i,t}$ and loss $\widetilde{L}_{i,t} = \mathcal{L}_{i,t}(\widetilde{C}_{i,t}, M_{i,t})$ over the same token positions. Action complexity uses full-context loss after excluding argument-value spans copied from preceding dialogue, yielding retained token positions $M'_{i,t} \subseteq M_{i,t}$ and loss $L'_{i,t} = \mathcal{L}_{i,t}(C_{i,t}, M'_{i,t})$. We define context dependence $\mathrm{CDep}(i)$ and action complexity $\mathrm{AComp}(i)$ by averaging the loss increase and full-context loss, respectively, over scored tokens:

$$\mathrm{CDep}(i) = \frac{\sum_{t=1}^{T_i} (\widetilde{L}_{i,t} - L_{i,t})}{\sum_{t=1}^{T_i} |M_{i,t}|}, \qquad (5) \qquad\qquad \mathrm{AComp}(i) = \frac{\sum_{t=1}^{T_i} L'_{i,t}}{\sum_{t=1}^{T_i} |M'_{i,t}|}. \qquad (6)$$

Here $|M_{i,t}|$ and $|M'_{i,t}|$ denote the numbers of scored tokens. Higher positive context dependence indicates stronger reliance on preceding context, while higher action complexity indicates actions that are harder to predict given the full context. We assign zero when the corresponding denominator is zero.

For each metric, let z_i be its value for trajectory i. We use $\mathrm{Norm}(z_i) = \mathrm{rank}_{\mathrm{avg}}(z_i)/n$, where $\mathrm{rank}_{\mathrm{avg}}$ gives the ascending rank among all n trajectories, starting at 1 and assigning average ranks to ties. We combine the normalized metrics through an equally weighted sum, which achieves the best performance among the aggregation strategies evaluated in Appendix A:

$$q(i) = \mathrm{Norm}\big(\mathrm{CDep}(i)\big) + \mathrm{Norm}\big(\mathrm{AComp}(i)\big). \qquad (7)$$

Higher scores favor stronger context dependence, higher action complexity, or both. The score $q(i)$ remains fixed during selection.

Trajectory similarity. We encode each trajectory's interaction structure as an ordered label sequence c_i, following the representation illustrated in Figure 1(b). Each tool call contributes its tool name, each contiguous natural-language span in an assistant message contributes `SAY`, and each tool return contributes `ERR` if its recorded status or error information indicates an execution error, or `OBS` otherwise. Labels follow the recorded order, including the listed order of parallel tool calls. This encoding preserves repeated calls while omitting user messages, tool arguments, and reply text. We define the similarity of τ_i and τ_j as follows:

$$e(i,j) = 1 - \frac{d_{\mathrm{edit}}(c_i, c_j)}{\max\{|c_i|, |c_j|\}}, \qquad (8)$$

where d_{edit} is the min number of label insertions, deletions, and substitutions needed to transform one sequence to the other, and $|c_i|$ is the sequence length. Similarity lies in $[0, 1]$, with higher values indicating more similar structures. We use $e(i,j)$ for both neighbor selection and edge weighting.

4 Experiments

4.1 Evaluation

We evaluate agentic capabilities on τ^3-bench (Barres et al., 2025) across the retail, airline, telecom, and banking domains, and on VitaBench (He et al., 2025) across the cross-scenario, delivery, in-store, and OTA settings. Following Barres et al. (2025), we run each task four times and report pass^1 and pass^4, i.e., the average task success rate and the proportion of tasks solved in all four runs. We use pass^1 as the primary agentic metric and denote its macro-average over the two benchmarks by μ_{AGENT}. For general capabilities, we report accuracy on MMLU-Pro (Wang et al., 2024) and SuperGPQA (Team et al., 2025) using lm-evaluation-harness (Gao et al., 2024), with their macro-average denoted μ_{GENERAL}; further details are in Appendix B.

4.2 Experimental Details

We fine-tune models on 100K trajectories selected from the deduplicated pool of 603,417 under the default budget. For CATS, we set the graph neighborhood size to $K = 20$ unless otherwise specified. To assess robustness across model families, scales, and architectures, we use the base versions of QWEN3.5-4B (Team, 2026), GEMMA-4-12B (Abd et al., 2026), and QWEN3.5-35B-A3B (Team, 2026); training dataset construction and training details are provided in Appendix C.

4.3 Baselines

We compare CATS with the following baselines (see Appendix D for implementation details):

Heuristic: (1) RANDOM randomly selects a subset from the entire dataset; (2) LONGEST sorts the trajectories in descending order by the total token length of their assistant turns.

Quality-based: (3) PERPLEXITY scores each trajectory by the perplexity a small proxy model assigns to its assistant turns given the preceding interaction, favoring harder-to-predict decisions; (4) SUPERFILTER selects trajectories with the highest ratio of response losses with and without the interaction context (IFD), computed by a small model (Li et al., 2024a); (5) ALPAGASUS prompts a strong LLM to judge the quality of each trajectory (Chen et al., 2024); (6) TOOLRM ranks trajectories by scores from a reward model trained on tool-use preference data (Agarwal et al., 2025).

Diversity-based: (7) KMEANS clusters trajectory embeddings and keeps the instances closest to each centroid, thereby spanning distinct task and tool-use patterns (Arthur & Vassilvitskii, 2007); (8) INSTAG annotates each trajectory with fine-grained intention tags and retains the subset covering the largest number of distinct tags (Lu et al., 2024).

Agentic: (9) ATLAS prompts an LLM to identify steps critical to task success within each trajectory and computes the training loss only on these steps (Chen et al., 2025c).

5 Experimental Results

5.1 Main Results

CATS achieves the best average agentic and general performance across backbones. As shown in Table 1, CATS achieves the highest μ_{AGENT} and μ_{GENERAL} on both backbones under the same 100K-trajectory budget, and ranks first or second on every individual metric. It improves μ_{AGENT} over the strongest agentic baselines (LONGEST on QWEN3.5-4B and ATLAS on GEMMA-4-12B) by 1.8% and 11.1%, and μ_{GENERAL} over ALPAGASUS, the strongest general baseline, by 1.2% and 1.8%, respectively. The advantage extends to the larger MoE backbone QWEN3.5-35B-A3B (Table 5, Appendix E), where CATS leads on all metrics with gains of 6.7% and 2.9% in μ_{AGENT} and μ_{GENERAL}. These results support the effectiveness of jointly considering trajectory quality and interaction structure when selecting training data for agentic SFT.

