BEYOND FIXED SIMILARITY: INSTRUCTION-FOLLOWING CLUSTERING VIA END-TO-END PARTITION GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-goal text clustering requires models to move beyond generic semantic similarity and organize text according to diverse user-specified criteria. However, existing benchmarks primarily assume fixed clustering goals or topical criteria, leaving reasoning-intensive, instruction-following clustering underexplored. We introduce REASONCLUSTER, an LLM-assisted and human-audited benchmark comprising 28 tasks across multiple domains. Its tasks require models to infer latent textual properties and produce instruction-aligned partitions. Using REASONCLUSTER, we study two routes toward instruction-following clustering. First, we study embedding-based clustering by comparing encoder-based and decoder-based embedders. Decoder-based embedders provide the strongest representations among the evaluated models, suggesting that generation-capable backbones are promising for capturing instruction-dependent textual properties. Yet all embedding-based pipelines still rely on separately configured clustering algorithms, leaving the final partition sensitive to algorithmic hyperparameter choices. We therefore explore generative clustering, where an LLM interprets the instruction, reasons over the corpus, and outputs a complete partition. Our experiments reveal that generative clustering remains challenging for general-purpose LLMs, which substantially underperform reasoning models and are prone to structurally invalid outputs as input complexity grows. We further show that task-specific post-training that optimizes output validity and partition quality can substantially strengthen this capability, enabling a 14B model to surpass o3 by 3.6 points on held-out evaluation split. These results establish REASONCLUSTER as a test bed for reasoning-intensive clustering and suggest generative clustering as a promising direction for reasoning-intensive end-to-end clustering.

1 INTRODUCTION

Text clustering is a core operation for organizing large text collections, supporting applications such as social media analysis (Park et al., 2022), accident diagnosis Xu et al. (2022), and data synthesis Zeng et al. (2025). Standard pipelines usually embed each text and then apply an external clustering algorithm such as K-means (MacQueen, 1967a; Reimers & Gurevych, 2019b). This approach is effective when clustering should reflect generic semantic similarity, but many real use cases require a different form of control: as shown in Figure 1, news can be organized by topic, argumentative stance, or another instruction-specific property (Wang et al., 2023; Peng et al., 2024). Instruction-following clustering therefore requires models to infer a partition from both the corpus and the user's goal.

Existing resources and methods only partially cover this regime. Standard intent and clustering benchmarks typically assume a fixed ontology or a single clustering goal (Casanueva et al., 2020b; Larson et al., 2019a; Demszky et al., 2020). Recent multi-goal or LLM-assisted clustering studies move closer to user-directed grouping (Zhang et al., 2023; Wang et al., 2023; Viswanathan et al., 2024), but reasoning-intensive clustering that requires reasoning about latent textual properties remains underexplored.

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.

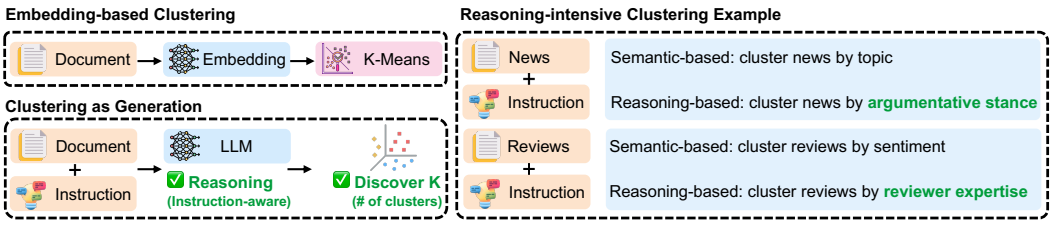


Figure 1: Overview of embedding-based vs. reasoning-driven clustering.

We introduce REASONCLUSTER, a human-audited, LLM-assisted benchmark for reasoning-intensive instruction-following clustering. It contains 28 tasks across conversational, legal, and financial domains, with 17 held-in tasks and 11 held-out tasks. The evaluation set contains 2,171 clustering instances organized into three splits, isolating different generalization axes. This split design lets us measure both robustness under familiar grouping criteria and transfer to new instruction-defined clustering goals.

Using REASONCLUSTER, we first revisit embedding-based clustering. Recent instruction-aware embedders condition representations on natural-language goals or build on generation-capable decoder backbones, making them natural candidates for goal-dependent clustering (Peng et al., 2024; Zhang et al., 2025). Among the evaluated models, decoder-based embedders produce the strongest clustering results, suggesting that generation-capable backbones are promising for encoding instruction-dependent properties. However, these pipelines still separate representation learning from partition induction. Their outputs depend on external clustering algorithms and choices such as similarity functions, initialization, and structural hyperparameters. Adapting to a new criterion may also require criterion-specific supervision and additional tuning Zhang et al. (2023).

We therefore explore *generative clustering*, in which an LLM maps a clustering instruction and an indexed corpus to a complete partition. The model must interpret the goal, compare texts globally, infer the grouping structure, and assign every item end-to-end. This formulation treats clustering as structured generation, replacing the conventional embedding–clustering pipeline with a unified model that reasons over the partition as a whole.

To address this challenge, we develop a post-training recipe for generative clustering that combines reasoning distillation with reinforcement learning (RL). The recipe first uses synthesized reasoning traces to warm up the model, and then optimizes structural validity, cluster-count accuracy, and partition quality through hybrid rewards. Our post-trained 14B model achieves the strongest split-level macro-average performance among the evaluated models, reaching 73.09, 73.08, and 62.83 V-measure on C0, C1, and C2, respectively. In particular, on the fully held-out C2 split, where clustering instructions, clusters, and texts are unseen during training, it surpasses o3 by 3.6 V-measure points (62.83 vs. 59.24).

Together, our results show that goal-sensitive representations are useful for instruction-conditioned clustering, while direct structured generation provides a viable route to jointly infer cluster structure and assignments. In summary, our contributions are threefold:

- We introduce REASONCLUSTER, a human-audited benchmark for evaluating reasoning-intensive, instruction-following clustering across diverse domains and generalization settings.
- We formulate clustering as end-to-end generation and develop a post-training recipe that addresses the challenge posed by REASONCLUSTER, resulting in a 14B model that outperforms o3.
- We provide empirical studies of representation, reasoning supervision, and scaling, offering practical insights into building instruction-following clustering systems.

2 RELATED WORK

Instruction-following embedders. General-purpose embedding models primarily capture task-agnostic semantic similarity (Mikolov, 2013; Pennington et al., 2014; Devlin, 2018; Reimers & Gurevych, 2019a; Gao et al., 2021; Sentence-Transformers, 2021), making them ill-suited for clustering under user-specified criteria. Instruction-tuned embedders (Su et al., 2023; Peng et al., 2024;

Zhang et al., 2025) incorporate natural-language prompts to better align representations with user intent, but still depend on external clustering algorithms with manually chosen hyperparameters and limited capacity to infer cluster structure. We instead cast clustering as an end-to-end generative task, training LLMs to infer both the cluster structure and grouping logic given the instruction.

LLM-involved clustering. Prior works have explored the use of LLMs for prompt-based clustering, leveraging natural-language descriptions, constraints, or iterative refinement to guide cluster grouping decisions (Zhang et al., 2023; Wang et al., 2023; Viswanathan et al., 2024; Feng et al., 2024; De Raedt et al., 2023; Tipirneni et al., 2024; Nakshatri et al., 2023). In contrast, we study a single-generation interface in which one LLM receives only a clustering instruction and an unlabeled corpus, jointly infers the number of clusters and the assignments, and directly emits the complete partition. We further explicitly post-train the model for this end-to-end partition-generation objective. See Appendix A for a broader discussion.

Clustering as Generation Task

You are a clustering assistant to do text clustering. Given a clustering goal and a list of indexed corpus...
Final Output Format should be:
<think> assistant's reasoning process here </think>
<answer>
Total clusters: [N].
cluster1: [item_numbers separated by commas].
cluster2: [item_numbers separated by commas].
...
</answer>
Now, please follow the format for the following clustering task:
Goal: {clustering instruction}
Text: {enumerated texts}

Figure 2: The system prompt for clustering.

3 Problem Definition

As illustrated in Figure 2, we treat instruction-following clustering as a generative task. The input consists of a single clustering instruction along with a list of texts, and the output involves the cluster number and the assignment of clusters.

Formally, let $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ denote a dataset of n texts, where each $\mathbf{x}_i \in \mathcal{X}$ is a token sequence from the input space $\mathcal{X}$. Let $I \in \mathcal{I}$ represent a natural language instruction drawn from the instruction space $\mathcal{I}$, describing the intended clustering objective. The task is to produce a clustering $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ that partitions $\mathcal{D}$ into k non-overlapping subsets. In contrast to traditional methods, the number of clusters k is not pre-specified but instead inferred from the instruction and corpus, i.e., $k = f(I, \mathcal{D})$, where f denotes a reasoning process that aligns latent corpus structure with user intent.

4 Benchmark Construction

In this section, we present ReasonCluster, a benchmark designed to evaluate whether clustering methods can group documents according to diverse user instructions. Sec 4.1 describes the LLM-assisted data synthesis, Sec 4.2 explains the data collection across three domains, Sec 4.3 presents the details of constructing the training set and evaluation set.

4.1 LLM-assisted Data Synthesis

Our data synthesis pipeline consists of three stages: cluster generation, cluster refinement, and consensus-based multi-agent labeling. Figure 3 visualizes this process using a concrete legal case example, which we reference below to clarify the role of each agent.

