From Tags to Trees: Structuring Fine-Grained Knowledge for Controllable Data Selection in LLM Instruction Tuning

Anonymous ACL submission

Abstract

Effective and controllable data selection is critical for LLM instruction tuning, especially with massive open-source datasets. Existing approaches primarily rely on instance-level quality scores, or diversity metrics based on embedding clusters or semantic tags. However, constrained by the flatness of embedding spaces or the coarseness of tags, these approaches overlook fine-grained knowledge and its intrinsic hierarchical dependencies, consequently hindering precise data valuation and knowledge-aligned sampling. To address this challenge, we propose Tree-aware Aligned Global Sampling (TAGS), a unified framework that leverages a knowledge tree built from fine-grained tags, thereby enabling joint control of global quality, diversity, and target alignment. Using an LLM-based tagger, we extract atomic knowledge concepts, which are organized into a global tree through bottom-up hierarchical clustering. By grounding data instances onto this tree, a tree-aware metric then quantifies data quality and diversity, facilitating effective sampling. Our controllable sampling strategy maximizes tree-level information gain and enforces leaf-level alignment via KL-divergence for specific domains. Extensive experiments demonstrate that TAGS significantly outperforms state-of-the-art baselines. Notably, it surpasses the full-dataset model by **+5.84%** using only **5%** of the data, while our aligned sampling strategy further boosts average performance by **+4.24%**.

1 Introduction

Large Language Models (LLMs) have become a cornerstone of natural language processing, demonstrating remarkable performance across diverse applications (Yang et al., 2025; Liu et al., 2024a; Achiam et al., 2023). While pre-training endows LLMs with extensive world knowledge, instruction tuning serves to bridge the gap between raw probabilistic modeling and human intent (Ouyang et al., 2022; Wei et al., 2021). As a result, the quality and diversity of instruction data have become critical factors in determining the performance of LLMs. The recent emergence of large-scale open-source instruction datasets (Lambert et al., 2024; Teknium, 2023) has significantly democratized access to instruction tuning. However, simply scaling up data size does not always guarantee better performance. Recent studies (Zhou et al., 2023a; Ge et al., 2024; Chen et al., 2025) reveal the *"less is more"* phenomenon, where models trained on carefully curated, high-quality subsets consistently outperform those trained on massive, unfiltered corpora. These findings have shifted research attention from indiscriminate scaling toward effective data selection. In particular, extracting high-value instruction subsets not only reduces training costs but also mitigates the *alignment tax*—the degradation in model capability caused by redundant, noisy instructions (Ouyang et al., 2022; Zhou et al., 2023a). As such, effective data selection has become critical in ensuring optimal LLM performance.

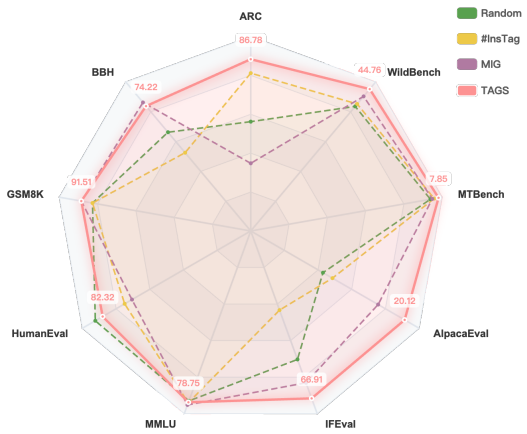

Figure 1: Performance comparison of Qwen3-8B fine-tuned on Tulu3. TAGS consistently outperforms other data selection strategies.

Driven by this demand, numerous automated data selection methods have been proposed (Lu

et al., 2023; Ge et al., 2024; Chen et al., 2025). These approaches can generally be categorized into three types: metric-based filtering, diversity-driven selection, and comprehensive strategies. Metric-based filtering methods (Chen et al., 2023; Li et al., 2024a; Liu et al., 2024b) rely on scalar indicators such as quality or complexity scores to filter out noisy data. However, they evaluate samples independently, failing to consider global diversity at the subset level. Diversity-driven selection methods (Yin et al., 2024; Bukharin et al., 2024) focus on maximizing semantic coverage. Nevertheless, the use of flattened representations limits their ability to capture fine-grained conceptual distinctions. Comprehensive strategies (Lu et al., 2023; Ge et al., 2024; Chen et al., 2025) attempt to balance quality and diversity by using tagging or clustering. Despite this, these methods typically operate over coarse labels (e.g., *"Math"*) and shallow structures, which cannot distinguish atomic concepts like *"matrix decomposition"* and *"linear equations"*. Crucially, all these paradigms operate over flat semantic spaces and overlook fine-grained knowledge and its intrinsic tree-structured hierarchy. This leads to two fundamental bottlenecks: (1) Imprecise assessment of data value and subset diversity due to the lack of a fine-grained topology. (2) Uncontrollable sampling that struggles to meet specific domain constraints or user requirements.

To address these limitations, we propose TAGS (**T**ree-aware **A**ligned **G**lobal **S**ampling), a unified framework that incorporates fine-grained knowledge topology into controllable data selection. First, we construct a global Knowledge Tree to capture fine-grained knowledge structures. Rather than relying on coarse labels, we utilize a specialized lightweight tagger to extract atomic concepts from large-scale data (Xu et al., 2024; LI et al., 2024; Zheng et al., 2023a; Teknium, 2023). These concepts are then organized via hierarchical clustering, forming a tree where leaf nodes represent specific knowledge units and upper levels represent broader semantic categories. Second, we introduce a Tree-based Information Propagation mechanism to evaluate data utility in a structured semantic space. Specifically, TAGS anchors data instances onto the constructed tree, evaluating data value through a tree-based information flow. Data quality drives the information intensity at leaf nodes, which then propagates to quantify global richness of the subset. To mitigate redundancy, we employ a concave aggregation function that ensures diminishing returns for repeated concepts, thus balancing local quality with global diversity. Finally, we implement an efficient data selection algorithm with two controllable strategies: maximizing global information gain for **General Sampling**, and incorporating KL-divergence for **Knowledge-Aligned Sampling** to approximate target domain distributions.

To validate the effectiveness of TAGS, we conducted extensive experiments using the Qwen3 (Yang et al., 2025) model family at varying scales, across a diverse set of knowledge-intensive (Clark et al., 2018; Suzgun et al., 2023; Hendrycks et al., 2020; Chen, 2021; Cobbe et al., 2021; Zhou et al., 2023b) and human-preference benchmarks (Zheng et al., 2023b; Dong et al., 2024; Lin et al., 2024). As shown in Figure 1, TAGS consistently achieves state-of-the-art performance, significantly outperforming strong baselines in both general capabilities and reasoning tasks. Remarkably, using only **5%** of the data, TAGS outperforms the full-dataset fine-tuning baseline by **2.23%** on knowledge-intensive benchmarks and achieves a substantial **12.34%** gain in human-preference evaluations. Moreover, TAGS demonstrates precise controllability through knowledge-aligned sampling, with an average performance improvement of **4.24%** on target domains compared to the general selection strategy, highlighting the efficacy of domain-specific adaptation.

In summary, our contributions are as follows:

- We propose TAGS, a unified framework that leverages hierarchical structures built from fine-grained tags to enable global quality, diversity, and target alignment in data selection.
- We introduce two sampling strategies, General Sampling and Knowledge-Aligned Sampling, offering flexible and controllable data selection to meet diverse requirements.
- Extensive evaluations across model scales demonstrate state-of-the-art data efficiency.