Existing selection criteria show inconsistent advantages across tasks and backbones. As shown in Table 1, ALPAGASUS achieves the highest μ_{GENERAL} among the baselines on both backbones, but does not lead in μ_{AGENT}. Similarly, the diversity-based method KMEANS achieves the best SuperGPQA result on GEMMA-4-12B, yet this advantage does not extend to agentic performance. Furthermore, the strongest agentic baseline changes from LONGEST on QWEN3.5-4B to ATLAS on GEMMA-4-12B. These differences indicate that the advantages of existing selection criteria do not generalize consistently across evaluation tasks and model backbones. In contrast, CATS combines context-aware quality assessment with structural selection to prioritize informative trajectories while reducing redundancy in interaction patterns.

CATS achieves strong performance with substantially fewer training trajectories. CATS selects 100K trajectories, only 16.6% of the full pool. As shown in Table 1, this subset achieves 95.3% and 93.4% of the μ_{AGENT} scores obtained by FULL on QWEN3.5-4B and GEMMA-4-12B, respectively. The small performance gap despite the substantial reduction in data demonstrates the value

Table 1: Main results given by QWEN3.5-4B and GEMMA-4-12B on the agentic and general benchmarks. MMLU-P and SGPQA correspond to MMLU-Pro and SuperGPQA, respectively. The best results are highlighted in **bold**, and the second-best results are highlighted in underline.

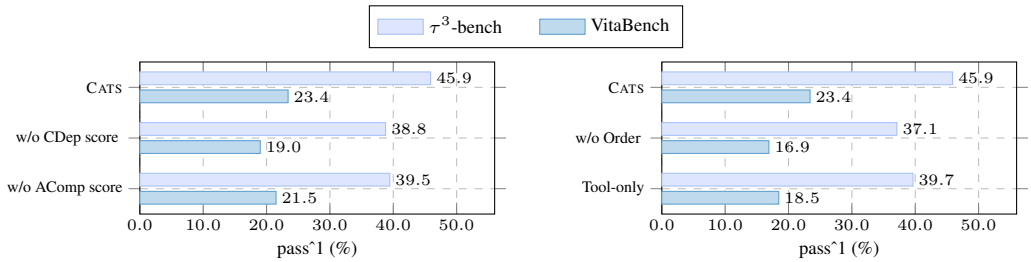
Method	Agentic					General		
	μ_{AGENT}	τ^3-bench		VitaBench		μ_{GENERAL}	MMLU-P	SGPQA
		pass^1	pass^4	pass^1	pass^4		Acc	Acc
Qwen3.5-4B								
FULL	36.4	47.1	25.2	25.6	7.5	62.1	74.0	50.2
RANDOM	31.7	43.1	23.8	20.3	4.5	58.3	68.6	48.0
LONGEST	34.1	43.6	22.3	**24.5**	6.5	64.6	78.0	51.1
PERPLEXITY	28.1	38.5	21.7	17.6	2.8	64.2	76.9	51.4
SUPERFILTER	27.1	40.2	22.9	14.0	1.5	64.5	77.6	51.3
ALPAGASUS	33.2	43.6	22.3	22.7	2.8	65.3	78.6	51.9
TOOLRM	31.6	42.2	23.2	20.9	4.5	64.6	78.2	51.0
KMEANS	32.3	43.6	23.6	21.0	**7.0**	64.8	78.1	51.4
INSTAG	31.6	41.9	20.9	21.3	2.8	64.0	77.5	50.5
ATLAS	32.9	43.8	24.7	22.0	4.0	58.1	69.2	46.9
CATS (Ours)	**34.7**	**45.9**	**26.1**	23.4	6.8	**66.1**	**79.8**	**52.3**
Gemma-4-12B								
FULL	25.7	32.9	14.2	18.4	10.8	31.3	44.3	18.3
RANDOM	19.5	26.8	10.9	12.1	5.8	32.7	43.5	21.9
LONGEST	14.1	18.1	4.6	10.1	4.8	31.9	44.6	19.1
PERPLEXITY	9.7	10.5	3.0	8.9	4.3	27.0	35.8	18.1
SUPERFILTER	11.8	14.0	3.9	9.6	4.5	29.1	38.4	19.7
ALPAGASUS	16.8	21.3	6.3	12.3	8.8	34.2	46.3	22.0
TOOLRM	16.3	23.9	7.0	8.6	4.0	30.4	42.0	18.8
KMEANS	13.8	22.2	8.3	5.3	2.5	31.9	41.5	**22.2**
INSTAG	14.8	21.6	10.4	7.9	4.8	29.2	39.1	19.2
ATLAS	21.6	28.9	11.1	14.2	8.8	30.8	43.2	18.4
CATS (Ours)	**24.0**	**31.8**	**13.6**	**16.1**	**9.3**	**34.8**	**47.5**	22.0

of selecting effective training trajectories. Moreover, CATS surpasses FULL in μ_{GENERAL} by 6.4% and 11.2%, respectively. This contrast suggests that full-data training offers modest gains in agentic performance at the cost of lower general performance. In contrast, CATS improves agentic capabilities while maintaining strong general performance. These findings highlight the data efficiency of CATS and the role of subset selection in balancing agentic and general performance.

5.2 ABLATION STUDY

Combining context dependence and action complexity improves trajectory selection. We conduct all ablations on QWEN3.5-4B and first examine how context dependence and action complexity contribute to $q(i)$. As shown in Figure 2(a), removing context dependence causes relative performance drops of 15.5% on τ^3-bench and 18.8% on VitaBench, while removing action complexity yields drops of 13.9% and 8.1%, respectively. This highlights the value of capturing both how strongly action-token predictions depend on preceding interactions and how difficult recorded decisions remain for the frozen probe to predict with full context. Combining both signals therefore enables CATS to better estimate trajectory training value and improve subset selection.