Cluster Generation. Given a dataset comprising heterogeneous corpora, we first create candidate clustering instructions that emphasize reasoning-intensive perspectives and their associated clusters. As shown in Figure 3 (step 1), given a set of legal case documents, the generation agent proposes candidate clustering instructions (e.g., "cluster legal cases by primary vulnerable population involved")

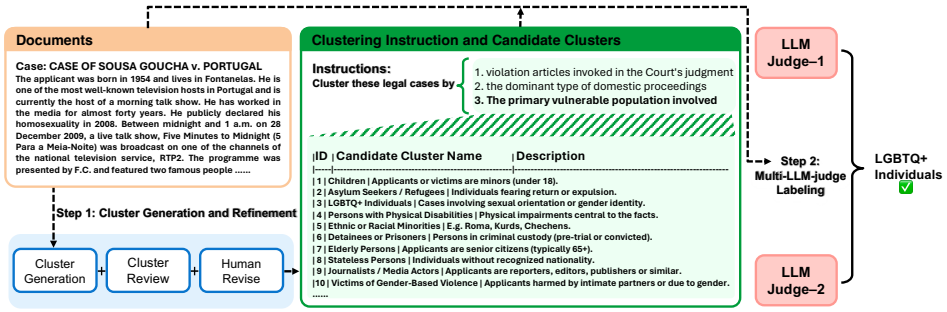


Figure 3: The overview of our multi-agent data synthesis pipeline.

together with an initial set of candidate clusters, such as "LGBTQ+ Individuals" or "Children." This stage yields an initial hypothesis space of possible instructions and clusters.

Cluster Refinement. The generation agent's outputs are passed to a refinement agent that enforces two formal constraints: (i) *mutual exclusivity*, ensuring non-overlapping categories, and (ii) *collective exhaustiveness*, ensuring coverage of the relevant category space for the retained benchmark subset. This agent yields a candidate cluster taxonomy with explicit names and descriptions (green box in Figure 7). After this automated refinement, human annotators conduct a final task-level review. Specifically, they check whether the clustering instruction reflects a reasoning-intensive grouping goal, whether the taxonomy is semantically reasonable, whether categories are mutually exclusive, and whether the category granularity is consistent. Instructions or taxonomies that are vague, overly broad, overlapping, or not sufficiently reasoning-intensive are revised before entering the labeling stage.

Cross-Model Consensus Labeling. After a refined cluster set is finalized, two independently prompted labeling models from different model families, GPT-OSS-120B and Phi-4-reasoning, assign each document to one cluster based on the refined taxonomy. We retain an assignment only when both models independently select the same cluster; cases where the models disagree or where a document does not clearly fit the taxonomy are discarded to avoid ambiguous boundary examples. Before large-scale labeling, we manually inspect 10 pilot examples from each domain and proceed only after both labeling models align with human verification on all pilot examples. To further assess the reliability of the resulting benchmark labels, we conduct a post-labeling human audit by randomly sampling 10 retained examples from each of the 28 tasks, yielding 280 audited examples in total. Human judgments agree with the retained consensus labels on 272 of 280 examples (97.1%). Detailed model assignments, human verification procedures, and domain-level audit results are provided in Appendix B.

Type	Benchmark	# Goals	# Clusters	Avg. Len (token)	Reasoning-based
Single Goal	Banking77 (Casanueva et al., 2020b)	1	77	13.9	No
	CLINC (Larson et al., 2019a)	1	150	8.6	No
	GoEmo (Demszky et al., 2020)	1	27	16.1	No
Multi Goal	GOALEX (Wang et al., 2023)	3	4	51.2	No
	NYTClustering (Peng et al., 2024)	2	10	887.8	No
	REASONCLUSTER	28	14	673.4	Yes

Table 1: Comparison of clustering benchmarks.

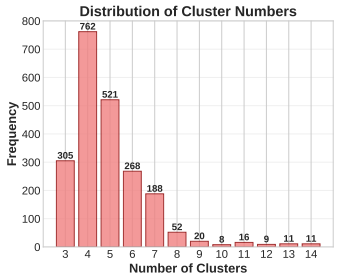
4.2 DATASETS

We construct our benchmark using three complementary datasets that span conversational, legal, and financial domains, enabling evaluation of instruction-following clustering under diverse linguistic structures and reasoning demands.

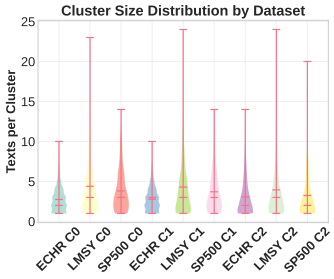
LMSYS-Chat. Prior intent-discovery benchmarks such as Bank77 and CLINC (Tabel 1) (Casanueva et al., 2020b; Larson et al., 2019a; FitzGerald et al., 2022) focus on short, single-turn utterances with narrowly defined intents, limiting their ability to capture the contextual and complex nature

Source	Split	text len	input len	#data
ECHR	C0	696.4	9872.4	150
	C1	687.9	10239.9	150
	C2	959.6	12643.6	132
LMSY	C0	410.5	9237.8	500
	C1	411.3	9042.7	500
	C2	478.1	7201.9	250
SP500	C0	772.5	13420.6	179
	C1	809.1	13810.5	170
	C2	836.6	14405.6	140

(a) Evaluation dataset statistics.



(b) Cluster number distribution.



(c) Cluster size by dataset.

Figure 4: Statistics of the benchmark splits, text and input lengths are measured in tokens.

of real-world dialogue. We therefore use LMSYS-Chat, a large-scale corpus of real human–LLM conversations, to evaluate instruction-following clustering under realistic conversational settings.

ECHR and S&P 500. We include long-form documents from legal and financial domains. These texts are structurally complex and require substantive domain-specific reasoning, posing clustering challenges that differ fundamentally from dialogue. We aim to test whether models can group documents according to diverse abstract criteria beyond surface semantics. Detailed data preprocessing procedures, filtering criteria, and dataset statistics are provided in Appendix H.

4.3 Benchmark Details

Building on the synthesis pipeline in Section 4.1, we curate a benchmark that comprises 28 tasks, partitioned into 17 held-in tasks (available during training) and 11 held-out tasks (reserved exclusively for evaluation). Each task consists of one clustering instruction and a set of clusters, each populated with domain-specific texts. For all held-in tasks, we randomly split the texts within every cluster into two pools: 60% of the texts are used to construct the training pool, while the remaining 40% are reserved for in-domain evaluation.

Training set. Training examples are generated by (i) randomly sampling a subset of clusters from a task and (ii) sampling multiple texts from the training pool of each selected cluster.

Evaluation splits. We build three evaluation splits that probe distinct generalization axes:

- **C0** (seen tasks and seen texts, unseen cluster combinations): created by re-sampling a new subset of clusters and texts from the 60% training pool that are not used to form training instances.
- **C1** (seen tasks, unseen texts and unseen cluster combinations): constructed from the 40% per-cluster reserved evaluation pool. The clustering instructions are seen during training, but none of the texts in the evaluation pool appear in the training.
- **C2** (unseen tasks, unseen texts, and unseen cluster combinations): drawn exclusively from the 11 held-out tasks whose clustering instructions, clusters, and texts never appear during training.

This protocol isolates different generalization axes: C0 tests recombination robustness with familiar instructions and texts, C1 tests transfer to new texts under familiar instructions, and C2 tests transfer to entirely new instructions and texts. The statistics of our benchmark are shown in Figure 4. Details of clustering instructions are provided in Appendix H.4.

5 Methodology

Following previous reasoning-oriented post-training pipelines (Guo et al., 2025; Huang et al., 2025; Chen et al., 2025), we adopt a two-stage post-training recipe for generative clustering: an SFT warm-up to stabilize subsequent reward optimization, followed by RL.

5.1 Reasoning Distillation

For a small subset of training instances, a teacher model receives the clustering goal and indexed corpus, then produces both a reasoning trace and a final clustering answer. We retain traces whose final

Model	LMSY			ECHR			SP500			AVG			OVERALL
	C0	C1	C2	C0	C1	C2	C0	C1	C2	C0	C1	C2	V
K-means													
Qwen3-Embedding	**55.88**	**56.28**	**53.26**	**74.58**	**71.87**	**59.45**	53.41	**54.67**	**60.81**	**61.29**	**60.94**	**57.84**	**60.02**
InBedder	52.19	52.05	45.95	67.46	66.08	48.13	53.06	53.97	49.72	57.57	57.37	47.93	54.29
Instructor	47.20	47.04	44.04	58.12	57.42	47.30	48.74	49.12	47.28	51.35	51.19	46.21	49.58
OpenAI-Embedding	50.63	51.28	49.11	56.58	55.99	45.75	47.74	48.82	48.47	51.65	52.03	47.78	50.49
all-MiniLM-L6-v2	44.39	45.07	43.27	58.00	57.58	49.07	48.44	48.78	48.75	50.27	50.48	47.03	49.26
Gaussian Mixture													
Qwen3-Embedding	55.64	56.13	52.23	72.22	71.64	59.23	**54.12**	54.53	59.77	60.66	60.77	57.08	59.50
InBedder	49.02	49.40	42.90	63.95	64.49	46.53	48.82	49.85	46.36	53.93	54.58	45.26	51.26
Instructor	42.89	43.39	41.52	55.75	55.16	44.51	44.55	46.46	44.80	47.73	48.34	43.61	46.56
OpenAI-Embedding	46.11	47.68	44.98	55.84	55.46	44.12	44.83	45.94	46.84	48.93	49.69	45.32	47.98
all-MiniLM-L6-v2	40.80	41.48	40.09	54.01	54.05	44.57	43.67	44.76	47.51	46.16	46.76	44.06	45.66

Table 2: Instruction-tuned embedders outperform generic embedders. Each embedder is paired with K-means or Gaussian Mixture Models, with the gold number of clusters provided to the clustering algorithm. For each split, the AVG column reports the macro-average across the three datasets. Bold indicates the best result in each column.

answer matches the reference partition and further pass them to a review agent that checks reasoning completeness, length appropriateness, and logical coherence. The resulting input–trace pairs are used to supervise the student model via the standard language modeling objective, encouraging it to internalize structured reasoning patterns aligned with instruction-following clustering. This warm-up is not intended to make imitation the final learning objective. Rather, it moves the policy into a region where outputs are more likely to be parseable and task-shaped, so that subsequent reward optimization can provide meaningful feedback on format validity, reasoning patterns and partition quality. Additional details on trace generation, filtering, and refinement are provided in Appendix C.