2 Related Work

Instruction Tuning and Data Efficiency. Instruction tuning is widely acknowledged as an important approach for aligning LLMs with human intent (Ouyang et al., 2022; Wei et al., 2021; Sanh et al., 2021). Recent studies (Zhou et al., 2023a; Ge et al., 2024; Chen et al., 2025) further demonstrate that during the Supervised Fine-Tuning (SFT) stage, utilizing a meticulously curated subset of high-quality instructions can yield performance superior

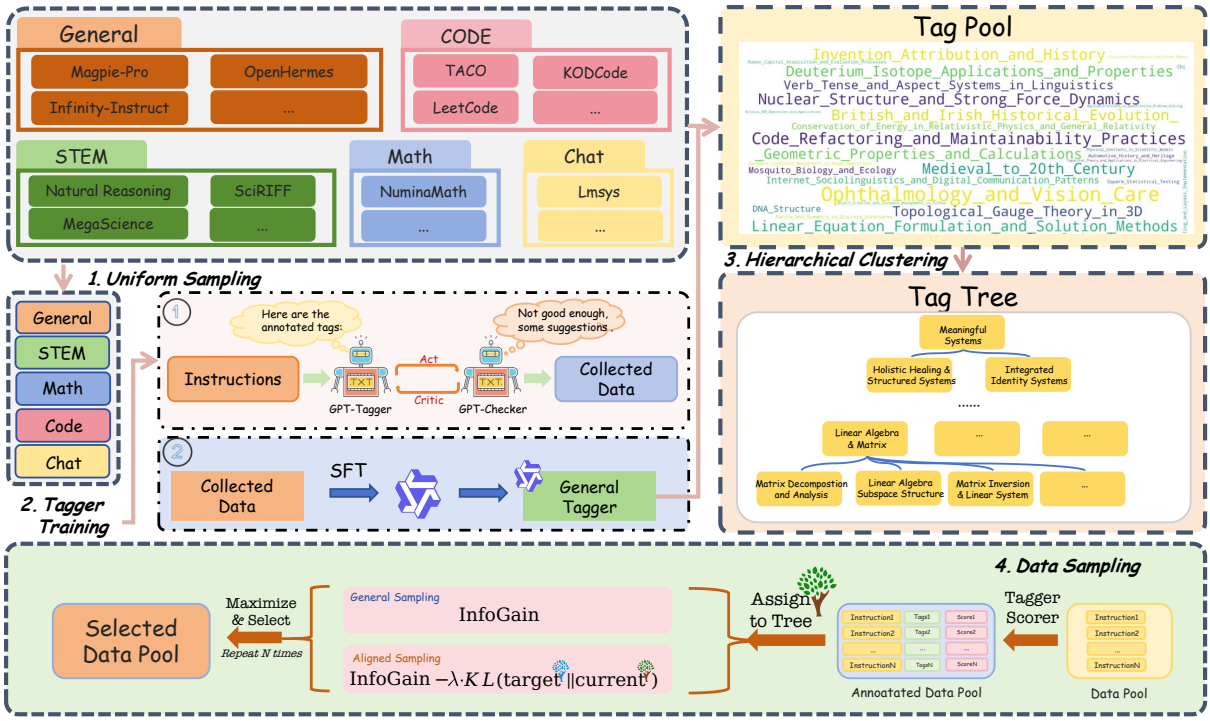

Figure 2: Overview of the TAGS framework: (1) **Uniform Sampling** from diverse data sources; (2) **Tagger Training**, which employs an LLM-based Act-Critic loop to synthesize fine-grained tags and fine-tune a General Tagger; (3) **Hierarchical Clustering** structures the generated tag pool into a global Tag Tree; and (4) **Data Sampling** maps instances to tree nodes and executes controllable selection based on Information Gain and KL-divergence.

to models trained on significantly larger datasets. These findings underscore the importance of strategic data selection, where developing automated frameworks to balance both data quality and diversity has emerged as a vital research topic.

Automated Data Selection. Existing automated data selection methods (Chen et al., 2023; Lu et al., 2023; Ge et al., 2024; Liu et al., 2024b; Chen et al., 2025) aim to identify training subsets that maximize efficiency and performance. Broadly, they can be categorized into metric-based filtering, diversity-driven selection, and comprehensive strategies. (1) ***Metric-based filtering*** relies on scalar criteria to assess data quality or difficulty. IFD (Li et al., 2024a) estimates instruction difficulty via the discrepancy between predicted and self-generated responses, while ALPAGASUS (Chen et al., 2023) employs LLM-based judges and SelectIT (Liu et al., 2024b) filters samples based on intrinsic uncertainty. Although effective at denoising, these methods evaluate instances independently, ignoring the global diversity. (2) ***Diversity-driven selection*** emphasizes coverage of the semantic space. ZIP (Yin et al., 2024) promotes diversity through a compression-based metric, whereas QDIT (Bukharin et al., 2024) selects representative samples that maximize current subset diversity. (3) ***Comprehensive strategies*** seek to jointly balance quality and diversity during the selection. CaR (Ge et al., 2024) performs clustering followed by intra-cluster quality selection, while DEITA (Liu et al.) greedily selects high-quality samples with semantic redundancy penalties. *#InsTag* (Lu et al., 2023) introduces LLM-generated coarse-grained tags and selects data based on tag coverage and complexity, MIG (Chen et al., 2025) further extends this by modeling semantic relations as a label graph and maximizing information gain. Despite these advances, existing methods overlook fine-grained knowledge and their explicit hierarchical dependencies, limiting the capability for precise data metric and knowledge-aligned selection. To address this issue, we introduce TAGS, which leverages fine-grained tags and tree structure to achieve global quality, diversity, and targeted sampling.

3 Method

As shown in Figure 2, we propose TAGS, a tree-aware data selection framework that enables controllable and knowledge-aligned sampling. TAGS

fine-tunes an LLM to extract fine-grained tags (Sec. 3.2), organizes them into a Tag Tree (Sec. 3.3), and performs data selection via Information Gain and KL-divergence (Sec. 3.4).

3.1 Problem Formulation & Notation

Task. Given a dataset D_P, a budget N and a large language model M, the task is to select a subset $D_S \subset D_P$ of size N such that the performance of the LLM fine-tuned on D_S, denoted by $F(M, D_S)$, is maximized. This can be formulated as:

$$D_S = \underset{D \subset D_P, |D|=N}{\operatorname{argmax}} F(M, D). \quad (1)$$

Data. Each data instance d_i consists of a query-response pair (q_i, r_i), along with associated attributes (T_i, Q_i, C_i), where q_i denotes the query and r_i denotes the response. T_i represents extracted fine-grained tags, while Q_i and C_i denote the quality score and complexity score, respectively, which are separately evaluated by the two scorers introduced in DEITA (Liu et al.).

3.2 Fine-grained Tagger Training

When dealing with massive instructions, existing approaches (Ge et al., 2024; Bukharin et al., 2024) typically rely on embeddings and clustering to evaluate data distributions and diversity. However, these methods lack interpretability, making it difficult to distinguish whether clusters are driven by shared linguistic style, core knowledge, or other latent factors. In contrast, explicit semantic tags map unstructured text into lower-noise tag spaces that better capture the underlying knowledge, enabling data to be measured in a more compact and controllable semantic space. Nevertheless, as illustrated in Figure 3, existing taggers (e.g., *#InsTag*) primarily focus on coarse-grained intents and struggle to model the specific entities, contexts, and constraints embedded in complex instructions. This limitation motivates the development of a more precise, fine-grained tagger, which enables reliable extraction of fine-grained knowledge in TAGS.

We construct a high-quality training corpus for fine-grained tagging, consisting of 50k samples spanning core domains, augmented with an additional 3k manually curated complicated instructions to enhance the tagger's capability in handling complex queries (see Table 7 for a detailed breakdown). Inspired by the Self-Refine mechanism (Madaan et al., 2023), we implement an Act-Critic annotation pipeline powered by `GPT-o4-mini`. In

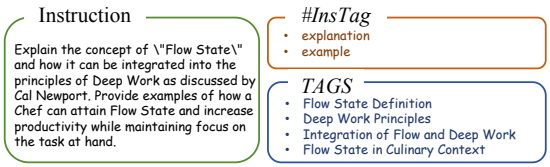

Figure 3: Comparison of *#InsTag* and TAGS tagger.

this pipeline, a *GPT-Tagger* generates initial tags, while a *GPT-Checker* reviews them and provides refinement suggestions. To balance quality and efficiency, we limit the annotation process to a maximum of three iterations. Finally, we perform SFT on `Qwen3-8B-Instruct` (Yang et al., 2025) using this synthetic data, yielding a general tagger that balances precise query understanding with inference efficiency for large-scale data processing.