Execution order and interaction labels improve structural trajectory selection. We examine the contributions of execution order and interaction labels through the ablations in Figure 2(b). Ignoring execution order leads to relative performance drops of 19.2% on τ^3-bench and 27.8% on VitaBench, while retaining only ordered tool names reduces performance by 13.5% and 20.9%, respectively. These declines indicate that identifying structural overlap requires considering both how tool calls are ordered and how they interact with tool returns and natural-language replies. By pre-



(a) Trajectory scoring (b) Structural representation

Figure 2: Ablation study of CATS on τ^3-bench and VitaBench, measured by pass^1. (a) Trajectory scoring: removing the context dependence or action complexity component from the trajectory quality score. (b) Structural representation: ignoring execution order (*w/o Order*) or retaining only ordered tool names without the SAY, OBS, and ERR labels (*Tool-only*) when computing trajectory similarity. All variants retain complete trajectories for SFT.

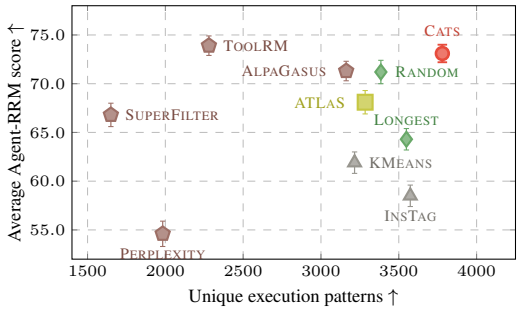


Figure 3: Quality–diversity trade-off of subsets selected by methods under the same budget B.

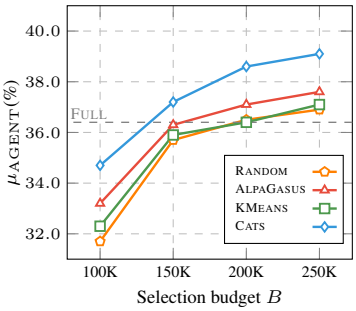


Figure 4: Effect of the selection budget B on μ_{AGENT}.

serving execution order and interaction labels, CATS distinguishes call sequences, reply placement, and tool-return statuses, providing a more informative basis for structural trajectory selection.

Jointly considering trajectory quality and structure improves subset selection. As shown in Table 2, removing the quality signal or replacing message passing with ranking by $q(i)$ reduces average performance by 7.5% and 7.8%, respectively, and either signal alone yields only marginal gains over RANDOM. The gains thus come from combining both: structural selection alone admits low-value trajectories, while ranking by $q(i)$ alone concentrates the budget on similar high-scoring ones and covers fewer distinct interaction label sequences c_i under the same budget, a diversity measure adopted from DIVE (Chen et al., 2026). Propagating quality through the similarity graph and updating priorities after each selection lets CATS reduce redundancy while preserving training value.

Table 2: Ablation of trajectory quality and structural selection in CATS. **Qual.**: use of the trajectory quality score $q(i)$; **Struct.**: message passing over the trajectory similarity graph. We report pass^1 (%) on τ^3-bench (τ^3) and VitaBench (Vita); Avg. is their macro-average.

Method	Signal		pass^1		Avg.
	Qual.	Struct.	τ^3	Vita	
RANDOM	✗	✗	43.1	20.3	31.7
w/o Quality Signal	✗	✓	43.2	20.9	32.1
w/o Graph Selection	✓	✗	43.5	20.4	32.0
CATS	✓	✓	**45.9**	**23.4**	**34.7**

6 ANALYSIS

The selected subset maintains high trajectory quality while covering a broader range of execution patterns. With QWEN3.5-4B as the backbone for all analyses, we assess subset quality

Table 3: Runtime (hours) to select 100K trajectories. Scoring / Features covers quality scoring, embedding, and tagging. Selection reports data selection time.

Method	Scoring / Features	Selection	Total (h)
RANDOM	–	0.0	0.0
LONGEST	–	0.3	0.3
PERPLEXITY	8.1	0.1	8.2
KMEANS	9.3	0.1	9.4
SUPERFILTER	11.7	0.1	11.8
TOOLRM	14.9	0.1	15.0
ALPAGASUS	24.1	0.1	24.2
ATLAS	26.9	–	26.9
INSTAG	27.9	1.8	29.7
CATS	17.6	0.2	17.8

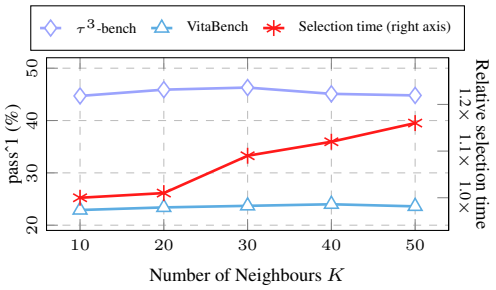


Figure 5: Sensitivity of CATS to the number of neighbors K. The left axis reports pass^1 on τ^3-bench and VitaBench, and the right axis reports selection time normalized to the runtime at (K=10).

using mean Agent-RRM scores (Fan et al., 2026) and diversity by counting distinct interaction label sequences c_i defined in Section 3.3 following DIVE (Chen et al., 2026). As shown in Figure 3, CATS achieves the broadest pattern coverage and second-highest quality under the same budget. It covers ∼3.8K patterns versus 2.3K for TOOLRM, with slightly lower quality. Meanwhile, KMEANS and INSTAG offer relatively broad coverage but substantially lower quality. This confirms that CATS expands execution-pattern coverage while maintaining high trajectory quality.

CATS remains effective across data selection budgets. We further examine CATS as the selection budget increases from 100K to 250K trajectories. As shown in Figure 4, KMEANS performs comparably to RANDOM at larger budgets, whereas ALPAGASUS consistently outperforms both, indicating that quality-based selection remains beneficial with larger subsets while clustering alone offers limited gains. CATS achieves the highest μ_{AGENT} at every budget, with its margin over ALPAGASUS widening from 150K to 250K. Moreover, CATS with 200K trajectories outperforms all baselines using 250K (20% fewer trajectories), and surpasses FULL with only 150K trajectories, roughly 24.9% of the full pool. These results show that CATS maintains its advantage as the budget grows while achieving stronger performance with fewer trajectories.