5.2 Reinforcement Learning with Multiplicative Hybrid Rewards

Starting from the SFT warm-up checkpoint, we apply reinforcement learning to optimize the properties required by generative clustering: structural validity and high-quality partition alignment. We decompose the overall reward into a format-level constraint and a clustering-level objective:

$$\mathcal{R}(y) = \mathcal{R}_{\text{format}}(y) + \mathcal{R}_{\text{clust}}(y). \tag{1}$$

Format reward. The format reward enforces strict syntactic and structural correctness of the generated outputs to avoid missing items, duplicate assignments, or malformed cluster declarations. Degenerated outputs are unusable regardless of semantic quality. We therefore define:

$$\mathcal{R}_{\text{format}}(y) = \begin{cases} 1 & \text{format constraints satisfied,} \\ -1 & \text{otherwise.} \end{cases} \tag{2}$$

Further details of format constraints are provided in Appendix D.

Clustering reward. Beyond structural validity, effective clustering requires both accurate inference of the number of clusters and high-quality cluster assignment. Accordingly, we design the clustering reward as a composition of *count reward* and *quality reward*:

$$\mathcal{R}_{\text{clust}}(y) = \big(\mathcal{R}_{\text{count}}(y) + \epsilon\big)\big(\mathcal{R}_{\text{qual}}(y) + \epsilon\big) - \epsilon, \tag{3}$$

where $\epsilon = 0.1$ is a small constant introduced for numerical stability. The count reward $\mathcal{R}_{\text{count}}$ supervises the model's ability to infer the number of clusters from the instruction and corpus. We define:

$$\mathcal{R}_{\text{count}}(y) = \gamma^{|\hat{K}-K|}, \tag{4}$$

where $\hat{K}$ and K denote the predicted and ground-truth cluster counts, respectively, and $\gamma \in (0, 1)$ is a decay factor (default $\gamma = 0.7$) that penalizes larger deviations more severely. The quality reward $\mathcal{R}_{\text{qual}}$ measures the semantic coherence of the predicted clustering using V-measure, a standard clustering metric that balances homogeneity and completeness (see Appendix G):

$$\mathcal{R}_{\text{qual}}(y) = V(\hat{\mathcal{C}}, \mathcal{C}^*). \tag{5}$$

Model	LMSY			ECHR			SP500			AVG			OVERALL
	C0	C1	C2	C0	C1	C2	C0	C1	C2	C0	C1	C2	V
Reasoning Models													
o3	63.60	63.42	**64.17**	**85.75**	**85.79**	47.04	57.01	59.04	66.51	68.79	69.42	59.24	65.82
Gemini 2.5 Pro	55.98	56.06	57.95	85.05	84.14	55.47	57.60	63.43	**69.34**	66.21	67.88	60.92	65.00
GPT-oss-120B	48.74	49.30	52.77	70.54	69.18	52.58	44.76	48.15	51.84	54.68	55.54	52.40	54.21
DeepSeek-R1	42.96	43.64	45.80	53.78	54.55	34.06	41.57	40.47	39.69	46.10	46.22	39.85	44.06
Distill-Llama-70B	37.80	39.04	47.76	63.45	66.40	57.79	39.96	43.78	42.13	47.07	49.74	49.23	48.68
Distill-Qwen-32B	26.50	27.89	39.45	54.43	53.15	44.04	31.82	33.91	38.74	37.58	38.32	40.74	38.88
QwQ-32B	47.51	48.84	53.74	75.98	72.63	56.68	51.95	55.30	63.23	58.48	58.92	57.88	58.43
General Models													
GPT-4o	26.95	30.15	33.38	58.86	56.15	52.86	40.53	40.26	31.15	42.11	42.19	39.13	41.14
GPT-4.1	37.27	39.99	43.42	64.93	59.87	40.05	42.45	44.64	41.27	48.22	48.17	41.25	45.88
Llama-3.1-70B-Instruct	24.57	27.50	28.97	42.69	44.58	35.77	36.21	37.66	32.28	34.49	36.58	32.34	34.47
Qwen2.5-72B-Instruct	18.85	20.08	27.78	45.57	44.03	51.77	35.34	36.17	33.28	33.25	33.43	37.61	34.76
Our Models													
C1-Qwen-7B	65.63	65.04	58.51	81.81	82.76	52.66	**69.35**	**68.08**	60.28	72.26	71.96	57.15	67.12
C1-Qwen-14B	**66.90**	**66.47**	62.65	84.05	84.80	**66.39**	68.33	67.96	59.44	**73.09**	**73.08**	**62.83**	**69.67**

Table 3: V-measure (%) across three datasets and evaluation splits (C0, C1, C2; see Section 4.3). Bold indicates the best result.

We optimize the model using the standard GRPO objective with KL regularization and group-relative advantage estimation (Shao et al., 2024).

6 Experimental Settings

Baselines For embedding-based methods, we evaluate general-purpose embeddings (e.g., OpenAI-Embedding (OpenAI, 2024a)[1], all-MiniLM-L6-v2 (Sentence-Transformers, 2021)) and instruction-tuned embedding models (e.g., InBedder (Peng et al., 2024), Instructor (Su et al., 2023)), each combined with classical clustering algorithms such as K-means and GMMs. The implementation details for all the baselines are provided in Appendix E. For generative clustering, this group includes proprietary and public reasoning models (e.g., o3 (OpenAI, 2025c), Gemini-2.5-Pro (Google DeepMind, 2025), GPT-oss-120B (OpenAI, 2025b), DeepSeek-R1 (Guo et al., 2025), DeepSeek-R1-Distill-Llama-70B (deepseek-ai, 2025a), DeepSeek-R1-Distill-Qwen-32B (deepseek-ai, 2025b), and QwQ-32B (Qwen Team, 2025)), general LLMs (e.g., GPT-4o (OpenAI, 2024b), GPT-4.1 (OpenAI, 2025a), Llama-3.1-70B-Instruct (Meta AI, 2024), and Qwen2.5-72B-Instruct (Yang et al., 2024)).

Implementation Details We adopt the Qwen-2.5-Instruct series models (Yang et al., 2024) as the backbone due to their strong reasoning capabilities acquired during pre-training. Comprehensive configuration and training datasets are provided in Appendix F. For evaluation, we employ the V-measure metric, following established practice in clustering assessments (Muennighoff et al., 2022; Rosenberg & Hirschberg, 2007). Formal description of V-measure is shown in Appendix G.

7 Main Results

Goal-sensitive embedders improve over generic embedders. Table 2 shows that Qwen3-Embedding achieves the strongest embedding-based performance, reaching 60.02% overall V-measure with K-means and 59.50% with Gaussian Mixture Models. InBedder also improves over generic semantic embedders in most settings, while Instructor shows more mixed gains. These results suggest that multi-goal clustering benefits from goal-sensitive representations, while cluster-count inference and partition construction remain delegated to a separate clustering algorithm.

Reasoning-oriented models are stronger than general LLMs under the generative interface. As shown in Table 3, general-purpose models such as GPT-4o and GPT-4.1 achieve 41.14% and 45.88% overall V-measure, respectively, whereas reasoning-oriented models achieve substantially higher scores, including 65.82% for o3, 65.00% for Gemini 2.5 Pro, and 58.43% for QwQ-32B. This

[1]The latest OpenAI embedding model as of September 12, 2025: `text-embedding-3-large` (`https://platform.openai.com/docs/guides/embeddings`)

Method	C0	C1	C2	Ave.
7B Base Model				
+ RL	46.72	48.19	46.52	44.90
+ RL + Reasoning Hint	48.19	49.99	46.07	46.42
+ Distill + RL + Reasoning Hint	66.05	65.97	56.46	61.80
14B Base Model				
+ RL	56.39	53.57	53.71	52.76
+ RL + Reasoning Hint	66.05	66.40	60.87	62.41
+ Distill + RL + Reasoning Hint	68.84	69.21	59.71	64.43

Table 4: Comparison of training strategies across 7B and 14B base models.

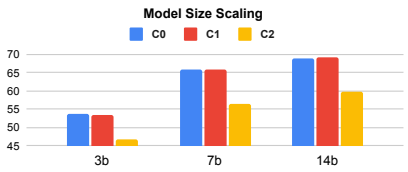


Figure 5: Effect of model size on clustering performance and generalization. The y-axis shows the average V-measure over all corresponding samples.

gap suggests that bounded-context generative clustering requires more than following the output instruction: models must maintain structural validity, compare items globally, and infer a coherent grouping criterion. Nevertheless, format accuracy alone is not sufficient for strong clustering quality; detailed format error analysis is presented in Figure 17.

C1-QWEN performs strongly under the reasoning-intensive generative clustering interface. Table 3 shows that C1-QWEN-14B achieves the highest overall V-measure among the evaluated models, reaching 69.67%. Its gains are clearest on C0 and C1, where it obtains split-level averages of 73.09% and 73.08%, respectively. On C2, where clustering goals, clusters, and texts are unseen during training, C1-QWEN-14B achieves a 62.83% average V-measure. Although this is the highest C2 macro-average among the evaluated models, frontier reasoning models remain stronger on particular domains, such as LMSY and SP500. We therefore treat C2 as the primary stress test for transfer to unseen clustering goals rather than relying only on the overall aggregate. Case studies are provided in Appendix J.

8 ANALYSIS

This section presents empirical analyses to identify the key steps required to enable effective generative clustering. Our study covers the impact of reasoning training, training strategies, and scaling effects.

8.1 TRAINING RECIPES FOR ELICITING EFFECTIVE REASONING

This section investigate the key steps for eliciting effective reasoning in instruction-following clustering. Specifically, we compare:

(1) RL: Applies cold-start RL with rule-based rewards focused on cluster correctness and format compliance (see Section 5). The model learns solely from reward feedback without any prior reasoning guidance.

(2) RL + Reasoning Hint: Extends the baseline RL setup by modifying the prompt to include explicit reasoning hints. The prompt first instructs the model to infer latent clusters and then assign items accordingly, encouraging step-by-step reasoning rather than direct prediction. The reasoning hint prompt remain consistent with the distillation prompt (see the think process in Figure 11).