3.3 Hierarchical Tag Tree Construction

Despite the semantic denoising achieved by TAGS tagger, the resulting tag pool still exhibits significant fragmentation. Similar concepts may be assigned different tags due to variations in expression (e.g., *"Value-to-Index Mapping"* vs. *"Index-based count computation"*), while flat tag representations struggle to reflect the hierarchical structure of complex instructions (e.g., *"Matrix Decomposition"* under *"Linear Algebra"*). As a result, isolated tags limit the global assessment of data coverage, motivating the need for a structured representation of tag semantics. Inspired by Clio (Tamkin et al., 2024), we construct a Tag Tree for downstream analysis and sampling. To balance efficiency with precision, we adopt an iterative bottom-up framework. At each iteration, we first aggregate the current set of nodes into clusters using K-Means based on semantic embeddings[1]. To resolve ambiguities inherent in embedding-based clustering, we employ an LLM[2] to refine these clusters through a structured four-step pipeline:

- **Summarization:** Abstracting a representative topic candidate for each cluster;
- **Deduplication:** Removing synonymous topics to ensure uniqueness;
- **Reassign:** Reviewing member tags and reassigning misclassified items to their appropriate topic to rectify vector-based errors;
- **Renaming:** Refining the topic name to accurately reflect the corrected cluster composition.

The generated topics serve as inputs for the subsequent iteration. This cycle repeats recursively

[1] We utilize `Qwen3-Embedding-0.6B` as encoder.
[2] `Qwen3-30B-A3B-2507`

until a specified tree depth is reached, ultimately constructing the global Tag Tree.

3.4 Tree-aware Data Sampling

Tree Topology & Data Anchoring. We utilize the semantic tag tree $\mathcal{T} = (V, E)$ constructed in Sec. 3.3, where V denotes the set of nodes and E represents the directed edges encoding hierarchical relations. To facilitate data anchoring, we transform this topology into an ancestry matrix $\mathbf{M} \in \{0,1\}^{|V| \times |V_{\text{leaf}}|}$. This matrix maps the relationships between all nodes V and the subset of fine-grained leaf nodes $V_{\text{leaf}} \subset V$:

$$\mathbf{M}_{ij} = \mathbb{I}(i \text{ is an ancestor of } j) \vee \mathbb{I}(i = j), \quad (2)$$

where $\mathbb{I}(\cdot)$ is the indicator function. This explicitly encodes the logical implication: if a leaf concept is active, all its ancestral categories are logically implicated. We further anchor the training instances to this topology to obtain a structure-aware representation. Each sample d_k is assigned to leaf nodes by matching its fine-grained tags T_k to the leaf-level concepts based on maximum semantic similarity, resulting in a binary leaf activation vector $\mathbf{h}_k^{(\text{leaf})} \in \{0,1\}^{|V_{\text{leaf}}|}$. To reconstruct the tree-level semantic profile, we propagate leaf-level signals upward via the ancestry matrix:

$$\mathbf{h}_k^{(\text{tree})} = \mathbf{M} \cdot \mathbf{h}_k^{(\text{leaf})}. \quad (3)$$

Tree-based Information Modeling. Given the hierarchical profile $\mathbf{h}_k^{(\text{tree})}$ of each data point d_k, we first quantify its utility by combining its quality score Q_k and complexity score C_k. We define a composite scalar weight s_k as:

$$s_k = \alpha \cdot Q_k + (1 - \alpha) \cdot C_k, \quad (4)$$

where $\alpha \in [0, 1]$ is a hyperparameter balancing the two metrics. To further model the effective contribution of d_k to the semantic space, we scale its profile by s_k, yielding the raw information vector:

$$\mathbf{e}_k = s_k \cdot \mathbf{h}_k^{(\text{tree})}. \quad (5)$$

Since concepts are organized along a hierarchy, treating them as isolated features fails to capture their semantic dependencies. To capture this semantic overlap, we allow information to propagate through the tag tree structure. Following Chen et al. (2025), we define the propagation coefficient $\mathbf{A}_{pq}$ from node p to node q based on the tree topology:

$$\mathbf{A}_{pq} = \frac{a_{pq}}{1 + \sum_{j \neq p} a_{pj}}, \quad (6)$$

where $a_{pq} = \mathbb{I}((p, q) \in E)$ indicates the binary structural connection strength. This propagation mechanism ensures that effective coverage is measured across related conceptual branches rather than discrete nodes.

Based on this propagation matrix $\mathbf{A}$, the effective information vector of d_k is transformed into $\mathbf{A}\mathbf{e}_k$. Finally, we quantify the utility of the selected subset D by aggregating the propagated information:

$$\mathcal{I}(D) = \sum_{j \in V} \Phi(\sum_{k \in D} A\mathbf{e}_k) \quad (7)$$

where $\Phi(x) = x^\gamma$ (with $\gamma \in (0, 1)$) is a strictly increasing, concave function. Crucially, Φ being increasing prioritizes samples with high composite scores, while its concavity imposes diminishing returns to mitigate redundancy, thereby balancing individual quality and global diversity.

Tree-aware Dual-mode Sampling. To curate a high-quality subset, we adopt a greedy strategy based on Information Gain. However, re-evaluating the full objective for every candidate is computationally expensive. To ensure efficiency, we approximate the marginal information gain using a first-order Taylor expansion:

$$\begin{aligned} d_k &= \operatorname*{argmax}_{d \in D_P^k} \left(\mathcal{I}(D_S^k \cup \{d\}) - \mathcal{I}(D_S^k) \right) \\ &\approx \operatorname*{argmax}_{d \in D_P^k} \left(\nabla_{\mathbf{v}} \mathcal{I}(\mathbf{v}_k)^\top (\mathbf{A}\mathbf{e}_d) \right) \\ &= \operatorname*{argmax}_{d \in D_P^k} (\Phi'(\mathbf{A} \sum_{i \in D_S^k} \mathbf{e}_i)^\top \mathbf{A}\mathbf{e}_d), \end{aligned} \quad (8)$$

where $\mathbf{v}_k = \mathbf{A} \sum_{i \in D_S^k} \mathbf{e}_i$ denotes the accumulated semantic profile, while D_S^k and D_P^k represent the selected subset and the remaining candidate pool at iteration k, respectively.

To accommodate diverse application needs, we extend this framework with two distinct sampling modes: (1) *General Sampling*, which prioritizes global quality and diversity across the entire Tag Tree; and (2) *Aligned Sampling*, which enforces adaptation to specific domains. For the latter, we introduce a target distribution $Q \in \Delta^{|V_{\text{leaf}}|}$ defined over the leaf nodes. Since leaves represent atomic knowledge units, Q allows for fine-grained control over the desired dataset composition, whether specified manually for domain customization or derived from a reference benchmark. We unify these strategies into a single objective by incorporating a regularization term based on the KL-divergence. Defining the leaf-level distribution of the current

Algorithm 1 TAGS Sampling

Input: Data Pool D_P, Sample Budget N, Target Distribution Q
Output: The Sampled Dataset D_S
Initialize Empty D_S;
Initialize Propagation Matrix A;
while $|D_S| < N$ **do**
  $G \leftarrow \Phi'(A \sum_{k \in D_S} \mathbf{e}_k)^\top A$;
  $d_i \leftarrow \arg\max_{d \in D_P} G\mathbf{e}_d - \lambda \mathcal{D}_{\mathrm{KL}}(Q || P(D_S \cup \{d\}))$;
  $D_S \leftarrow D_S \cup \{d_i\}$;
  $D_P \leftarrow D_P \setminus \{d_i\}$;
end while
return D_S

subset as $P(D_S) \propto \sum_{d \in D_S} \mathbf{h}_d^{(\text{leaf})}$, we select the sample maximizing the joint score at iteration k:

$$d_k = \underset{d \in D_P^k}{\operatorname{argmax}} (\Delta\mathcal{I}(d) - \lambda \cdot \mathcal{D}_{\mathrm{KL}}(Q \parallel P(D_S^k \cup \{d\}))), \quad (9)$$

where $\Delta\mathcal{I}(d)$ denotes the approximated gain from Eq. 8. Setting $\lambda = 0$ reduces to general sampling, while $\lambda > 0$ enforces distribution alignment with the target Q. The procedure is outlined in Algorithm 1. More details are shown in Appendix A.2.