CATS introduces little structural selection overhead and maintains stable performance across neighborhood sizes. We compare runtimes for selecting 100K trajectories from the full pool using eight A100 GPUs. The results in Table 3 show that CATS completes scoring and selection in 17.8 hours, reducing runtime by 26.4% and 40.1% compared with ALPAGASUS and INSTAG, respectively. Scoring accounts for 17.6 hours, while graph construction and selection require only 0.2 hours (1.1% of the total). We further examine sensitivity to neighborhood size. As shown in Figure 5, increasing K from 10 to 50 raises selection time, while pass^1 remains relatively stable on both τ^3-bench and VitaBench. These results show that CATS incorporates structural diversity with little additional cost and remains robust across neighborhood sizes.

7 CONCLUSION

In this work, we introduce CATS, a graph-based data selection method for agentic supervised fine-tuning that jointly considers trajectory quality and interaction structure. To balance quality and structural diversity, CATS combines quality scores derived from context dependence and action complexity with similarity between ordered interaction sequences through message passing, updating selection priorities to favor informative trajectories and reduce structural redundancy. Our experiments demonstrate the effectiveness of CATS across three model backbones and four benchmarks. Compared with nine baseline methods, CATS consistently achieves superior average agentic and general performance. Our analyses validate the contributions of quality scoring and structural selection, show that the selected subsets maintain high average trajectory quality while covering a broader range of execution patterns, and demonstrate the method's continued effectiveness as selection budgets increase. These findings highlight the value of jointly considering context-dependent decisions and execution-pattern diversity for effective data selection in agentic supervised fine-tuning.

AI USE STATEMENT

During the preparation of this manuscript, we used AI-assisted tools (Codex) to refine the language and improve the clarity of presentation, including grammar correction, sentence revision, and adjustments for readability and terminological consistency. The human authors formulated the research questions, developed the methodology, designed and conducted the experiments, and interpreted the findings. All scientific claims, experimental results, and references were established and verified by the authors. All AI-assisted revisions were reviewed for accuracy and consistency with the intended meaning. The authors assume full responsibility for the integrity and final content of this manuscript.

REFERENCES

Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Carbune, Michelle Casbon, Mayank Chaturvedi, Aditya Chawla, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, Johan Ferret, Filippo Galgani, Sertan Girgin, Jean-Bastien Grill, Maarten Grootendorst, Jiaxian Guo, Cassidy Hardin, Yanzhang He, Steven M. Hernandez, Omri Homburger, Léonard Hussenot, Ju-yeong Ji, Armand Joulin, Aishwarya Kamath, Parnian Kassraie, Olivier Lacombe, Preethi Lahoti, Gaël Liu, Gus Martins, Luciano Martins, Tatiana Matejovicova, Ramona Merhej, Nikola Momchev, Sneha Mondal, Ryan Mullins, Sindhu Raghuram Panyam, Shreya Pathak, Sarah Perrin, André Susano Pinto, Etienne Pot, Angéline Pouget, Alexandre Ramé, Sabela Ramos, Douglas Reid, David Rim, Morgane Rivière, Karsten Roth, Louis Rouillard, Omar Sanseviero, Pier Giuseppe Sessa, Shane Settle, Danila Sinopalnikov, Sara Smoot, Piotr Stanczyk, Andreas Steiner, Lawrence Stewart, Ilya O. Tolstikhin, Michael Tschannen, Anton Tsitsulin, Nino Vieillard, Renjie Wu, Pingmei Xu, Haichuan Yang, Edouard Yvinec, Biao Zhang, Li Zhang, Joe Zou, Nicolas Aagnes, Abdelrahman Abdelhamed, Jakub Adámek, Shivani Agrawal, Shubham Agrawal, Ibrahim Alabdulmohsin, Jean-Baptiste Alayrac, Uri Alon, Chandramouli Amarnath, Ankesh Anand, Chrysovalantis Anastasiou, Setareh Ariafar, François-Xavier Aubet, Kyriakos Axiotis, Federico Barbero, Joelle K. Barral, Alexei Bendebury, Urs Bergmann, Stanley Bileschi, Kat Black, Mathieu Blondel, Sebastian Borgeaud, and Arthur Brazinskas. Gemma 4 technical report. *CoRR*, abs/2607.02770, 2026. doi: 10.48550/ARXIV.2607.02770. URL https://doi.org/10.48550/arXiv.2607.02770.

Mayank Agarwal, Ibrahim Abdelaziz, Kinjal Basu, Merve Unuvar, Luis A. Lastras, Yara Rizk, and Pavan Kapanipathi. Toolrm: Outcome reward models for tool-calling large language models. *CoRR*, abs/2509.11963, 2025. doi: 10.48550/ARXIV.2509.11963. URL https://doi.org/10.48550/arXiv.2509.11963.

David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein (eds.), *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pp. 1027–1035. SIAM, 2007. URL http://dl.acm.org/citation.cfm?id=1283383.1283494.

Yi Bai, Wenhao Zhang, Yao Chen, Jiao Xue, Zhumin Chen, and Pengjie Ren. MADS: model-aware diverse core set selection for instruction tuning. *CoRR*, abs/2605.30857, 2026. doi: 10.48550/ARXIV.2605.30857. URL https://doi.org/10.48550/arXiv.2605.30857.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982.

Aili Chen, Chi Zhang, Junteng Liu, Jiangjie Chen, Chengyu Du, Yunji Li, Ming Zhong, Qin Wang, Zhengmao Zhu, Jiayuan Song, Ke Ji, Junxian He, Pengyu Zhao, and Yanghua Xiao. DIVE: scaling diversity in agentic task synthesis for generalizable tool use. *CoRR*, abs/2603.11076, 2026. doi: 10.48550/ARXIV.2603.11076. URL https://doi.org/10.48550/arXiv.2603.11076.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. Fireact: Toward language agent fine-tuning. *CoRR*, abs/2310.05915, 2023. doi: 10.48550/ARXIV.2310.05915. URL https://doi.org/10.48550/arXiv.2310.05915.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=FdVXgSJhvz.

Xuanzhong Chen, Zile Qiao, Guoxin Chen, Liangcai Su, Zhen Zhang, Xinyu Wang, Pengjun Xie, Fei Huang, Jingren Zhou, and Yong Jiang. Agentfrontier: Expanding the capability frontier of LLM agents with zpd-guided data synthesis. *CoRR*, abs/2510.24695, 2025a. doi: 10.48550/ARXIV.2510.24695. URL https://doi.org/10.48550/arXiv.2510.24695.