(3) SFT + RL + Reasoning Hint: Builds on the previous setup with additional distillation from teacher models as a warm start before RL training. With RL alone, weaker models (especially at smaller scales) often fail to explore high-quality reasoning chains for clustering. Distilling strong reasoning traces on a small subset of data mitigates this limitation.

SFT + RL + Reasoning Hint achieves the strongest overall performance. Table 4 shows that incorporating reasoning hints generally improves clustering performance, with particularly clear gains for the 14B model. The larger improvement observed for the 14B model suggests that the effectiveness of explicit reasoning guidance may depend on model capacity. One plausible explanation is that stronger base models, which generally exhibit stronger reasoning and zero-shot instruction-following capabilities, are better positioned to interpret and leverage such structured cues during RL. Building on reasoning hints, introducing SFT with distilled reasoning traces further improves the overall performance, especially for the 7B model, indicating that a supervised warm start can

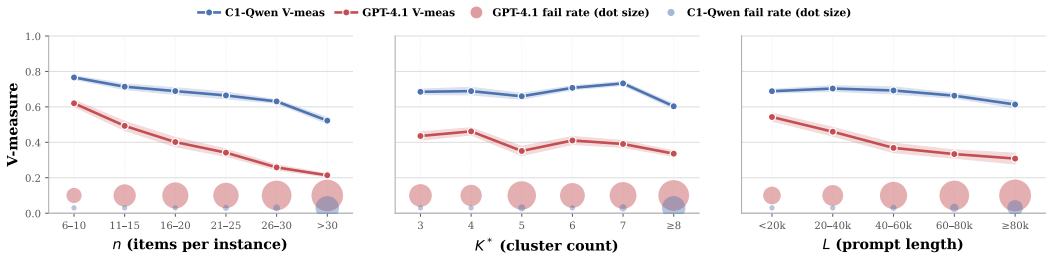


Figure 6: Scaling behavior across three input complexity dimensions.

provide useful task-specific reasoning patterns before reward optimization. As shown in Figure 13, the SFT-initialized model also exhibits smoother response-length dynamics and reaches higher reward values earlier during subsequent RL training. Overall, these results suggest that reasoning hints can help elicit more structured clustering behavior, while combining reasoning-oriented SFT with RL provides a more effective training recipe for instruction-following clustering.

8.2 Effect of Scaling Model size

We investigate how instruction-following clustering performance scales with model capacity. Using the Qwen-2.5-Instruct model series, we compare models with 3B, 7B, and 14B parameters. All models are trained using our full two-stage pipeline (Section 5). As illustrated in Figure 5, performance scales smoothly with model size across all evaluation conditions. Larger models exhibit both higher average V-measures and smaller performance gaps between seen and unseen clustering tasks, indicating enhanced generalization and reasoning robustness.

8.3 Scaling analysis

To study robustness under increasing input complexity, we group evaluation instances along three axes: the number of items per instance n, the true number of clusters K^*, and the input prompt length L. For each bin, we compute the mean V-measure and format failure rate using 30 bootstrap resamples of 100 instances. Figure 6 shows a consistent pattern across all three axes. As input complexity increases, GPT-4.1 suffers substantial drops in V-measure together with sharply rising format failure rates, suggesting difficulty in maintaining both structural validity and clustering quality. In contrast, C1-Qwen degrades more gradually and remains highly stable across most bins, with low failure rates even under longer prompts and larger instances. These results indicate that our training pipeline improves not only clustering quality but also robustness to scaling-related failure modes. Additional numerical results and a detailed breakdown are provided in Appendix J.2.

9 Conclusion

We study instruction-following clustering in the multi-goal setting, where models must align clustering behavior with diverse user instructions rather than fixed semantic similarity. To support this setting, we introduce ReasonCluster, a benchmark that systematically evaluates reasoning-intensive clustering across domains and instruction types. We further cast clustering as a structured generation problem, enabling models to jointly interpret instructions and induce cluster structure within a unified reasoning process. Combined with a reasoning-driven training pipeline, this formulation allows LLMs to perform clustering end-to-end without relying on external algorithms. We provide empirical studies of representation, reasoning supervision, and scaling, offering practical insights into enbaling instruction-following clustering. Overall, we hope this work provides a testbed for studying instruction-following clustering in more realistic settings, and inspires future research on memory-augmented, scalable clustering systems that overcome context limitations and extend to web-scale data.

REFERENCES

Roger K Blashfield and Mark S Aldenderfer. The literature on cluster analysis. *Multivariate behavioral research*, 13(3):271–295, 1978.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*, 2020a.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, 2020b.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. Neural legal judgment prediction in English. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, et al. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*, 2025.

Maarten De Raedt, Fréderic Godin, Thomas Demeester, and Chris Develder. Idas: Intent discovery with abstractive summarization. *arXiv preprint arXiv:2305.19783*, 2023.

deepseek-ai. Deepseek-r1-distill-llama-70b. `https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-70B`, 2025a. Model card; Accessed: 2025-09-22.

deepseek-ai. Deepseek-r1-distill-qwen-32b. `https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B`, 2025b. Model card; Accessed: 2026-08-04.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1): 1–22, 1977.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. GoEmotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pp. 226–231, 1996.

Zijin Feng, Luyang Lin, Lingzhi Wang, Hong Cheng, and Kam-Fai Wong. Llmedgerefine: Enhancing text clustering with llm-based boundary point refinement. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18455–18462, 2024.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages, 2022.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.552. URL `https://aclanthology.org/2021.emnlp-main.552/`.

Google DeepMind. Gemini 2.5 pro: Our most advanced model. https://deepmind.google/models/gemini/pro/, 2025. Accessed: 2025-09-22.

Dong Guo et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint*, 2025. URL https://arxiv.org/abs/2501.12948.

Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42:177–196, 2001.

Peihao Huang, Yan Huang, Wei Wang, and Liang Wang. Deep embedding network for clustering. In *2014 22nd International conference on pattern recognition*, pp. 1532–1537. IEEE, 2014.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Xu Tang, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint*, 2025. URL https://arxiv.org/abs/2503.06749.

Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019a.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019b.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967a.

J MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*, 1967b.

Meta AI. Introducing Llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1/, 2024. Accessed: 2026-08-04.

Tomas Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 3781, 2013.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*, 2022.

Nishanth Nakshatri, Siyi Liu, Sihao Chen, Dan Roth, Dan Goldwasser, and Daniel Hopkins. Using llm for improving key event discovery: Temporal-guided news stream clustering with event summaries. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 4162–4173, 2023.

Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14, 2001.

OpenAI. Vector embeddings. https://platform.openai.com/docs/guides/embeddings, 2024a. Accessed: 2025-09-22.

OpenAI. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/, 2024b. Accessed: 2025-09-22.

OpenAI. Introducing GPT-4.1 in the API. https://openai.com/index/gpt-4-1/, 2025a. Accessed: 2026-08-04.

OpenAI. Introducing gpt-oss. https://openai.com/index/introducing-gpt-oss/, 2025b. Accessed: 2026-08-04.

OpenAI. Introducing OpenAI o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/, 2025c. Accessed: 2026-08-04.

June Young Park, Evan Mistur, Donghwan Kim, Yunjeong Mo, and Richard Hoefer. Toward human-centric urban infrastructure: Text mining for social media data to identify the public perception of covid-19 policy in transportation hubs. *Sustainable Cities and Society*, 2022.

Letian Peng, Yuwei Zhang, Zilong Wang, Jayanth Srinivasa, Gaowen Liu, Zihan Wang, and Jingbo Shang. Answer is all you need: Instruction-following text embedding via answering the question. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Qwen Team. QwQ-32B. https://huggingface.co/Qwen/QwQ-32B, 2025. Model card; Accessed: 2026-08-04.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019a. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of EMNLP-IJCNLP*, 2019b. URL https://arxiv.org/abs/1908.10084.

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

Sentence-Transformers. all-MiniLM-L6-v2 model card. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2, 2021. Accessed: 2025-09-22.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023.

Sindhu Tipirneni, Ravinarayana Adkathimar, Nurendra Choudhary, Gaurush Hiranandani, Rana Ali Amjad, Vassilis N Ioannidis, Changhe Yuan, and Chandan K Reddy. Context-aware clustering using large language models. *arXiv preprint arXiv:2405.00988*, 2024.

U.S. Securities and Exchange Commission. Electronic data gathering, analysis, and retrieval (edgar) system. https://www.sec.gov/edgar, n.d. Accessed: 2025-09-16.

Vijay Viswanathan, Kiril Gashteovski, Kiril Gashteovski, Carolin Lawrence, Tongshuang Wu, and Graham Neubig. Large language models enable few-shot clustering. *Transactions of the Association for Computational Linguistics*, 12:321–333, 2024.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Zihan Wang, Jingbo Shang, and Ruiqi Zhong. Goal-driven explainable clustering via language descriptions. *arXiv preprint arXiv:2305.13749*, 2023.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pp. 478–487. PMLR, 2016.

Hui Xu, Yi Liu, Chi-Min Shu, Mingqi Bai, Mailidan Motalifu, Zhongxu He, Shuncheng Wu, Penggang Zhou, and Bing Li. Cause analysis of hot work accidents based on text mining and deep learning. *Journal of Loss Prevention in the Process Industries*, 76:104747, 2022.

An Yang et al. Qwen2.5 technical report. *arXiv preprint*, 2024.

Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5147–5156, 2016.

Zhiyuan Zeng, Yizhong Wang, Hannaneh Hajishirzi, and Pang Wei Koh. Evaltree: Profiling language model weaknesses via hierarchical capability trees. In *Conference on Language Modeling (COLM)*, 2025.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shangwen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew Arnold, and Bing Xiang. Supporting clustering with contrastive learning. *arXiv preprint arXiv:2103.12953*, 2021.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Arxiv Preprint*, 2025.

Yuwei Zhang, Zihan Wang, and Jingbo Shang. ClusterLLM: Large language models as a guide for text clustering. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13903–13920, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.858. URL `https://aclanthology.org/2023.emnlp-main.858/`.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in neural information processing systems*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *International Conference on Learning Representations*, 2023.

Jie Zhou, Xingyi Cheng, and Jinchao Zhang. An end-to-end neural network framework for text clustering. *arXiv preprint arXiv:1903.09424*, 2019.