4 Experiments

We validate the TAGS framework through extensive experiments across multiple benchmarks, following established settings (Lu et al., 2023; Chen et al., 2025). Our evaluation follows a bottom-up design, addressed by three key research questions:

- **RQ1:** Can the TAGS Tagger accurately capture instruction intent and knowledge semantics?
- **RQ2:** Does the hierarchical Tag Tree exhibit a coherent and interpretable structure?
- **RQ3:** Do the TAGS sampling strategies effectively improve SFT performance?

4.1 Tasks and Datasets.

Tagger Evaluation. To evaluate the proposed tagger, we introduce TaggerEval, a human-aligned evaluation framework (Figure 4). We construct a benchmark of 100 high-quality samples spanning diverse domains, and use an powerful LLM[3] to assess the generated tags along three binary dimensions, following (Lee et al., 2025):

- **Coverage** evaluates whether each golden-truth concept is included in the generated tags.
- **Precision** assesses whether each generated tag correctly reflects the input instruction.
- **Fine-grained** measures whether each generated tag represents a fine-grained, atomic concept.

[3]We utilize `GPT-4.1` as our judge model.

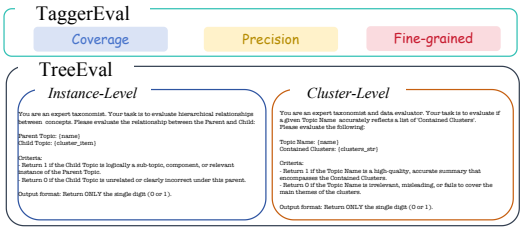

Figure 4: The TaggerEval and TreeEval Frameworks.

We report the average score on these metrics as the tagger performance. To ensure evaluation reliability, we also report Cohen's kappa (Sim and Wright, 2005; Zlabinger et al., 2020) between the LLM judge and human ratings.

Tree Evaluation. To ensure the semantic coherence of our taxonomy, we introduce *TreeEval*, as shown in Figure 4. We construct a benchmark consisting of 200 randomly sampled tag-parent pairs and 50 parent clusters to assess structural quality from two perspectives:

- **Instance-Level** verifies whether the tags are logical sub-topics of their assigned clusters.
- **Cluster-Level** checks whether the cluster name captures the shared semantics of its children.

Similar to TaggerEval, we employ LLM to score these samples based on binary correctness.

Data Sampling Evaluation. We conducted experiments on the Tulu3 dataset (Lambert et al., 2024) using Qwen3 models (Yang et al., 2025) of various sizes. Model performance was comprehensively evaluated using knowledge-intensive and human-preference benchmarks. Evaluations were performed with OpenCompass (Contributors, 2023), and average results are reported as normalized scores in percentage.

- **Knowledge-intensive Benchmarks:** We examine the model's factual knowledge, reasoning, coding, math, and instruction-following abilities across ARC (Clark et al., 2018), BBH (Suzgun et al., 2023), MMLU (Hendrycks et al., 2020), HumanEval (Chen, 2021), GSM8k (Cobbe et al., 2021), and IFEval (Zhou et al., 2023b).
- **Human-preference Benchmarks:** We assess model's open-ended performance on AlpacaEvalv2 (Dubois et al., 2024), MTBench (Zheng et al., 2023b), and WildBench (Lin et al., 2024).
- **Baselines:** We compare our methods against strong data selection baseline including: random selection (Xia et al., 2024), IFD (Li et al., 2024a), *#InsTag* (Lu et al., 2023), DEITA (Liu et al.), CaR (Ge et al., 2024), QDIT (Bukharin et al., 2024) and MIG (Chen et al., 2025).

Base Model	Method	DataSize	ARC	BBH	GSM	HE	MMLU	IFEval	Avg_{obj}	AE	MT	Wild	Avg_{sub}	Avg
Qwen3-8B	Pool	939K	88.47	69.71	88.86	77.44	75.27	**69.87**	78.27	9.69	7.34	44.52	42.54	60.40
	Random	50K	80.68	72.29	90.75	82.93	78.60	64.14	78.23	9.44	7.51	41.74	42.09	60.16
	IFD	50K	**90.85**	**77.65**	91.36	81.71	78.79	57.12	79.58	15.16	7.62	42.32	44.56	62.07
	#InsTag	50K	85.42	70.78	90.75	80.49	78.81	60.63	77.81	10.68	7.67	42.16	43.18	60.50
	DEITA	50K	60.00	71.59	**92.65**	**85.37**	78.61	60.44	74.78	14.04	7.62	44.16	44.80	59.79
	CaR	50K	79.66	73.40	91.36	82.32	78.91	56.19	76.97	13.29	7.49	42.52	43.57	60.27
	QDIT	50K	76.95	77.40	90.67	81.71	78.82	58.23	77.30	14.04	7.41	40.90	43.01	60.16
	MIG	50K	76.61	74.49	91.51	79.88	**79.04**	65.80	77.89	16.65	7.59	43.49	45.35	61.46
	Ours	50K	86.78	74.22	91.51	82.32	78.75	66.91	**80.08**	**20.12**	**7.85**	**44.76**	**47.79**	**63.93**
Qwen3-4B	Random	50K	85.42	67.75	89.23	78.05	75.29	61.18	76.15	9.07	7.25	38.16	39.91	58.03
	IFD	50K	91.53	72.40	78.17	76.22	75.23	57.86	75.23	**15.65**	7.26	40.15	42.80	59.02
	#InsTag	50K	84.07	66.88	86.96	78.66	75.35	63.59	75.92	9.57	7.38	38.26	40.54	58.23
	DEITA	50K	76.95	67.83	81.35	78.66	**75.56**	63.22	73.93	8.20	7.26	38.14	39.64	56.78
	CaR	50K	91.86	68.51	79.30	77.44	75.26	54.90	74.54	13.54	7.33	38.80	41.88	58.21
	QDIT	50K	91.19	72.90	85.06	76.83	75.16	58.78	76.65	12.80	7.24	38.86	41.35	59.00
	MIG	50K	91.53	72.10	70.43	**81.10**	75.47	65.80	76.07	14.04	**7.64**	40.66	**43.70**	59.88
	Ours	50K	**92.54**	70.78	**89.69**	76.83	75.33	**66.73**	**78.65**	14.78	7.34	**40.98**	43.05	**60.85**
Qwen3-1.7B	Random	50K	78.64	51.68	77.26	65.85	64.33	42.14	63.32	4.84	6.64	25.02	32.09	47.70
	IFD	50K	78.31	53.92	54.97	67.07	64.20	35.49	58.99	6.21	6.60	25.46	32.56	45.77
	#InsTag	50K	78.31	50.88	77.26	66.46	64.27	43.44	63.44	5.09	6.51	26.17	32.12	47.78
	DEITA	50K	71.53	51.42	62.55	65.85	64.29	42.33	59.66	5.22	6.33	25.40	31.31	45.48
	CaR	50K	71.86	52.70	73.77	61.59	**64.36**	38.45	60.45	6.46	6.43	26.42	32.40	46.42
	QDIT	50K	76.95	**54.62**	58.07	67.07	64.13	37.71	59.76	7.08	6.26	24.49	31.39	45.58
	MIG	50K	80.34	53.53	78.77	66.46	64.00	44.36	64.58	9.69	6.54	27.92	34.34	49.46
	Ours	50K	**82.03**	54.31	**80.14**	**67.68**	63.31	**48.43**	**65.98**	**10.93**	**6.76**	**29.83**	**36.12**	**51.05**

Table 1: Comparison of data selection methods on Tulu3 pool. Avg is mean of Avg_{obj} and Avg_{sub}. TAGS achieves best performance across all base models.

Method	Coverage	Precision	Fine-grained	Avg.
#InsTagger	0.4527	0.2514	0.1286	0.2776
Ours(w/o Act-Critic)	0.7387	0.7815	0.6328	0.7177
Ours(w Act-Critic)	**0.7623**	**0.817**	**0.7793**	**0.7862**
kappa (vs human)	0.7069	0.7826	0.7218	-

Table 2: Tag Performance Between Different Taggers.

Method	Cluster-Level	Instance-Level
Hierarchical Clustering	98.00%	96.00%

Table 3: Hierarchical Clustering Performance.

4.2 Main Results & Analysis

4.2.1 Analysis of Tagger Performance

Q1: How does the proposed TAGS Tagger perform compare with baselines?