Yicheng Chen, Yining Li, Kai Hu, Zerun Ma, Haochen Ye, and Kai Chen. MIG: automatic data selection for instruction tuning by maximizing information gain in semantic space. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 9902–9915. Association for Computational Linguistics, 2025b. doi: 10.18653/V1/2025.FINDINGS-ACL.515. URL https://doi.org/10.18653/v1/2025.findings-acl.515.

Zhixun Chen, Ming Li, Yuxuan Huang, Yali Du, Meng Fang, and Tianyi Zhou. ATLAS: agent tuning via learning critical steps. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 25334–25349. Association for Computational Linguistics, 2025c. doi: 10.18653/V1/2025.FINDINGS-ACL.1299. URL https://doi.org/10.18653/v1/2025.findings-acl.1299.

Kaixuan Fan, Kaituo Feng, Manyuan Zhang, Tianshuo Peng, Zhixun Li, Yilei Jiang, Shuang Chen, and Xiangyu Yue. Exploring reasoning reward model for agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 1975–1991. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-ACL.95. URL https://doi.org/10.18653/v1/2026.findings-acl.95.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, Man Gao, Xi Su, Xiaodong Cai, Xunliang Cai, Yu Yang, and Yunke Zhao. Vitabench: Benchmarking LLM agents with versatile interactive tasks in real-world applications. *CoRR*, abs/2509.26490, 2025. doi: 10.48550/ARXIV.2509.26490. URL https://doi.org/10.48550/arXiv.2509.26490.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 14255–14273. Association for Computational Linguistics, 2024a. doi: 10.18653/V1/2024.ACL-LONG.769. URL https://doi.org/10.18653/v1/2024.acl-long.769.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024, pp. 7602–7635. Association for Computational Linguistics, 2024b. doi: 10.18653/V1/2024.NAACL-LONG.421. URL https://doi.org/10.18653/v1/2024.naacl-long.421.

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. In *International Conference on Learning Representations*, volume 2024, pp. 22353–22373, 2024.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. #instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=pszewhybU9.

Junyu Luo, Bohan Wu, Xiao Luo, Zhiping Xiao, Yiqiao Jin, Rong-Cheng Tu, Nan Yin, Yifan Wang, Jingyang Yuan, Wei Ju, and Ming Zhang. A survey on efficient large language model training: From data-centric perspectives. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 30904–30920. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-LONG.1493. URL https://doi.org/10.18653/v1/2025.acl-long.1493.

Adyasha Maharana, Prateek Yadav, and Mohit Bansal. D2 pruning: Message passing for balancing diversity & difficulty in data pruning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=thbtoAkCe9.

Baharan Mirzasoleiman, Jeff A. Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6950–6960. PMLR, 2020. URL http://proceedings.mlr.press/v119/mirzasoleiman20a.html.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Manoj Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/5e3661f7fe4c8ac5652d62eb3d3c96ea-Abstract-Datasets_and_Benchmarks_Track.html.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Yifan Song, Weimin Xiong, Xiutian Zhao, Dawei Zhu, Wenhao Wu, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. Agentbank: Towards generalized LLM agents via fine-tuning on 50000+ interaction trajectories. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, volume EMNLP 2024 of *Findings of ACL*, pp. 2124–2141. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-EMNLP.116. URL https://doi.org/10.18653/v1/2024.findings-emnlp.116.

Yueqi Song, Ketan Ramaneti, Zaid Sheikh, Ziru Chen, Boyu Gou, Tianbao Xie, Yiheng Xu, Danyang Zhang, Apurva Gandhi, Fan Yang, Joseph Liu, Tianyue Ou, Zhihao Yuan, Frank Xu, Shuyan Zhou, Xingyao Wang, Xiang Yue, Tao Yu, Huan Sun, Yu Su, and Graham Neubig. Agent data protocol: Unifying datasets for diverse, effective fine-tuning of LLM agents. *CoRR*, abs/2510.24702, 2025. doi: 10.48550/ARXIV.2510.24702. URL https://doi.org/10.48550/arXiv.2510.24702.

M-A-P Team, Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Kang Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, Kaixing Deng, Shuyue Guo, Shian Jia, Sichao Jiang, Yiyan Liao, Rui Li, Qinrui Li, Sirun Li, Yizhi Li, Yunwen Li, Dehua Ma, Yuansheng Ni, Haoran Que, Qiyao Wang, Zhoufutu Wen, Siwei Wu, Tianshun Xing, Ming Xu, Zhenzhu Yang, Zekun Moore Wang, Junting Zhou, Yuelin Bai, Xingyuan Bu, Chenglin Cai, Liang Chen, Yifan Chen, Chengtuo Cheng, Tianhao Cheng, Keyi Ding, Siming Huang, Yun Huang, Yaoru Li, Yizhe Li, Zhaoqun Li, Tianhao Liang, Chengdong Lin, Hongquan Lin, Yinghao Ma, Zhongyuan Peng, Zifan Peng, Qige Qi, Shi Qiu, Xingwei Qu, Yizhou Tan, Zili Wang, Chenqing Wang, Hao Wang, Yiya Wang, Yubo Wang, Jiajun Xu, Kexin Yang, Ruibin Yuan, Yuanhao Yue, Tianyang Zhan, Chun Zhang, Jingyang Zhang, Xiyue Zhang, Xingjian Zhang, Yue Zhang, Yongchi Zhao, Xiangyu Zheng, Chenghua Zhong, Yang Gao, Zhoujun Li, Dayiheng Liu, Qian Liu, Tianyu Liu, Shiwen Ni, Junran Peng, Yujia Qin, Wenbo Su, Guoyin Wang, Shi Wang, Jian Yang, Min Yang, Meng Cao, Xiang Yue, Zhaoxiang Zhang, Wangchunshu Zhou, Jiaheng Liu, Qunshu Lin, Wenhao Huang, and Ge Zhang. Supergpqa: Scaling llm evaluation across 285 graduate disciplines, 2025. URL https://arxiv.org/abs/2502.14739.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/ad236edc564f3e3156e1b2feafb99a24-Abstract-Datasets_and_Benchmarks_Track.html.

Kai Wei, Rishabh K. Iyer, and Jeff A. Bilmes. Submodularity in data subset selection and active learning. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1954–1963. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/wei15.html.