A Related Work

Traditional clustering. Classical clustering methods have long been foundational in natural language processing and data analysis. Partition-based algorithms such as K-means (MacQueen, 1967b; Lloyd, 1982) and Gaussian mixture models (Dempster et al., 1977) rely on distance or probabilistic assumptions to partition data into groups. Hierarchical approaches (Johnson, 1967; Blashfield & Aldenderfer, 1978) construct tree-like taxonomies of nested relationships, while density-based methods such as DBSCAN (Ester et al., 1996) identify clusters of arbitrary shapes without requiring the number of clusters in advance. Spectral clustering (Ng et al., 2001) leverages eigen-decomposition of similarity graphs to capture complex structures. Additionally, topic modeling approaches—including probabilistic latent semantic analysis (Hofmann, 2001) and latent Dirichlet allocation (LDA) (Blei et al., 2003)—have been widely used to uncover latent semantic topics. In the neural era, deep clustering methods (Xie et al., 2016; Yang et al., 2016; Huang et al., 2014; Zhou et al., 2019; Zhang et al., 2021) combine representation learning with clustering objectives, yielding substantial improvements over static pipelines.

Instruction-following embedders. Recent work highlights the limitations of general-purpose embeddings (e.g., Sentence-BERT (Reimers & Gurevych, 2019b), MiniLM (Wang et al., 2020), Qwen Embeddings (Zhang et al., 2025)) when clustering tasks must adapt to user-specific objectives rather than generic semantic similarity. To address this gap, instruction-following embedders condition embeddings on natural language prompts. For example, Instructor (Su et al., 2023) and InBedder (Peng et al., 2024) finetune encoders to explicitly follow task instructions, improving controllability across diverse downstream applications. While effective at aligning embeddings with user intent, these methods still treat clustering as a downstream algorithmic step, requiring separate unsupervised methods such as K-means or GMMs for grouping. Consequently, they lack the ability to autonomously infer the number of clusters or reason about structural properties of the corpus (Zeng et al., 2025).

LLM-involved clustering. Beyond embeddings, large language models (LLMs) and large reasoning models (LRMs) are increasingly used as active participants in clustering. Instead of the "embed-then-cluster" paradigm, these models unify instruction following with reasoning over latent corpus structure. Zhang et al. (2023) proposed ClusterLLM, which leverages interactive triplet and pairwise feedback for refining clusters. Viswanathan et al. (2024) showed that LLMs can augment document representations and generate pseudo-constraints for semi-supervised clustering. Instruction-guided approaches also employ LLMs to clarify boundaries via natural language explanations (Wang et al., 2023; De Raedt et al., 2023), or to iteratively refine edge cases for more coherent clusters (Feng et al., 2024). More recently, reasoning-focused LRMs have been trained with hybrid supervised distillation and reinforcement learning to produce interpretable clustering rationales, faithful taxonomies, and stable cluster counts. These approaches represent a shift toward generation-based clustering that tightly integrates user instructions with autonomous structural discovery.

B Full Prompts and Human Verification

The multi-agent data synthesis pipeline (Section 4) employs structured prompting to ensure consistency and interpretability across stages. Specifically, Figure 7 shows the taxonomy generation expert prompt used to design reasoning-based clustering dimensions, Figure 8 illustrates the refinement prompt enforcing clarity and consistency, and Figure 9 presents the classification agent prompt for assigning categories under the refined taxonomy.

Model assignment. For benchmark synthesis, the instruction-and-taxonomy generation agent is instantiated with Qwen2.5-Instruct-72B. The two independent labeling agents are instantiated with GPT-OSS-120B[2] and Phi-4-reasoning[3], respectively, so retained labels require agreement across different model families.

Human verification. Human involvement occurs at both the task and item levels. At the task level, after the refinement agent revises a candidate taxonomy, PhD-level annotators review every clustering

[2]https://openai.com/index/introducing-gpt-oss/

[3]https://huggingface.co/microsoft/Phi-4-reasoning

Domain	# Tasks	Held-in	Held-out	Overall
ECHR	6	29/30 (96.7%)	29/30 (96.7%)	58/60 (96.7%)
LMSYS	15	98/100 (98.0%)	47/50 (94.0%)	145/150 (96.7%)
S&P 500	7	39/40 (97.5%)	30/30 (100.0%)	69/70 (98.6%)
Total	28	166/170 (97.6%)	106/110 (96.4%)	**272/280 (97.1%)**

Table 5: Human audit of retained benchmark labels. We randomly sample 10 labeled examples from each of the 28 clustering tasks. Entries report the number of examples whose retained consensus label agrees with the human judgment, together with the corresponding agreement rate.

Cluster Generation Prompt

You are an expert in taxonomy design. Given (i) a description of a dataset and (ii) example text entries from it, your task is to propose meaningful clustering dimensions that capture distinct reasoning-based perspectives.
For each clustering dimension: 1. Provide a concise description of what the dimension represents. 2. List the possible cluster labels under this dimension. 3. Justify why each cluster is distinct, highlighting reasoning factors such as cause, intent, context, or outcome.
(i) {Dataset description} (ii) {Datapoint examples}
Your output must follow this structured table format:

id	name	description
1	Category A	what it is + why it's distinct
2	Category B	what it is + why it's distinct
3	Category C	what it is + why it's distinct

Figure 7: The system prompt of cluster generation agent.

instruction and its corresponding candidate clusters before the task enters labeling. Annotators verify that the instruction defines a reasoning-intensive grouping criterion, that the taxonomy is semantically reasonable, that categories are mutually exclusive, and that category granularity is consistent. Instructions or clusters with vague boundaries, overlapping definitions, or catch-all categories are revised or removed.

At the item level, we first conduct the pilot checks described in Section 4 and launch large-scale labeling only after both labeling models align with human verification on the pilot examples. After benchmark labeling is completed, we additionally conduct a task-stratified human audit. Specifically, we randomly sample 10 retained labeled examples from each of the 28 tasks, yielding 280 audited examples in total. Human judgments agree with the retained cross-model consensus labels on 272/280 examples, corresponding to an overall agreement rate of 97.1%. Table 5 provides a breakdown by domain and held-in/held-out task status.

The labeling prompt additionally allows the models to return `no_match` when an item is ambiguous or does not clearly fit the refined taxonomy. Such cases, as well as cross-model disagreements, are excluded from the retained benchmark labels rather than being forced into an uncertain category.

C Details of Reasoning Chain Generation

The SFT warm-up is designed to initialize the policy with task-shaped clustering traces before RL. For each sampled training instance, the generation agent receives only the clustering goal and the indexed corpus as input, following the prompt in Figure 11. It then produces a reasoning trace followed by a final clustering answer. We parse the final answer and compare it against the reference partition. Candidate traces are retained only if the final answer satisfies the required output format and matches the reference clustering. To further control reasoning quality, we introduce a second review agent, whose prompt is shown in Figure 12. The review agent evaluates the retained candidate traces along three dimensions: whether the reasoning provides a complete logical flow, whether its level of detail is appropriate, and whether it proceeds coherently from the clustering goal to the final grouping.

Taxonomy Refinement Expert Prompt

You are a taxonomy refinement expert. Your goal is to revise the given taxonomy so that it is clear, consistent, and practically usable. Follow the requirements below:
Requirements

- **Category name**: $\leq$ {name_length} words; concise, specific, and informative.
- **Description**: $\leq$ {desc_length} words; clearly explain what distinguishes this category.
- **Mutual exclusivity**: Categories must not overlap or contradict each other.
- **Collective exhaustiveness**: Categories together should cover all possible intents in the given context.
- **Granularity**: All categories must be defined at the same level of specificity.
- **No vague labels**: Avoid terms like "Other," "General," or "Miscellaneous."

Tasks

1. Review the existing taxonomy and suggest improvements (e.g., renaming, merging, splitting, or adding new categories).
2. Ensure each category has a clear reasoning justification and aligns with the data context.
3. If categories are missing, add enough to make the taxonomy collectively exhaustive.

Clustering Instruction Based on the refined taxonomy, check the instruction and provide a *neutral clustering instruction* that accurately reflects the categorization principle without revealing category labels. Do not include or hint at the actual category names in the instruction.

Figure 8: The system prompt for taxonomy refinement and clustering instruction design.

Expert Classification Agent Prompt

You are an expert classification agent. Your task is to analyze the given data item and assign it to the most appropriate category from the provided taxonomy.
Taxonomy {taxonomy}
Data to Classify {data_item}
Classification Instructions

1. Carefully read the data item.
2. Compare it against the taxonomy categories and their descriptions.
3. Identify the **single best-fitting category** that captures the primary characteristic of the item.
4. If the item clearly does not fit any category, or if it is ambiguous, respond with `no_match`.

Key Guidelines

- Focus only on the essential content of the data item.
- Use the category descriptions (especially "distinct because") to guide differentiation.
- Do not guess—if uncertain, use `no_match`.
- Always select **one and only one** category (or `no_match`).

Required Output Format Your response must strictly follow this format:

```
CATEGORY: [category number OR "no_match"]
REASON: [1-2 sentences explaining why this category is the best fit]
```

Figure 9: The system prompt for expert classification using a predefined taxonomy.