As shown in Table 2, the TAGS Tagger consistently outperforms *#InsTagger* across all metrics, achieving a **+183.21%** relative improvement in the average score and demonstrating substantial gains in coverage, precision, and granularity.

Q2: What is the impact of Act-Critic pipeline?

Ablation studies further confirm the necessity of the Act-Critic design. Compared to the direct prompting baseline (*w/o Act-Critic*), employing the Act-Critic pipeline (*w/ Act-Critic*) yields consistent gains across all dimensions (average gain of **+9.54%**). This validates that the Act-Critic mechanism effectively enhance data synthesis performance, as validated in (Madaan et al., 2023).

4.2.2 Analysis of Hierarchical Clustering

Q3: Is the constructed Tag Tree reliable?

Results in Table 3 demonstrate the high reliability of our taxonomy. The constructed hierarchy attains strong validation scores, with 98.00% accuracy at Cluster level and 96.00% at Instance level. These results indicate that the parent–child relations are both logically sound and semantically coherent.

4.2.3 Analysis of Data Sampling

Q4: How does the General Sampling perform regarding overall effectiveness and data efficiency?

As illustrated in Table 1, TAGS consistently outperforms all baselines across objective and subjective metrics. Specifically, on the Qwen3-8B backbone, TAGS achieves the highest average score, surpassing the strongest baseline by **+2.99%** and the Random selection baseline by **+6.27%**. Notably, while maintaining competitive capability on knowledge benchmarks (**+0.63%**), TAGS delivers a substantial breakthrough in human-preference alignment (**+5.38%**) compared to the strongest baseline. This suggests that the general-purpose sampled, diverse data significantly improves the model's performance on open-ended instructions. Crucially, TAGS demonstrates a profound *"less is more"* phenomenon: by utilizing only **5%** of the data pool, it outperforms the model trained on the full dataset by **+5.84%**. This confirms that the tree-aware sam-

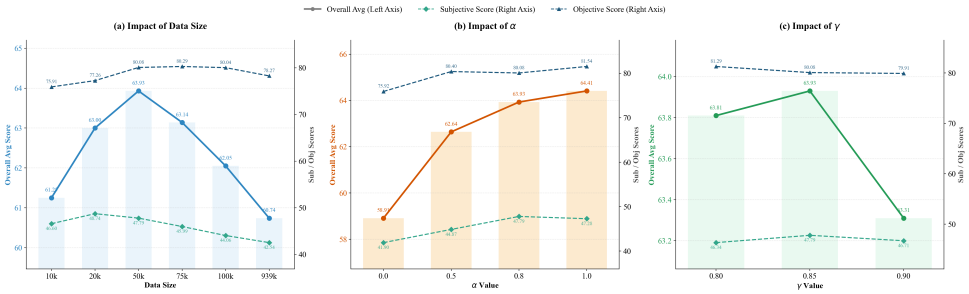

Figure 5: Ablation studies on training data size, hyperparameter α of complicated score s, and γ value of $\Phi(x)$.

Method	Data Size	ARC	GSM8K	GPQA-Diamond
TAGS	10K	91.19	85.67	30.30
TAGS-*Align*	10K	93.90	89.69	31.83
Improvement	-	**+2.97%**	**+4.69%**	**+5.06%**

Table 4: Performance of aligned sampling strategy.

Method	DataSize	Avg_{obj}	Avg_{sub}	Avg
MIG	50k	77.89	45.35	61.46
TAGS(w/o Tree)	50k	80.05	47.00	63.52
TAGS	50k	**80.08**	**47.79**	**63.93**

Table 5: Ablation study of TAGS.

pling effectively identifies high-quality, diversity samples, eliminating redundancy and noise.

Q5: Is the proposed strategy scalable across different model sizes?

TAGS demonstrates strong scalability, consistently ranking first across 1.7B, 4B, and 8B models, as shown in Table 1. Notably, it yields the largest gains on the 1.7B model, improving the average score by **+3.21%** over the best baseline, indicating its effectiveness independent of model scale.

Q6: Is the Aligned Sampling strategy effective across diverse benchmarks?

To verify adaptability, we map specific benchmarks (ARC (Clark et al., 2018), GSM8K (Cobbe et al., 2021), and GPQA-Diamond (Rein et al., 2024)) to Tag Tree and perform aligned sampling (set $\lambda = 5$) with a budget of 10k samples. As shown in Table 4, this strategy yields comprehensive improvements, boosting average performance by **+4.24%**. These results indicate that TAGS effectively aligns sampled data with diverse, target domains, enabling fine-grained steering of model capabilities toward specific domain constraints or user requirements.

4.3 Ablation Study

Q7: What are the contributions of the fine-grained tagger and the tree structure?

To decouple the contributions of our key components, we compare three configurations as detailed in Table 5: (1) **MIG**, which serves as the baseline using coarse-grained tags without tag tree; (2) **TAGS (w/o Tree)**, which employs our fine-grained tagger but relies on flat label graph; and (3) **TAGS**, which integrates both the fine-grained tagger and the hierarchical tag tree. The results reveal two key insights. First, the transition from coarse to fine-grained tagging (**MIG** vs. **TAGS w/o Tree**) yields a substantial performance leap of **+3.35%**, underscoring the critical importance of atomic data representation. Second, incorporating the hierarchical structure (**TAGS**) provides further optimization, particularly in subjective alignment, where it boosts performance by **+1.68%**, raising the final average score by **+0.64%**. These findings suggest that fine-grained tags enable precise data assessment, while the tag tree ensures diversity and balance.

Q8: How do key hyperparameters and data scale affect the effectiveness of TAGS?

Figure 5 shows that TAGS performance peaks at 50K samples, indicating that selection is more critical than sheer scaling. While setting $\alpha = 1.0$ improves average scores, incorporating complexity ($\alpha = 0.8$) yields the best results on challenging reasoning tasks (BBH, MMLU) and open-ended benchmarks (MT-Bench, WildBench), suggesting that a 20% complexity weight helps select harder, more informative samples. Additionally, setting $\gamma = 0.85$ balances noise filtering with sample retention. Detailed results are shown in the Table 9.

5 Conclusion

In this paper, we present TAGS, a framework that integrates fine-grained tags into a hierarchical tree for controllable data selection. By organizing atomic tags into a global taxonomy, TAGS enables sampling that balances data quality and diversity. Extensive experiments show that TAGS consistently outperforms strong baselines across diverse benchmarks, achieving superior performance with only 5% of the original data, demonstrating its efficacy for efficient instruction tuning.

Limitations

1. **Tagger Data Scale:** To balance performance with proprietary API costs, our Tagger was fine-tuned on a compact dataset of 53K samples. While this effectively validates data efficiency, it precludes a comprehensive analysis of the Tagger's scaling behavior on larger synthetic datasets.

2. **Model & Task Scope:** Due to computational constraints, our evaluation focuses on the `Qwen3-1.7B, 4B, 8B` series for SFT tasks. Verification on larger-scale models (e.g., `Qwen3-30B`) and other paradigms like Reinforcement Learning remains for future exploration.

References

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

Alexander Bukharin, Shiyang Li, Zhengyang Wang, Jingfeng Yang, Bing Yin, Xian Li, Chao Zhang, Tuo Zhao, and Haoming Jiang. 2024. Data diversity matters for robust instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3411–3425.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and 1 others. 2023. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*.

Mark Chen. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yicheng Chen, Yining Li, Kai Hu, Ma Zerun, HaochenYe HaochenYe, and Kai Chen. 2025. Mig: Automatic data selection for instruction tuning by maximizing information gain in semantic space. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9902–9915.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 177–198.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. 2024. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*.

Yuan Ge, Yilun Liu, Chi Hu, Weibin Meng, Shimin Tao, Xiaofeng Zhao, Mahong Xia, Zhang Li, Boxing Chen, Hao Yang, and 1 others. 2024. Clustering and ranking: Diversity-preserved instruction selection through expert-aligned quality estimation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 464–478.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, and 1 others. 2025. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Yukyung Lee, Joonghoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. 2025. Checkeval: A reliable llm-as-a-judge framework for evaluating text generation using checklists. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15782–15809.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-1.5](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).