Minghao Wu, Thuy-Trang Vu, Lizhen Qu, and Gholamreza Haffari. The best of both worlds: Bridging quality and diversity in data selection with bipartite graph. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/wu25ac.html.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: selecting influential data for targeted instruction tuning. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

Jinluan Yang, Yuxin Liu, Zhengyu Chen, Chengcheng Han, Yueqing Sun, Qi Gu, Hui Su, Xunliang Cai, Fei Wu, and Kun Kuang. Topocurate:modeling interaction topology for tool-use

agent training. *CoRR*, abs/2603.01714, 2026. doi: 10.48550/ARXIV.2603.01714. URL https://doi.org/10.48550/arXiv.2603.01714.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, volume ACL 2024 of *Findings of ACL*, pp. 3053–3077. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.181. URL https://doi.org/10.18653/v1/2024.findings-acl.181.

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Akshara Prabhakar, Haolin Chen, Zhiwei Liu, Yihao Feng, Tulika Manoj Awalgaonkar, Rithesh R. N., Zeyuan Chen, Ran Xu, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. xlam: A family of large action models to empower AI agent systems. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pp. 11583–11597. Association for Computational Linguistics, 2025a. doi: 10.18653/V1/2025.NAACL-LONG.578. URL https://doi.org/10.18653/v1/2025.naacl-long.578.

Yunxiao Zhang, Guanming Xiong, Haochen Li, and Wen Zhao. EDGE: efficient data selection for LLM agents via guideline effectiveness. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025, Montreal, Canada, August 16-22, 2025*, pp. 8384–8392. ijcai.org, 2025b. doi: 10.24963/IJCAI.2025/932. URL https://doi.org/10.24963/ijcai.2025/932.

Dewu Zheng, Ruizhe Ye, Yanlin Wang, Yang Ye, Hongyu Zhang, Ensheng Shi, Xilin Liu, Yuchi Ma, Jianxing Yu, and Zibin Zheng. Swe-prime: Fewer trajectories, better performance. *arXiv preprint arXiv:2608.27449*, 2026.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: less is more for alignment. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/ac662d74829e4407ce1d126477f4a03a-Abstract-Conference.html.

A AGGREGATION OF CDEP AND ACOMP

We examine how different aggregation strategies for context dependence and action complexity affect trajectory selection. Let $u_i = \mathrm{Norm}(\mathrm{CDep}(i))$ and $v_i = \mathrm{Norm}(\mathrm{AComp}(i))$ denote the rank-normalized scores defined in the main text. We compare four strategies: sum, product, minimum, and maximum. Each strategy replaces the definition of $q(i)$ while keeping the trajectory similarity graph and message-passing procedure unchanged. All variants select 100K trajectories from the same candidate pool and fine-tune QWEN3.5-4B using identical training and evaluation settings.

Table 4: Comparison of aggregation strategies for CDep and AComp. We report pass^1 (%) on τ^3-bench and VitaBench, together with their macro-average μ_{AGENT}. The best results are highlighted in bold.

Aggregation	$q(i)$	τ^3-bench	VitaBench	μ_{AGENT}
Sum (default)	$u_i + v_i$	**45.9**	**23.4**	**34.7**
Product	$u_i v_i$	42.7	21.9	32.3
Minimum	$\min(u_i, v_i)$	42.0	20.1	31.1
Maximum	$\max(u_i, v_i)$	43.2	21.3	32.3

As shown in Table 4, the equally weighted sum achieves the best performance on both benchmarks, reaching a μ_{AGENT} of 34.7% and outperforming the next-best strategies by 2.4 percentage points. The product and minimum yield lower average scores of 32.3% and 31.1%, respectively. Both aggregation rules limit the extent to which a high score in one metric can compensate for a low score in the other. Maximum aggregation yields an average score of 32.3%, indicating that retaining the stronger signal alone is less effective than combining both. In contrast, additive aggregation preserves contributions from both metrics while allowing a higher score in one dimension to compensate for a lower score in the other. These results demonstrate that equally weighted addition provides the most effective combination of context dependence and action complexity among the evaluated strategies, yielding the strongest downstream agentic performance.

B EVALUATION DETAILS

We evaluate the fine-tuned models on four benchmarks covering agentic capabilities and general knowledge and reasoning:

- **τ^3-bench** evaluates conversational agents on customer-service tasks requiring tool use, user interaction, and adherence to domain-specific policies. Building on τ^2-bench (Barres et al., 2025), the updated benchmark includes retail, airline, telecom, and banking domains. Agents interact with simulated users and domain-specific tools to complete tasks, with success determined by the benchmark's task-specific evaluation criteria. We use `gpt-4.1-2025-04-14` as the user simulator and retain the official default limit of 200 simulation steps.
- **VitaBench** (He et al., 2025) evaluates agents on interactive tasks in food delivery, in-store consumption, and online travel services. The benchmark provides 66 tools and 400 tasks, comprising 100 cross-scenario tasks and 300 single-scenario tasks. These tasks require agents to coordinate tool calls, clarify ambiguous requests, and track changing user intent through multi-turn interactions. Following the original setup, we use `gpt-4.1-2025-04-14` as the user simulator and CLAUDE-3.7-SONNET as the rubric-based sliding-window evaluator, without imposing a fixed interaction-round limit.
- **MMLU-Pro** (Wang et al., 2024) assesses knowledge and reasoning across 14 academic disciplines, including mathematics, physics, engineering, law, and psychology. The dataset contains approximately 12,000 multiple-choice questions with up to ten answer options, placing greater emphasis on reasoning than the original MMLU benchmark. We report answer accuracy.
- **SuperGPQA** (Team et al., 2025) assesses graduate-level knowledge and reasoning across 285 disciplines. The multiple-choice questions span a broad range of specialized academic

subjects and evaluate models' ability to apply domain-specific knowledge. We report answer accuracy.

For both agentic benchmarks, we run each task four times with the LLM temperature set to 0.0. We report pass^1 and pass^4, where pass^k measures the probability of task success in all k independent trials, averaged across tasks. We use pass^1 as the primary agentic metric and compute μ_{AGENT} as its macro-average across the two benchmarks. For MMLU-Pro and SuperGPQA, we use `lm-evaluation-harness`[1] and compute μ_{GENERAL} as the macro-average of their accuracy scores.