Traces with superficial explanations, logical gaps, inconsistent assignments, or abrupt unsupported conclusions are discarded. In our implementation, GPT-OSS-120B[4] is used as the generation agent,

[4]https://openai.com/index/introducing-gpt-oss/

Clustering Inference System Prompt

You are a helpful clustering assistant. You final answer after reasoning and thinking must follow this exact format for all responses:
Required Structure:

```
<answer>
Total clusters: [N]
cluster1: [item1,item2,item3]
cluster2: [item4,item5]
cluster3: [item6,item7,item8]
</answer>
```

Format Rules:

1. Use exactly one answer section
2. In the answer section, start with "Total clusters: [N]" where N is the number of clusters
3. List each cluster as "cluster1:", "cluster2:", etc. with items in square brackets
4. Use 1-based indexing for items (1, 2, 3, ...)
5. Every item must be assigned to exactly one cluster
6. No item should appear in multiple clusters or be missing

Figure 10: The system prompt for clustering assistant with required output format.

and Phi-4-reasoning[5] is used as the review agent. This two-stage filtering process yields warm-up data that contains both correct final partitions and usable reasoning demonstrations. The resulting distillation dataset is:

$$\mathcal{D}_{\text{warm}} = \{(x^{(i)}, y^{(i)}_{\text{trace}})\}_{i=1}^{M}, \tag{6}$$

where $x^{(i)}$ contains the clustering goal and indexed corpus, and $y^{(i)}_{\text{trace}}$ contains the retained reasoning trace and final answer. We train the student model with the standard autoregressive objective:

$$\mathcal{L}_{\text{warm}}(\theta) = -\sum_{(x,y)\in\mathcal{D}_{\text{warm}}} \sum_{t=1}^{|y|} \log p_\theta(y_t \mid x, y_{<t}). \tag{7}$$

D Format Constraints for Instruction-Following Clustering

This appendix provides a detailed explanation of the format constraints implemented in our instruction-following clustering framework. These constraints ensure that LRMs generate well-structured, parseable, and complete clustering solutions that adhere to our evaluation protocol.

Response Structure Requirements The output response should have exactly one occurrence each of `<think>`, `</think>`, `<answer>`, and `</answer>` tags. The correct tag ordering: `<think></think><answer></answer>`. No malformed, nested, or missing structural elements.

Clustering Solution Format Within the `<answer>` section, clustering solutions must adhere to the following canonical format:

```
Total clusters: [K]
cluster1: [i1, i2, i3, ...]
cluster2: [j1, j2, j3, ...]
...
clusterK: [m1, m2, m3, ...]
```

[5] https://huggingface.co/microsoft/Phi-4-reasoning

Reasoning Chain generation Prompt

Tasks
You are a clustering assistant to do text clustering.
Given a clustering goal and a list of indexed corpus (|1|. text, |2|. text, |3|. text...), cluster them following the think process below.
The clustering results should be in square brackets and use 1-based indexing for both total cluster and items (1, 2, 3, ...).

Think process of the task
First, read through all texts and think how can they be clustered based on the goal, determine the total number of clusters and each cluster's central meaning.
Then think about how to assign all texts into these clusters, ensure each text match the cluster's central meaning.
Finally, assign each text to a cluster based on the reasoning above.
Check the answer format before giving the final answer: every item must be assigned to exactly one cluster, and no item should appear in multiple clusters or be missing.

Goal: {instruction}
Text:
{enumerated_text}

Output format
First output your reasoning under `<think>` and `</think>`.
Then output the final clustering under `<answer>` and `</answer>`.
The final clustering must include the total number of clusters and a complete assignment of all item indices.

Figure 11: The system prompt for reasoning chain generation.

In the answer section, the response needs to start with "Total clusters: [N]" where N is the number of clusters. Then list each cluster as "cluster1:", "cluster2:", etc., with items in square brackets. Use 1-based indexing for items (1, 2, 3, ...).

Completeness and Consistency Validation ① The `Total clusters` line must be parseable. ② Cluster definitions must match the declared count. ③ All item indices from `1` to `N` (total items) must appear exactly once. ④ Duplicate, missing, or out-of-bound indices result in format errors. Violations of any of these rules are classified as format errors.

Cluster Count Consistency The declared cluster count K must match the actual number of clusters provided for the cluster. This prevents inconsistencies where models declare "Total clusters: [3]" but provide only two cluster assignments, ensuring alignment between high-level cluster count and actual cluster assignments.

E Baselines Implementation Details

E.1 Embedding-based Clustering Baselines

We evaluate several embedding-based clustering methods, following the official implementation for each baseline. For all models, a task-specific prompt is provided to guide the generation of embeddings tailored to the clustering objective. We detail the prompting strategies and implementation specifics below.

Instructor We follow the official implementation and parameterization. Instructor is an instruction-tuned embedding model: it takes as input a pair [instruction, text]. During embedding, the instruction is prepended to guide the semantic intent, e.g., "`Represent this text for clustering analysis: {clustering_instruction}`". For example, given the user instruction *"Cluster by research domain"*, the full input to the model for a given document is the pair: ["`Represent this text for clustering analysis: Cluster by`

Reasoning Chain Evaluation Prompt

You are an expert AI judge tasked with evaluating the quality and completeness of reasoning chains. Please evaluate the following reasoning chain based on these criteria:

1. **Completeness**: Does the reasoning chain generally provide a complete logical flow?
2. **Length Appropriateness**: Is the reasoning chain of appropriate length (not unnaturally short or abruptly cut off)?
3. **Logical Coherence**: Does the reasoning follow a logical sequence without gaps or jumps?

Reasoning Process of Clustering Task:
First, read through all texts and think how can they be clustered based on the goal, determine the total number of clusters and each cluster's central meaning.
Then think about how to assign all texts into these clusters, ensure each text match the cluster's central meaning.
Finally, assign each text to a cluster based on the reasoning above.

Reasoning Chain to Evaluate:
{reasoning_chain}

Response Format:
Provide your assessment in the following format:

```
ASSESSMENT: [ACCEPT/REJECT]
REASON: [Brief explanation of why you accepted or rejected this
reasoning chain]
```

Examples:

- **ACCEPT** if the reasoning is complete, logical, appropriately detailed, and reaches a proper conclusion
- **REJECT** if the reasoning is too short, incomplete, illogical, abruptly cut off, or doesn't properly address the question

Your assessment:

Figure 12: The system prompt for reasoning chain refinement.

`research domain`", "`The abstract discusses AI ethics...`"]. Embeddings are obtained via mean pooling over the final hidden states and are subsequently normalized.

InBedder We adopt the official configuration and implementation from InBedder (Peng et al., 2024). Following the default parameters, we use the recommended `CausalLMEncoder` backbone, `BrandonZYW/llama-2-7b-InBedder`. Generation hyperparameters are set to `temperature=0.6`, `top_p=0.9`, and `max_new_tokens=3`. Inputs are formatted using the official template: "`Input:{input} Instruction:{instruction} Response:`", where the placeholders are replaced with the raw text and clustering instruction. Embeddings are taken from the final hidden layer output and normalized prior to clustering.

Qwen Embedding Inputs for the Qwen embedding model follow the template "`Represent this text for clustering analysis. Task:{user_instruction} Query:{text}`", which symmetrically treats all texts as cluster candidates. Embeddings are extracted from the final hidden state corresponding to the end-of-sequence (EOS) token and normalized. The model supports input lengths of up to 8192 tokens.

OpenAI Embedding. We use OpenAI's `text-embedding-3-large` model to generate embeddings via the API. The input prompt concatenates the task description and text: "`Represent this text for clustering. Task: {user_instruction} Text: {text}`". For example: "`Represent this text for clustering. Task: Cluster by topic. Text: The news article covers politics...`". We use the largest embedding dimension of 3072.

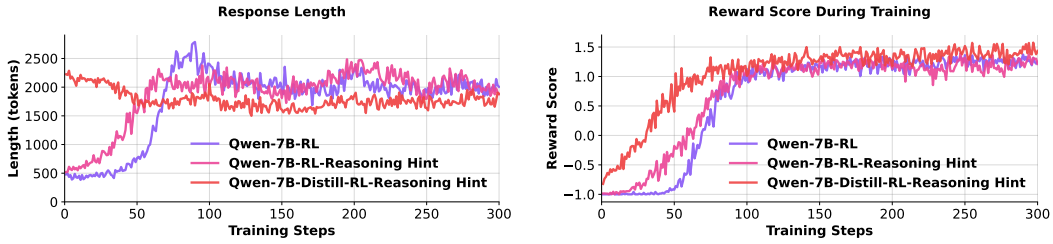


Figure 13: Training dynamics of the 7B model.

Sentence Transformers. For sentence-level baselines such as `all-MiniLM-L6-v2`, we prepend a simple task prefix: "`Clustering task: {user_instruction}. Text: {text}`". Embeddings are obtained via mean pooling and normalization.

Clustering and Pipeline. Clustering is performed using `scikit-learn` implementations of `KMeans` or `GaussianMixture`, with `random_state=43` and the ground-truth cluster count. Inputs exceeding 28k tokens are filtered using `tiktoken`.

E.2 LLMs BASELINES

To ensure a fair comparison between proprietary models and our proposed approaches, we provide a detailed specification of the required output format. This specification explicitly defines all formatting constraints, thereby guaranteeing that clustering results from both LRMs and LLMs baselines can be consistently and correctly parsed. In addition, we carefully designed and refined the answer-parsing functions to ensure robustness and accuracy during evaluation. The system prompt used for inference is illustrated in Figure 10.

F TRAINING CONFIG DETAILS

Our training framework is based on `verl` Sheng et al. (2024) and `trl` von Werra et al. (2020).

Distillation Stage. We conduct SFT over 510 distilled training examples. Each example is constructed by sampling 30 different text combination from the 17 held-in tasks (see Section 4.3) and then conduct distillation. We use a fixed batch size of 128 and a micro-batch size of 1, training for a single epoch. To optimize GPU memory usage, we enable gradient checkpointing, FlashAttention, and Adam offloading. The learning rates are set based on the model size: $2e{-5}$, $1e{-5}$, and $3e{-6}$ for models of size 3B, 7B, and 14B, respectively.

RL Stage. We perform GRPO optimization over 8.5k training examples, where each example is constructed by sampling text from the 17 held-in tasks (see Section 4.3). Training is conducted with a global batch size of 32 and a mini-batch size of 128. We adopt Fully Sharded Data Parallel (FSDP) training. Rollouts are generated using `vLLM` with a tensor parallelism degree of 4, and GPU memory utilization capped at 50%. Sampling follows the default decoding configuration (temperature = 1.0, top-p = 1.0). KL regularization is applied with a coefficient of 1×10^{-3} and a clipping ratio of 0.2. For each prompt, we sample 5 candidate rollout responses. The maximum input and output sequence lengths are set to 28,000 and 4,000 tokens, respectively. Learning rates are tuned by model scale: 1×10^{-6} for the 3B and 7B models, and 5×10^{-7} for the 14B model. We train the 3B, 7B, and 14B variants on 1, 1, and 2 nodes, respectively, with each node equipped with 8 GPUs.