Jijie Li, Li Du, Hanyu Zhao, Bo-wen Zhang, Liangdong Wang, Boyan Gao, Guang Liu, and Yonghua Lin. 2025. Infinity instruct: Scaling instruction selection and synthesis to enhance language models. *arXiv preprint arXiv:2506.11116*.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024a. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7602–7635.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. 2023. Taco: Topics in algorithmic code generation dataset. *arXiv preprint arXiv:2312.14852*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024b. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. 2024. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. *arXiv preprint arXiv:2406.04770*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liangxin Liu, Xuebo Liu, Derek F Wong, Dongfang Li, Ziyi Wang, Baotian Hu, and Min Zhang. 2024b. Selectit: Selective instruction tuning for llms via uncertainty-aware self-reflection. *Advances in Neural Information Processing Systems*, 37:97800–97825.

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 *The Twelfth International Conference on Learning Representations*.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. # instag: Instruction tagging for analyzing supervised fine-tuning of large language models. *arXiv preprint arXiv:2308.07074*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, and 1 others. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Julius Sim and Chris C Wright. 2005. The kappa statistic in reliability studies: use, interpretation, and sample size requirements. *Physical therapy*, 85(3):257–268.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Alex Tamkin, Miles McCain, Kunal Handa, Esin Durmus, Liane Lovitt, Ankur Rathi, Saffron Huang, Alfred Mountfield, Jerry Hong, Stuart Ritchie, and 1 others. 2024. Clio: Privacy-preserving insights into real-world ai use. *arXiv preprint arXiv:2412.13678*.

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, and 76 others. 2025. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *Preprint*, arXiv:2502.14739.

Teknium. 2023. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants.

Hemish Veeraboina. 2023. Aime problem set 1983-2024.

David Wadden, Kejian Shi, Jacob Morrison, Alan Li, Aakanksha Naik, Shruti Singh, Nitzan Barzilay, Kyle Lo, Tom Hope, Luca Soldaini, Shannon Zejiang Shen, Doug Downey, Hannaneh Hajishirzi, and Arman Cohan. 2025. SciRIFF: A resource to enhance language model instruction-following over scientific literature. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*,

pages 6083–6120, Suzhou, China. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Tingyu Xia, Bowen Yu, Kai Dang, An Yang, Yuan Wu, Yuan Tian, Yi Chang, and Junyang Lin. 2024. Rethinking data selection at scale: Random selection is almost all you need. *arXiv preprint arXiv:2410.09335*.

Yunhui Xia, Wei Shen, Yan Wang, Jason Klein Liu, Huifeng Sun, Siyue Wu, Jian Hu, and Xiaolong Xu. 2025. Leetcodedataset: A temporal dataset for robust evaluation and efficient training of code llms. *arXiv preprint arXiv:2504.14655*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *arXiv preprint arXiv:2406.08464*.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. 2025. KodCode: A diverse, challenging, and verifiable synthetic dataset for coding. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6980–7008, Vienna, Austria. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Mingjia Yin, Chuhan Wu, Yufei Wang, Hao Wang, Wei Guo, Yasheng Wang, Yong Liu, Ruiming Tang, Defu Lian, and Enhong Chen. 2024. Entropy law: The story behind data compression and llm performance. *arXiv preprint arXiv:2407.06645*.

Weizhe Yuan, Jane Yu, Song Jiang, Karthik Padthe, Yang Li, Dong Wang, Ilia Kulikov, Kyunghyun Cho, Yuandong Tian, Jason E Weston, and Xian Li. 2025. Naturalreasoning: Reasoning in the wild with 2.8m challenging questions. *Preprint*, arXiv:2502.13124.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. Swift:a scalable lightweight infrastructure for fine-tuning. *Preprint*, arXiv:2408.05517.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric. P Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. 2023a. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *Preprint*, arXiv:2309.11998.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023b. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, and 1 others. 2023a. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023b. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Markus Zlabinger, Marta Sabou, Sebastian Hofstätter, Mete Sertkan, and Allan Hanbury. 2020. Dexa: Supporting non-expert annotators with dynamic examples from experts. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 2109–2112, New York, NY, USA. Association for Computing Machinery.

A More TAGS Implementation Details

A.1 Baseline Settings

We adhere to the comparison settings established by MIG (Chen et al., 2025) and utilize 50k sampled instances from each MIG-released baseline method[4] to conduct a consistent and fair SFT evaluation. This setup ensures that all methods are assessed under uniform data conditions, enabling a direct comparison of their intrinsic capabilities.

A.2 Implementation Details

A.2.1 Tagger Training.

We first collect 53k Supervised Fine-Tuning samples synthesized by `GPT-o4-mini` using an Actor–Critic pipeline to train the TAGS Tagger, and then further fine-tune it on `Qwen3-8B-Instruct`[5] . The distribution of the training data is summarized in Table 7. For non-complex queries, we apply uniform sampling over the entire pool. For complex queries, we perform uniform sampling from several inherently high-difficulty datasets spanning the Math, Code, and STEM domains:

- **Math**: AIME and ArenaHard-Math.
- **Code**: ArenaHard-Coding.
- **STEM**: SuperGPQA.

[4] https://huggingface.co/collections/xsample/mig-datasets

[5] https://huggingface.co/Qwen/Qwen3-8B

A.2.2 Tree Constructing.

We adopt the bottom-up hierarchical clustering algorithm described in Clio (Tamkin et al., 2024). Starting from 7M high-quality open-source samples (Teknium, 2023; Xu et al., 2024; LI et al., 2024; Xu et al., 2025; Li et al., 2025), we first apply the TAGS Tagger to extract fine-grained tags. To ensure computational efficiency at scale, we use all-mpnet-base-v2[6] as the encoder to perform an initial coarse-grained clustering, reducing the data into 10,000 preliminary clusters. We then employ `Qwen3-30B-A3B-2507`[7] to summarize, deduplicate, reassign, and rename cluster labels, producing cleaner and more semantically coherent tag groups. After this refinement step, we further apply Qwen3-Embedding-0.6B[8] to conduct subsequent clustering, which provides a better balance between clustering quality and computational cost. This multi-stage pipeline ultimately yields a 10-level hierarchical tree, offering a structured and interpretable organization of the underlying data space.

A.2.3 TAGS Sampling & Training.

We sample 50k training instances from the full 939k Tulu3 dataset (Lambert et al., 2024) and use 8 GPUs to accelerate both the sampling process and the subsequent SFT training. In the initial stage of sampling, we initialize G as a zero vector. For training stage, we set the batch size to 128, learning rate to 5e-6, warmup ratio to 0.03, and the maximum length to 8192. The implementation is conducted using the Swift framework (Zhao et al., 2024). As shown in Table 6, General Sampling is extremely efficient, occupying less than 0.2% of the full training time while selecting a high-quality 50k subset. Even for Aligned Sampling, which involves complex KL-divergence calculations, the computational overhead remains minimal, ensuring practical usability. Additionally, our sampling strategy compresses the total training time to 5%, yet achieves superior performance, fully demonstrating its efficiency and practical value.

Stage	Time(min)
General Sampling	3.13
Aligned Sampling	35.51
SFT(50k Data)	90.85
SFT(939k Data)	1810.53

Table 6: Computational Overhead of Sampling and Training Phases.

B Detailed Experimental Results

We present detailed experimental results across varying data scales (Table 8), hyperparameter α (Table 9), hyperparameter γ (Table 10), and ablation study of fine-grained Tagger and Tag Tree (Table 5).

C Prompt Template

C.1 Tagging Data Synthesis

We instruct an LLM to act as GPT-Tagger for initial tagging (Table 12), then act as a GPT-Checker for tag refinement (Table 13) to synthesize high-quality tagger training data.

C.2 TaggerEval

We prompt an LLM as a judge to evaluate the generated tags from coverage (Table 14), precision (Table 15), fine-grained (Table 16) dimensions.

C.3 TreeEval

We further prompt the LLM to evaluate the constructed Tag Tree from Instance-Level (Table 17) and Cluster-Level (Table 18) dimensions.