C Training Data and Details

Training Dataset. We construct our candidate pool from two publicly available agentic corpora, Nemotron-Agentic-v1[2] and Nemotron-SFT-Agentic-v2[3], which provide synthetic multi-turn trajectories interleaving user requests, tool calls, and environment observations. Since our evaluation focuses on tool-calling and interactive agents, we retain the tool-calling and interactive-agent portions and exclude retrieval data. We merge and deduplicate the retained data to obtain a static pool of 603,417 trajectories, with one recorded trajectory per task. Unless otherwise stated, each selection method selects 100K trajectories from this pool, while Full uses the entire pool.

Hyperparameters. All experiments use the same training hyperparameters across selection methods and model backbones. We train for 5 epochs with a global batch size of 32 and a micro-batch size of 1 per device. We use a learning rate of 1×10^{-5}, a warmup ratio of 0.05, and a cosine learning rate schedule with a minimum learning rate of 1×10^{-6}. The optimizer parameters are $\beta_1 = 0.9$, $\beta_2 = 0.95$, and $\epsilon = 10^{-8}$. We set the weight decay to 0.1 and the gradient clipping threshold to 1.0. The maximum sequence length is 65,536 tokens, and the random seed is fixed at 42.

Computation Infrastructure. All models are trained using `Megatron-SWIFT` on 16 NVIDIA A100 GPUs, each with 80 GB of memory. We use BF16 precision and enable sequence packing, FlashAttention, and full activation recomputation to support efficient training on long agent trajectories.

D Baselines

We compare Cats with nine baselines covering heuristic, quality-based, diversity-based, and agentic approaches. All methods use the same candidate pool and a budget of 100K trajectories unless otherwise specified. For Cats, Perplexity, and SuperFilter, we use a frozen base model from the same family as the fine-tuning backbone: Qwen3.5-0.8B-Base[4] for the Qwen3.5 backbones and Gemma-4-E2B[5] for Gemma-4-12B. Within each backbone setting, these methods share the same probe checkpoint, input formatting, and truncation policy. We describe the baseline implementations below.

- Random uniformly samples trajectories from the candidate pool without replacement.
- Longest ranks trajectories by the total number of tokens in their assistant turns and selects the longest trajectories. The token count includes both natural-language responses and tool calls, excluding user messages and tool observations. We use the tokenizer of Qwen3.5-0.8B-Base to obtain a consistent length ranking across fine-tuning backbones.
- Perplexity scores trajectories using the corresponding frozen probe. Each assistant turn is evaluated under the complete preceding interaction, including user messages, earlier assistant responses, and tool observations. We compute trajectory-level perplexity by averaging the negative log-likelihood over all assistant response tokens and exponentiating the result. Trajectories with the highest perplexity scores are retained.

[1] `https://github.com/EleutherAI/lm-evaluation-harness`
[2] `https://huggingface.co/datasets/nvidia/Nemotron-Agentic-v1`
[3] `https://huggingface.co/datasets/nvidia/Nemotron-SFT-Agentic-v2`
[4] `https://huggingface.co/Qwen/Qwen3.5-0.8B-Base`
[5] `https://huggingface.co/google/gemma-4-E2B`

- SUPERFILTER (Li et al., 2024a) selects trajectories according to instruction-following difficulty (IFD), defined as the ratio of response losses with and without the interaction context. We use the same frozen probe as PERPLEXITY for each fine-tuning backbone. For each assistant turn, the conditional setting retains the complete preceding interaction, whereas the reference setting removes user messages and preceding assistant and tool messages while retaining system instructions and tool definitions. Both settings score the same assistant response tokens. We compute the mean negative log-likelihood over all assistant response tokens across turns in each setting, and obtain the trajectory-level IFD by dividing the conditional loss by the reference loss. We retain the highest-scoring trajectories with IFD scores below 1.
- ALPAGASUS (Chen et al., 2024) uses an LLM to assess training-data quality. We prompt `qwen3.7-max` to assign each complete trajectory an integer accuracy score from 0 to 5 based on its adherence to the user instruction, the applicable policy, and tool outputs. The input contains JSON-serialized tool definitions, the initial user request, and the complete interaction. For trajectories from the `interactive_agent` source, we extract the longest `<policy>` block from the system message; otherwise, the policy field is set to `None`. We retain the highest-scoring trajectories under the common selection budget. The scoring prompt is provided in Table 6.
- TOOLRM (Agarwal et al., 2025) uses a reward model trained on tool-use preference data to assess trajectory quality. We evaluate each assistant turn with the preceding conversation and available tool definitions as context. The scalar rewards are averaged across assistant turns to obtain a trajectory-level score, and the highest-scoring trajectories are retained.
- KMEANS (Arthur & Vassilvitskii, 2007) selects representative trajectories through clustering in the embedding space. We encode each complete trajectory with QWEN3-EMBEDDING-0.6B[6], taking the hidden state of the last non-padding token and applying ℓ_2 normalization to obtain a 1,024-dimensional representation. We partition the candidate pool into 1,000 clusters using mini-batch k-means and allocate selection quotas in proportion to cluster size. Within each cluster, we retain the trajectories closest to the centroid, adjusting integer quotas to match the total selection budget.
- INSTAG (Lu et al., 2024) selects trajectories based on the coverage of fine-grained intention tags. We provide `qwen3.7-max` with the user messages from each session in chronological order and request three to eight lowercase noun phrases describing the user's goals and the concepts involved. The prompt covers intentions expressed across turns while excluding tool implementation details and non-substantive acknowledgments. Sessions without substantive goals receive an empty tag set. We encode the generated tag vocabulary with QWEN3-EMBEDDING-0.6B to consolidate near-duplicate expressions and select trajectories to increase distinct tag coverage. The tagging prompt is provided in Table 6.
- ATLAS (Chen et al., 2025c) identifies critical steps within trajectories and restricts supervision to those steps. To accommodate the 100K-trajectory budget, we independently sample 100K trajectories uniformly without replacement from the common candidate pool using a fixed random seed. This subset is sampled separately from RANDOM and shared across all backbones. Each trajectory is presented to `qwen3.7-max` as a sequence of indexed agent steps. Under the specified step-selection budget, the annotator identifies critical steps involving planning, interpretation of key observations, consequential actions, or self-correction. Complete trajectories remain available as context during fine-tuning, while the loss is computed only on assistant tokens in the selected steps. Step annotations are converted into token masks using each backbone's tokenizer. Annotation prompts are provided in Table 6.