G V-MEASURE METRIC

The V-measure (Rosenberg & Hirschberg, 2007) is an external clustering evaluation metric based on conditional entropy. It quantifies the agreement between a predicted clustering $\hat{C} = \{\hat{C}_1, \ldots, \hat{C}_k\}$ and a gold standard partition $C^* = \{C^*_1, \ldots, C^*_m\}$ in terms of two complementary criteria:

Bin (n)	V-meas		V-valid		Fail Rate	
	GPT-4.1	C1-Qwen	GPT-4.1	C1-Qwen	GPT-4.1	C1-Qwen
6–10	0.620±0.019	**0.766±0.011**	0.683±0.012	**0.769±0.010**	9.2%±2.0	**0.4%±0.5**
11–15	0.493±0.026	**0.714±0.016**	0.614±0.018	**0.719±0.016**	19.7%±3.4	**0.7%±0.6**
16–20	0.401±0.026	**0.689±0.019**	0.551±0.019	**0.693±0.017**	27.1%±4.3	**0.5%±0.8**
21–25	0.342±0.020	**0.664±0.019**	0.459±0.016	**0.674±0.015**	25.6%±3.5	**1.4%±1.2**
26–30	0.258±0.014	**0.631±0.010**	0.398±0.018	**0.643±0.009**	35.1%±3.4	**1.9%±1.0**
>30	0.214±0.009	**0.522±0.017**	0.354±0.011	**0.669±0.009**	39.6%±2.4	**21.9%±2.3**

Table 6: Scaling analysis by number of items per instance (n). V-meas: V-measure over all instances. V-valid: V-measure over valid predictions only. Fail: format failure rate.

Bin (K^*)	V-meas		V-valid		Fail Rate	
	GPT-4.1	C1-Qwen	GPT-4.1	C1-Qwen	GPT-4.1	C1-Qwen
3	0.436±0.023	**0.685±0.015**	0.547±0.016	**0.685±0.015**	20.4%±3.6	**0.0%±0.0**
4	0.462±0.020	**0.689±0.021**	0.564±0.021	**0.693±0.020**	18.0%±3.6	**0.5%±0.6**
5	0.351±0.026	**0.660±0.017**	0.517±0.028	**0.673±0.017**	32.1%±3.5	**2.0%±1.3**
6	0.410±0.022	**0.707±0.011**	0.543±0.018	**0.710±0.011**	24.4%±3.2	**0.3%±0.5**
7	0.391±0.019	**0.732±0.011**	0.546±0.013	**0.748±0.009**	28.5%±3.1	**2.1%±1.1**
≥8	0.336±0.016	**0.603±0.014**	0.537±0.012	**0.771±0.004**	37.4%±2.4	**21.8%±1.7**

Table 7: Scaling analysis by true cluster count (K^*). See Table 6 for metric definitions.

- **Homogeneity** (h): each predicted cluster should contain only members of a single gold class.
- **Completeness** (c): all members of a given gold class should be assigned to the same predicted cluster.

Formally, let $H(C^*)$ denote the entropy of the gold labels and $H(C^* \mid \hat{C})$ the conditional entropy given the predicted clustering. Homogeneity and completeness are defined as:

$$h = \begin{cases} 1, & \text{if } H(C^*) = 0, \\ 1 - \dfrac{H(C^* \mid \hat{C})}{H(C^*)}, & \text{otherwise.} \end{cases} \tag{8}$$

$$c = \begin{cases} 1, & \text{if } H(\hat{C}) = 0, \\ 1 - \dfrac{H(\hat{C} \mid C^*)}{H(\hat{C})}, & \text{otherwise.} \end{cases} \tag{9}$$

The V-measure is then given by the harmonic mean:

$$V = \frac{2 \cdot h \cdot c}{h + c}. \tag{10}$$

This symmetric formulation ensures that a high V is achieved only when clusters are both internally consistent (high h) and well aligned with the gold classes (high c). Values range from 0 (no agreement) to 1 (perfect match).

H Details of Data Pre-processing

H.1 LMSYS-Chat

Current intent-discovery benchmarks such as Bank77 and CLINC (Li et al., 2021; Casanueva et al., 2020a; Larson et al., 2019b; FitzGerald et al., 2022) primarily consist of short, single-turn utterances associated with narrowly defined intents. While useful for intent classification, these datasets fail to capture the complexity of real-world interactions, where dialogues are longer, context-dependent,

Bin (L)	V-meas		V-valid		Fail Rate	
	GPT-4.1	C1-Qwen	GPT-4.1	C1-Qwen	GPT-4.1	C1-Qwen
<20k	0.543±0.022	**0.688±0.011**	0.619±0.014	**0.688±0.011**	12.4%±2.8	**0.0%±0.0**
20k–40k	0.460±0.025	**0.703±0.016**	0.556±0.017	**0.708±0.015**	17.3%±3.9	**0.6%±0.7**
40k–60k	0.368±0.026	**0.692±0.020**	0.522±0.020	**0.703±0.019**	29.4%±4.2	**1.5%±1.1**
60k–80k	0.334±0.023	**0.663±0.014**	0.512±0.019	**0.683±0.010**	34.9%±3.1	**2.8%±1.3**
≥80k	0.308±0.030	**0.613±0.020**	0.515±0.024	**0.679±0.011**	40.2%±4.5	**9.6%±2.8**

Table 8: Scaling analysis by input prompt length (L, in characters). See Table 6 for metric definitions.

Dataset	Split	Prompt
ECHR	Held-in	Cluster these legal cases by the primary vulnerable population involved. Cluster these legal cases by the dominant type of domestic proceedings that preceded the European Court application. Cluster these legal cases by the violation articles invoked in the Court's judgment.
	Held-out	Cluster the following texts based on the primary underlying category of legal dispute or issue. Cluster the following texts based on the main types of parties or entities involved as adversaries or plaintiffs. Cluster the following texts based on the timing, duration, or current stage of resolution for the described legal proceedings.
LMSY	Held-in	Cluster the following user requests based on explanation audience level. Cluster these requests by temporal regime they focus on. Cluster the dialogue by identifying the root cause of the issue. Cluster the dialogue based on the type of transformation needed to convert it into a different form or function. Cluster these creative generation requests by content type. Group each dialogue according to the primary social function or interpersonal purpose it serves. Cluster these decision support requests based on the structure and format of the solution being requested. Group these recommendation requests by identifying the type of justification used. Cluster the dialogues by the distinct perspective or evaluative role the agent adopts. Group the conversations based on the type of structured planning or scheduling being discussed.
	Held-out	Group these computational tasks by archetype. Cluster these data analysis requests by tooling context. Cluster these texts based on the kind of external context or disambiguation needed. Group these factual lookup requests by the structure and format of the expected output. Cluster these factual lookup requests based on the type of source or origin of information.
SP500	Held-in	Cluster these SP500 companies by their asset ownership and utilization patterns. Cluster these SP500 companies by their geographic distribution of assets and operations. Cluster these SP500 companies by their industry-specific asset composition. Cluster these SP500 companies by their risk exposure and asset encumbrance patterns.
	Held-out	Cluster the texts based on the strategies companies use for distributing earnings to shareholders via dividends. Cluster the texts based on the approaches companies take toward buying back their own shares. Cluster the texts based on the varieties of equity instruments issued by companies.

Table 9: Clustering instructions for both held-in and held-out tasks from each dataset.

and often involve nuanced shifts in user goals. To address this gap, we leverage the LMSYS-Chat-1M corpus (Zheng et al., 2023), which contains over one million real-world conversations collected from more than 210,000 unique users through the Vicuna demo and Chatbot Arena (Zheng

LMSYS-Chat Data Example

User: What features are relevant in a visualization of a streaming platform? (From a business intelligence viewpoint)

Assistant: When it comes to visualizing a streaming platform from a business intelligence viewpoint, several key features are relevant:
1. *User engagement* — number of active users, average time spent, number of streams viewed. 2. *Content distribution* — number of channels/streams, countries/regions available, devices/platforms supported. 3. *Revenue and monetization* — number of paying subscribers, ARPU, ad views/clicks. 4. *User acquisition and retention* — cost of acquiring a user, number of new users, retention rates. 5. *Technical performance* — uptime, load times, outages/issues. These visualizations provide insights into platform performance and growth, enabling informed decision-making.

User: Relevant features when comparing content streaming platforms, such as Netflix vs Disney+

Assistant: When comparing platforms like Netflix and Disney+, the following features matter:
1. *User engagement* — active users, time spent, streams viewed. 2. *Content library* — total titles, originals, exclusives. 3. *Geographic coverage* — countries/regions available, supported devices/platforms. 4. *Pricing and monetization* — pricing strategy, subscribers, ARPU, ad revenues. 5. *Marketing and promotion* — budget, partnerships, user acquisition. 6. *Technical performance* — uptime, stream load time, outages.
These comparisons yield insights into competitiveness and positioning in the market.

Figure 14: LMSYS-Chat data example.

et al.). For our benchmark, we focus on the first 500,000 English-language dialogues and apply rigorous filtering to exclude unsafe or low-quality content. Although multi-turn conversations better reflect natural dialogue, they frequently exhibit shifting or overlapping intents that complicate cluster assignment. Therefore, we restrict our selection to one- and two-turn exchanges, preserving conversational richness while ensuring clearer intent detection. Distinct from prior intent-clustering datasets that focus solely on user utterances, we also incorporate the assistant's responses into the clustering process. For example, we introduce clustering dimensions based on expected output formats, thereby capturing a broader range of realistic use cases beyond user intent alone. This dual perspective acknowledges that effective clustering in human–AI dialogue requires jointly modeling user communicative goals and the structural or stylistic properties of system replies. Given the scale of the corpus, we implement our multi-agent pipeline hierarchically. In the first stage, the pipeline groups dialogues into 12 coarse-grained intent clusters. In the second stage, each cluster is further subdivided into multiple sub-clusters, yielding a more granular taxonomy of conversational patterns. This hierarchical strategy balances scalability with interpretability and supports downstream applications in dialogue understanding and controllable generation. An illustrative example is shown in Figure 14.