[6] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[7] https://huggingface.co/Qwen/Qwen3-30B-A3B-Thinking-2507

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

Domain	Quantity	Source
General	10K	Magpie-Pro (Xu et al., 2024), Infinity-Instruct (Li et al., 2025), OpenHermes (Teknium, 2023), ...
Math	10K	NuminaMath-1.5 (LI et al., 2024), DeepMath-103K (He et al., 2025), Aqua-Rat (Ling et al., 2017), ...
Code	10K	TACO (Li et al., 2023), KODCode (Xu et al., 2025), LeetCodeDataset (Xia et al., 2025), ...
STEM	10K	Natural-Reasoning (Yuan et al., 2025), MegaScience (Xu et al., 2024), SciRiFF (Wadden et al., 2025), ...
Chat	10K	lmsys-Chat (Zheng et al., 2023a)
Complex	3k	AIME (Veeraboina, 2023), ArenaHard(Math, Coding) (Li et al., 2024b), SuperGPQA (Team et al., 2025), ...

Table 7: Domain distribution and source datasets used in tagger training.

Method	DataSize	ARC	BBH	GSM	HE	MMLU	IFEval	Avg_{obj}	AE	MT	Wild	Avg_{sub}	Avg
TAGS	10k	**91.19**	70.15	85.67	69.51	78.49	60.44	75.91	18.63	7.74	43.77	46.60	61.25
	20k	85.42	70.16	86.88	79.27	76.74	65.06	77.26	**22.11**	**7.85**	**45.62**	**48.74**	63.00
	50k	86.78	**74.22**	91.51	**82.32**	78.75	66.91	80.08	20.12	**7.85**	44.76	47.79	**63.93**
	75k	86.78	74.00	91.51	**82.32**	78.74	68.39	**80.29**	17.02	7.68	44.16	45.99	63.14
	100k	84.07	73.27	**91.66**	81.10	**79.01**	**71.16**	80.04	15.78	7.38	42.58	44.06	62.05
ALL	939k	88.47	69.71	88.86	77.44	75.27	69.87	78.27	9.69	7.34	44.52	42.54	60.74

Table 8: Performance comparison of different data size on objective and subjective benchmarks.

Method	DataSize	ARC	BBH	GSM	HE	MMLU	IFEval	Avg_{obj}	AE	MT	Wild	Avg_{sub}	Avg
TAGS (α=0.0)	50K	80.34	73.82	78.24	84.15	78.89	60.07	75.92	10.93	7.28	41.96	41.90	58.91
TAGS (α=0.5)	50K	85.08	73.59	91.13	**84.76**	**78.89**	68.95	80.40	16.02	7.53	43.29	44.87	62.64
TAGS (α=0.8)	50K	86.78	**74.22**	91.51	82.32	78.75	66.91	80.08	20.12	**7.85**	**44.76**	**47.79**	63.93
TAGS (α=1.0)	50K	**87.80**	74.12	**92.19**	84.15	78.71	**72.27**	**81.54**	**20.75**	7.67	44.40	47.28	**64.41**

Table 9: Performance comparison of different hyperparameter α on objective and subjective benchmarks.

Method	DataSize	ARC	BBH	GSM	HE	MMLU	IFEval	Avg_{obj}	AE	MT	Wild	Avg_{sub}	Avg
TAGS ($\gamma = 0.80$)	50K	**89.83**	73.86	**92.42**	**82.93**	**79.02**	**69.69**	**81.29**	19.13	7.55	44.38	46.34	63.81
TAGS ($\gamma = 0.85$)	50K	86.78	74.22	91.51	82.32	78.75	66.91	80.08	**20.12**	**7.85**	44.76	**47.79**	**63.93**
TAGS ($\gamma = 0.90$)	50K	86.44	**75.02**	91.05	78.66	78.79	69.50	79.91	19.38	7.59	**44.86**	46.71	63.31

Table 10: Performance comparison of different hyperparameter γ on objective and subjective benchmarks.

Method	DataSize	ARC	BBH	GSM	HE	MMLU	IFEval	Avg_{obj}	AE	MT	Wild	Avg_{sub}	Avg
MIG	50K	76.61	**74.49**	91.51	79.88	**79.04**	65.80	77.89	16.65	7.59	43.49	45.35	61.46
TAGS(w/o Tree)	50K	85.08	74.05	**91.58**	**83.54**	78.98	**67.10**	80.05	**20.25**	7.58	**44.95**	47.00	63.52
TAGS	50K	**86.78**	**74.22**	91.51	82.32	78.75	66.91	**80.08**	20.12	**7.85**	44.76	**47.79**	**63.93**

Table 11: Detailed performance of ablation study on objective and subjective benchmarks.

Instruction template for GPT-Tagger

You are a general knowledge extraction expert. Given a problem and its corresponding solution (which can be code, natural language, or other forms), your task is to identify all the fine-grained and atomic-level knowledge components involved.

Improvement Guidance
- Previous Tag: {old_tags}
- Improvement Hint: {improvement_reason} - If "None": This is a cold start, generate tags normally.
- If not "None": This is feedback from the checker, refine your output accordingly.
- For tags without explicitly identified issues, please keep them as they are. Only adjust the parts where problems are indicated.

You must consider multiple dimensions, including but not limited to:

* **Problem Domain** (e.g., "Number Theory", "Greedy Algorithms", "Ancient Rome History"),
* **Problem Type** (e.g., "Calculation", "Proof", "Code Implementation", "Conceptual Explanation", "Dialogue Response"),
* **Knowledge / Skill Points** (e.g., "Greatest Common Divisor", "Prime Factorization", "Binary Search", "Close Reading"),
* **User Intent** (e.g., "Learning a concept", "Practical application", "Exam preparation"),

Each knowledge point must be:

* **Atomic**: the smallest meaningful concept or technique involved.
* **Precise**: no vague categories like "Math" or "Data Structures".
* **Unabbreviated**: e.g., use "Dynamic Programming" instead of "DP".
* **Non-overlapping**: do not include multiple tags that describe the same idea with slight variation; only one representative tag should be included.
* **Core-only**: include only the most essential knowledge points necessary to understand or implement the solution. Do not include secondary or peripheral techniques unless they are critical.
* **Specifically Naming**: Your tag should be specific, include the core concept and entities into the tag.
* **Focus** on the user's primary intent while ignoring irrelevant context.
* If the problem is in a multi-turn dialogue format, use the previous conversation history as contextual background, but focus primarily on the current query.
* **Maximally concise**: output no more than *5* knowledge points.

Output Format

Return your results as a list of dictionaries, where each dictionary includes

* ''tag'': the name of the knowledge point
* ''explanation'': a brief and accurate explanation of its role in the problem and solution

Example

Problem: Find the GCD of 84 and 60
Reference Answer: Let's find the **GCD (Greatest Common Divisor)** of 84 and 60. Step 1: Prime factorization* $84 = 2 \times 2 \times 3 \times 7 = 2^2 \times 3^1 \times 7^1$* $60 = 2 \times 2 \times 3 \times 5 = 2^2 \times 3^1 \times 5^1$ Step 2: Take the lowest powers of common primesprimes: 2 and 3* $2^{\min(2,2)} = 2^2 = 4$* $3^{\min(1,1)} = 3^1 = 3$multiply:

$$4 \times 3 = 12$$

GCD(84, 60) = 12
Output:
{{"tag": "Greatest Common Divisor", "explanation": "The problem requires computing the largest integer that divides both 84 and 60."}}, {{"tag": "Prime Factorization", "explanation": "One method of finding the GCD involves decomposing numbers into their prime factors."}}, {{"tag": "Euclidean Algorithm", "explanation": "An efficient iterative method to compute the GCD without full factorization."}}

Problem: Where is the Peking University
Reference Answer: No. 5 Yiheyuan Road, Haidian District, Beijing, China.
Output:
{{"tag": "Peking University Location", "explanation": "Identifying the geographical location of Peking University, including its city and district."}}, {{"tag": "Beijing Geography", "explanation": "Knowledge of the layout and districts of Beijing to contextualize the university's location."}}, {{"tag": "Higher Education Institutions in China", "explanation": "Understanding the placement and distribution of major universities in China for reference."}}

Problem:Which Sun sign, Rising sign, and Moon sign does a person born on September 17 (Gregorian calendar) belong to? Why are there differences among them?
Reference Answer:Sun sign: Virgo (because September 17 falls within the Virgo date range). Rising sign (Ascendant): Requires the **exact time and place of birth** to determine. Moon sign: Also requires the **exact time and place of birth** to calculate. The Sun sign depends only on the birth date, while the Rising sign and Moon sign vary with the **specific birth time and location**, as they change much more rapidly.