For ALPAGASUS, INSTAG, and ATLAS, we use the same `qwen3.7-max` configuration, with thinking disabled and the sampling temperature set to 0.0. All methods follow the backbone-specific training configuration in Appendix C. Standard supervision over all assistant response tokens is used except for the critical-step masking in ATLAS. Comparisons therefore match the number of training trajectories, while the number of supervised tokens depends on trajectory length and the supervision procedure.

[6] `https://huggingface.co/Qwen/Qwen3-Embedding-0.6B`

Table 5: Main results given by QWEN3.5-35B-A3B on the agentic and general benchmarks. MMLU-P and SGPQA correspond to MMLU-Pro and SuperGPQA, respectively. The best results are highlighted in **bold**, and the second-best results are highlighted in underline.

Method	Agentic					General		
	μ_{AGENT}	τ^3-bench		VitaBench		μ_{GENERAL}	MMLU-P	SGPQA
		pass^1	pass^4	pass^1	pass^4		Acc	Acc
Qwen3.5-35B-A3B								
RANDOM	36.9	46.8	24.2	26.9	5.8	64.6	75.5	53.6
ALPAGASUS	37.5	48.3	24.7	26.6	6.0	<u>69.9</u>	<u>81.2</u>	<u>58.5</u>
ATLAS	38.6	<u>49.2</u>	<u>27.8</u>	<u>28.0</u>	<u>7.8</u>	61.7	73.5	49.9
CATS (Ours)	**41.2**	**51.4**	**30.9**	**31.0**	**10.0**	**71.9**	**83.4**	**60.3**

E RESULTS ON A LARGER MOE BACKBONE

CATS achieves the best agentic and general performance on a larger MoE backbone. We further evaluate CATS on QWEN3.5-35B-A3B to examine whether its benefits extend to a larger mixture-of-experts model. As shown in Table 5, CATS achieves the highest μ_{AGENT} and μ_{GENERAL} among the compared methods, while also leading on every individual metric. Specifically, CATS improves μ_{AGENT} over ATLAS, the strongest agentic baseline, by 6.7%, and surpasses ALPAGASUS, the strongest general baseline, by 2.9% in μ_{GENERAL}. The improvements in both pass^1 and pass^4 on the two agentic benchmarks indicate gains in average task success and consistency across repeated runs. Together with the main results, these findings support the effectiveness of jointly considering trajectory quality and interaction structure across the tested model scales and architectures, improving agentic capabilities while maintaining strong general performance.

Table 6: Prompts used for trajectory scoring with ALPAGASUS, intention tagging with INSTAG, and critical-step annotation with ATLAS. Braced fields are populated for each trajectory. For ATLAS, `n` and `n_steps` denote the number of indexed agent steps, `m` denotes the selection percentage, and `k` denotes the maximum number of selected steps.

ALPAGASUS: Trajectory scoring

System message

```
We would like to request your feedback on the performance of an AI agent
in completing the user's task using the available tools, as displayed
following.

### Available Tools:
{tools}

### Policy:
{policy}

### Task:
{task}

### Agent Trajectory:
{trajectory}
```

User message

```
Please rate according to the accuracy of the agent's trajectory in
following the instruction, the policy, and the tool outputs.  Each
trajectory receives a score on a scale of 0 to 5, where a higher score
indicates higher level of the accuracy.  Please first output a single
line containing value indicating the scores.  In the subsequent line,
please provide a comprehensive explanation of your evaluation, avoiding
any potential bias.
(Output format:  the first line MUST be a single integer from 0 to 5 and
nothing else.)
```

INSTAG: Intention tagging

```
Below are all the substantive messages a user sent across one session
while interacting with an AI agent, separated by "---".  Together they
express the user's overall goal(s) for this session.

Output a set of fine-grained intention/semantics tags that describe WHAT
THE USER WANTS and WHAT CONCEPTS ARE INVOLVED, following these rules:
- open-ended (no fixed tag set); each tag a short lowercase noun phrase
- describe user intent/topic, NOT the tools or how it is implemented
- cover the union of the user's goals across all messages
- ignore pure confirmations, IDs, and acknowledgments
- output 3 to 8 tags; if the user expresses no substantive goal at all,
output []
- output ONLY a JSON array of tag strings, no explanation, no code fences

User messages:
{user_messages}
```

Continued on next page

Table 6 (continued)

ATLAS: Critical-step annotation

System message

```
You are an expert annotator for AI-agent training data.  You will be
shown a full multi-turn trajectory in which an AI agent solves a task
using tools.  The agent's turns are segmented into STEPS (marked as
"[STEP k | ...]").  Each STEP is one agent action:  its reasoning/reply
text and/or the tool call it makes.

Your task:  identify the CRITICAL steps | the small subset of steps that
are decisive for task success and most worth training a model on.  A step
is critical if it fits ONE of these four categories:
- "plan_creation":  the agent creates or substantially revises a plan, or
decomposes the task into sub-goals.
- "critical_observation":  the agent correctly extracts or interprets KEY
information from a tool result / environment feedback that shapes what it
does next.
- "critical_action":  the agent takes a decisive action (typically THE key
tool call) that directly drives task success.
- "self_correction":  the agent detects a mistake, dead-end, or wrong
assumption and corrects course.

Rules:
- Select AT MOST {k} of the {n} steps (i.e.  ≤ {m}%).  Fewer is better
than more.  Select at least 1.
- Only STEP indices may be selected.  Never select user or observation
turns.
- Prefer non-obvious, pivotal steps.  SKIP routine/boilerplate/filler:
greetings, restating the request, trivially predictable formatting,
obvious confirmations.
- Judge criticality from the WHOLE trajectory (a step can be critical
because of what it enables later).

Output STRICT JSON on a single line, no prose, no markdown fences:
{"critical_steps":  [{"step":  <int>, "category":  "<one of the four>"},
...]}
```

User message

```
Trajectory has {n_steps} steps (STEP 1..{n_steps}).  Select at most {k}
critical steps.

=== TRAJECTORY START ===

{transcript}

=== TRAJECTORY END ===
```