Output:
{{"tag": "Sun Sign Determination", "explanation": "Identifying the sun sign based on the Gregorian date of birth"}}, {{"tag": "Ascendant Sign Calculation", "explanation": "Determining the rising sign based on precise birth time and location"}}, {{"tag": "Moon Sign Calculation", "explanation": "Determining the moon sign based on the moon's position at the time of birth"}}, {{"tag": "Astrological Principles", "explanation": "Understanding why sun sign, moon sign, and ascendant can differ due to celestial mechanics and astrological rules"}}

**Now analyze the following input and return the list of atomic knowledge points. You should only output the final json. **

Problem:

{problem}

Reference Answer:

{reference_answer}

Output: '''

Table 12: Instruction template for GPT-Tagger Eval.

Instruction template for GPT-Checker

You are a tag validation expert.
You are given a problem description, reference answer and the extracted tag results.
Your task is to **evaluate the tag results** based on the following criteria:

Validation Criteria
1. **Output Format**:
- Must be a JSON object.
- "tags" must be a list of dictionaries, each with fields "explanation" and "tag".

2. **Content Reasonableness**:
- Tags must match the problem description.
- Tags must be meaningful and reasonably specific.

3. **Accuracy and Granularity**:
- No vague categories (e.g. "Math" or "Data Structures").
- Tags should not be overly broad (e.g. "Probability") nor overly fine-grained (e.g. "apply addition in Bayes formula").
- Aim for an appropriate middle-level granularity (e.g. "Discrete Event Probability", "Binary Search Tree Traversal", "Video file handling in Python").
- Tags must be correct, contains the core constraints and relevant to the problem.

4. **Coverage**:
- Tags should cover the **essential core knowledge points** required to understand or solve the problem.
- Maximum 5 tags; unnecessary extra tags should not appear.

Your Task
- If the output fully satisfies the criteria, respond with:
"check":"Yes"

- If the output fails in any aspect, respond with:
"check":"No", "Reason": "<concise reason for failure, with direct guidance to refine the tagging>"

—
Problem:
{problem_description}

Reference Answer:

{reference_answer}
Extracted Tags:
{tag_output}

Table 13: Instruction template for GPT-checker.

Instruction template for Coverager Eval

You are given two tag lists:
1. **Raw Tags**: These are the ground-truth tags that represent concepts or labels in the data.
2. **Generated Tags**: These are tags automatically generated by a system.

Your task is to check whether each raw tag is **covered** by the generated tags. Please determine whether each raw tag was covered by the generated tag.

Coverage criteria include:
1. Synonymous or closely related expressions If the generated tag expresses the same or very similar meaning as the raw tag (including synonyms or paraphrases).
2. Containment relationship If the generated tag directly contains the core concept of the raw tag, it counts as coverage.For example, "subarray counting" covers "array".
3. Hypernym-hyponym coverage If the generated tag is a more specific concept or includes the core meaning of the raw tag, it is considered coverage. For example, "sliding window algorithm" covers "array".
4. Consistent domain terminology Different expressions of the same technical term count as mutual coverage. For example, "greatest common divisor" and "GCD" cover each other.
5. Composite tag inclusion If the generated tag is a composite concept and its core includes the raw tag's core term with precise semantics, it counts as coverage. For example, "quick sort algorithm" covers "sorting".

Excluded cases (do not count as coverage):
Overly broad or generic tags are not considered coverage, even if they belong to the same general domain.
The generated tag must express a specific concept that directly includes or aligns with the raw tag's meaning.

Output format:
Return a dictionary, where each key is a raw tag, and the value is '1' if it is covered by any generated tag, otherwise '0'.

Example:
Raw Tags: ["Named Entity Recognition", "POS Tagging", "DFS"]
Generated Tags: ["Part-of-Speech Tagging", "NER"]
Output: {{ "Named Entity Recognition": 1, "POS Tagging": 1, "DFS": 0 }}

You should output the dict in str format only, no analysis. Please carefully consider the intrinsic relationship between the generated tag and the raw tag, not just based on surface-level meaning. You may also make reasonable inferences — for example, character-level operations can be considered part of the string category.

Now evaluate:
Raw Tags: {raw_tags}
Generated Tags: {generate_tags}

Table 14: Instruction template for Coverager Eval.

Instruction template for Fine-grained Eval

You are given two tag lists:
1. **Raw Tags**: These are the ground-truth tags that represent concepts or labels in the data.
2. **Generated Tags**: These are tags automatically generated by a system.

Your task is to check whether each generated tag is **accurate**, meaning it correctly corresponds to at least one raw tag.

Accuracy criteria include:
1. Synonymous or closely related expressions
If the generated tag expresses the same or very similar meaning as any raw tag (including synonyms or paraphrases).
2. Containment relationship
If the generated tag directly contains the core concept of any raw tag, it counts as accurate. For example, "subarray counting" is accurate if the raw tags contain "array".
3. Hypernym-hyponym relationship
If the generated tag is a more specific concept or includes the core meaning of a raw tag, it is considered accurate.
4. Consistent domain terminology
Different expressions of the same technical term count as mutual accuracy. For example, "greatest common divisor" and "GCD" are accurate matches.
5. Composite tag inclusion
If the generated tag is a composite concept and its core includes the raw tag's core term with precise semantics, it counts as accurate.

Excluded cases (do not count as accurate):
- Overly broad or generic generated tags that do not specifically match any raw tag.
- Tags unrelated to any raw tag in meaning.

Output format:
Return a dictionary, where each key is a generated tag, and the value is ‘1‘ if it is accurate, otherwise ‘0‘.

Example:
Raw Tags: ["Named Entity Recognition", "POS Tagging", "DFS"]
Generated Tags: ["Part-of-Speech Tagging", "NER", "Graph Algorithm"]

Output:
{{ "Part-of-Speech Tagging": 1, "NER": 1, "Graph Algorithm": 0 }}

You should only output the dict in str format, donnot output analysis.

Now evaluate:
Raw Tags: {raw_tags}
Generated Tags: {generate_tags}

Table 15: Instruction template for Precision Eval.

Instruction template for Precision Eval

You are given a list of generated tags.

Your task is to determine for each tag whether it is a **fine-grained** knowledge concept.

Criteria:
- Fine-grained tags are specific, well-defined concepts or techniques.
- Tags that are broad should be marked as not fine-grained.
- A moderate level of granularity is also acceptable — for example, 'solving quadratic equations' is considered fine-grained.
- For example, "Quick Sort" is fine-grained, but "Sorting Algorithms" is not.

Output format:
Return a dictionary where each key is a generated tag, and the value is '1' if it is fine-grained, or '0' if not.

Example:
Generated Tags: ["Quick Sort", "Sorting Algorithms", "Graph Traversal"]

Output:
{{ "Quick Sort": 1, "Sorting Algorithms": 0, "Graph Traversal": 0 }}

Only output the dictionary in string format without any explanation.

Now evaluate:
Generated Tags: {generate_tags}

Table 16: Instruction template for Fine-grained Eval.

Instruction template for Instance-level Eval

You are an expert taxonomist. Your task is to evaluate hierarchical relationships between concepts. Please evaluate the relationship between the Parent and Child:

Parent Topic: {name}
Child Topic: {cluster_item}

Criteria:
- Return 1 if the Child Topic is logically a sub-topic, component, or relevant instance of the Parent Topic.
- Return 0 if the Child Topic is unrelated or clearly incorrect under this parent.

Output format: Return ONLY the single digit (0 or 1).

Table 17: Instruction template for Instance-Level Eval.

Instruction template for Cluster-Level Eval

You are an expert taxonomist and data evaluator. Your task is to evaluate if a given Topic Name accurately reflects a list of 'Contained Clusters'.
Please evaluate the following:

Topic Name: {name}
Contained Clusters: {clusters_str}

Criteria:
- Return 1 if the Topic Name is a high-quality, accurate summary that encompasses the Contained Clusters.
- Return 0 if the Topic Name is irrelevant, misleading, or fails to cover the main themes of the clusters.

Output format: Return ONLY the single digit (0 or 1).

Table 18: Instruction template for Cluster-Level Eval.