H.2 S&P 500 Financial Reports

To broaden coverage beyond conversational and legal texts, we include financial documents in the form of public company annual reports (Form 10-K filings). Specifically, we leverage the SP500-EDGAR-10K dataset[6], which contains reports for all historical S&P 500 constituents from 2010 to 2022 U.S. Securities and Exchange Commission (n.d.). Each report consists of multiple structured items (e.g., Business, Risk Factors, Management's Discussion & Analysis, Market Risk Disclosures, Financial Statements). To ensure compatibility with context-length constraints during model training and inference, we select a subset of SP500 financial filings in which the length of each text item remains within a tractable range. This allows multiple document segments to be processed jointly without exceeding the model's input limits. We focus on three representative items from the 10-K filings:

- **Item 2 – Properties:** Describes corporate assets such as manufacturing plants, distribution centers, and office facilities, including their location, usage, and ownership status.

[6] https://huggingface.co/datasets/jlohding/sp500-edgar-10k

SP500 Data Example

Item: 2 – Properties

General Overview: - Company and subsidiaries owned or leased material facilities (over 50,000 sq. ft.) in 16 U.S. states and 14 foreign countries. - Facilities used for manufacturing, distribution, and sales offices. - Properties used by multiple segments are attributed to the segment with highest usage. - Facilities considered suitable and adequate for business.

Owned Material Facilities: - *Security:* Farmington (CT); Sterling, Rock Falls (IL); Indianapolis (IN); Nicholasville (KY); Richmond (VA); Cobourg (Canada); Nueva Leon (Mexico); Xiaolan (China). - *Industrial:* Phoenix (AZ); Two Harbors (MN); Columbus, Georgetown, Sabina (OH); Allentown (PA); Dallas (TX); Pecky (Czech Rep.); Epernay, Ezy Sur Eure, Feuquieres en Vimeu, Morangis, Villeneuve Le Roi (France); Fano, Gemonio, Monvalle (Italy). - *CDIY:* Clinton, New Britain (CT); Shelbyville (IN); East Greenwich (RI); Cheraw (SC); Smiths Falls (Canada); Hellaby, Northampton (England); Arbois, Besancon Cedex, Lassiey (France); Puebla (Mexico); Jiashan City, Langfang, Xiaolan (China); Wroclaw (Poland); Taichung Hsien (Taiwan); Amphur Bangpakong (Thailand).

Leased Material Facilities: - *Corporate Offices:* New Britain (CT). - *Security:* Noblesville (IN); Kentwood (MI). - *Industrial:* Highland Heights, Westerville (OH); Milwaukie (OR); Morangis (France). - *CDIY:* Miramar (FL); Fishers (IN); Kannapolis (NC); Epping (Australia); Mechelen (Belgium); Oakville (Canada); Leeds (England); Karmiel, Migdal (Israel); Biassono, Figino Serenza (Italy); Pietermaritzburg (South Africa).

Facilities Not in Use: - *Security:* Richmond (VA, owned). - *Industrial:* Ezy Sur Eure, Villeneuve Le Roi (France, owned). - *CDIY:* Clinton (CT, owned); Smiths Falls (Canada, owned); Leeds (England, leased); Amphur Bangpakong (Thailand, one property owned).

Figure 15: Example of Item 2 ("Properties") from SP500 dataset.

- **Item 3 – Legal Proceedings:** Summarizes ongoing or pending litigation and regulatory investigations. By aligning structurally with ECHR case law—though differing substantially in domain—this item supports evaluation of cross-domain generalization in legal clustering tasks.

- **Item 5 – Market for Registrant's Common Equity:** Covers stock exchange listings, price histories, dividends, and shareholder matters. Including this item enables assessment of model robustness when transferred to financial topics unrelated to legal reasoning.

An example of Item 2 ("Properties") is presented in Figure 15.

H.3 ECHR

We incorporate case law documents from the European Court of Human Rights (ECHR) (Chalkidis et al., 2019), which introduce long-form texts that pose distinct reasoning challenges compared to conversational data. Each case typically comprises: (i) a facts section organized into numbered paragraphs summarizing events and evidence; (ii) the alleged violations of the European Convention on Human Rights; and (iii) the court's decision indicating which allegations are upheld or dismissed. We use an enriched version of the publicly available ECHR corpus containing approximately 11,000 cases. To ensure data quality and consistency, we exclude non-English cases and those lacking sufficient information across the core components (facts, articles, or decisions). To facilitate effective clustering based on legal reasoning rather than superficial features, we preprocess each case from the ECHR corpus by removing metadata fields that are irrelevant to the core legal content, such as "Importance Level" and "Judges." We also omit the conclusion and the explicitly stated violated articles. This ensures that models must engage in substantive understanding and normative reasoning to infer which articles of the European Convention on Human Rights may have been violated. Clustering is then performed using the instruction: *"Cluster legal cases by the violated article."* An illustrative example is shown in Figure 16, which presents a structured summary of the case *Schvarc v. Slovakia* (2014) prior to redaction. The "Facts" section outlines the legal proceedings, while conclusion and violated articles are removed during training to simulate realistic inference conditions.

ECHR Data Example

Case Name: CASE OF SCHVARC v. SLOVAKIA **Item ID:** 001-139999 **Respondent:** Slovakia (SVK) **Court Branch:** Chamber **Date:** 2014 **Judges:** Alvina Gyulumyan; Ján Šikuta; Johannes Silvis; Josep Casadevall; Kristina Pardalos; Luis López Guerra **Importance Level:** 4

Facts: The applicant, born in 1978, resides in Krupina. Arrested, charged, and remanded in custody on 20–22 May 2009 for drug-related offences under Article 172 §1(c) of the Criminal Code. On 22 July 2009, he requested release, pledging lawful conduct and non-interference with justice. On 3 August 2009, the District Prosecutor recommended dismissal; the Zvolen District Court dismissed the request on 19 August 2009, citing a prior theft conviction. The applicant appealed orally on 19 August, later amending with legal assistance. Written decision served on 24 September 2009. On 8 October 2009, the Banská Bystrica Regional Court dismissed the appeal; written version served on 30 October 2009. On 11 November 2009, the applicant lodged a complaint with the Constitutional Court under Article 127 of the Constitution and Article 5 §4 of the Convention, claiming €1,500. The Constitutional Court admitted the complaint on 18 February 2010 and ruled on 30 March 2010, finding a violation of the right to speedy review and awarding €500 compensation. Written judgment served on 24 May 2010.

Conclusion: - *Violation of Article 5* – Right to liberty and security (Article 5-4: Speediness of review).

Violated Articles/Paragraphs: - Article 5 (para. 4)

Figure 16: ECHR data example.

H.4 Benchmark Details

In Table 9, we present the clustering instructions for both held-in and held-out tasks across each dataset.

I Discussion: Instruction-Following Clustering vs. Classification

Several aspects of ReasonCluster resemble classification at the benchmark construction stage, because taxonomies are used to define reliable reference partitions. We clarify here that this taxonomy is an annotation instrument for constructing evaluation labels, not part of the model input at inference time. During inference, an evaluated model receives only a natural-language clustering instruction and an unlabeled corpus. It is not given candidate category names, taxonomy descriptions, cluster labels, or the gold number of clusters.

Benchmark construction uses taxonomies only to create gold partitions. The data synthesis pipeline first proposes a clustering dimension and an associated taxonomy, refines the taxonomy for mutual exclusivity and consistent granularity, and then assigns documents to categories through consensus labeling. These categories define the gold partition used for training and post hoc evaluation with external clustering metrics such as V-measure. In this sense, the taxonomy plays the same role as class labels in many standard clustering benchmarks: it provides a reference partition against which an induced clustering can be measured. It does not imply that the evaluated system performs classification into a provided label set.

Inference is taxonomy-free and label-free. At test time, the model input contains the instruction and the indexed texts only. The model must infer which textual property the instruction makes relevant, decide how many groups are needed, and assign every item to exactly one group. For example, the same legal corpus could be organized by violated article, vulnerable population, proceeding type, or factual circumstance, depending on the instruction. These different instructions induce different valid partitions over the same documents. Because no candidate ontology is supplied, the model cannot solve the task by selecting among predefined labels; it must induce the partition structure from the instruction and corpus.

Difference from structured classification. Structured classification typically assumes a fixed output ontology that is known before prediction, and each item is assigned to one of those known

classes. REASONCLUSTER differs in three ways. First, the ontology is task-dependent: the relevant grouping criterion changes with the instruction. Second, the number of groups is not supplied to the model and must be inferred. Third, the output is a corpus-level partition rather than independent per-document labels, so the model must maintain global consistency across all assignments. These properties make the task closer to instruction-conditioned latent structure induction than to ordinary classification.

Training is supervised, but the inference task remains clustering. Our training recipe uses partition-level supervision. The RL reward uses both the gold cluster count and V-measure against the gold partition. Therefore, the method should not be interpreted as a classical unsupervised clustering algorithm. The intended claim is narrower: supervised post-training can teach an LLM to perform taxonomy-free, label-free clustering inference under natural-language instructions. This distinction is especially important for the C0 and C1 splits, where instructions are seen during training. C0 and C1 primarily test recombination and new-text generalization under familiar grouping criteria, while C2 tests transfer to unseen instructions, clusters, and texts.

J ANALYSIS OF MAIN RESULTS

J.1 FORMAT ERRROR ANALYSIS

As shown in Figure 17, models with higher format accuracy consistently achieve higher V-measure scores, highlighting format correctness as a fundamental prerequisite for clustering quality. This correlation arises because an incorrectly formatted response that cannot be parsed receives a V-measure score of 0. Consequently, general-purpose language models often underperform, as they frequently violate the required output format. Nevertheless, high format accuracy alone does not guarantee superior clustering performance. For instance, models such as Llama-3.1-70B-Instruct exhibit high format accuracy but still attain low V-measure scores. This discrepancy suggests that while high format accuracy ensures evaluability, effective clustering further depends on a model's capacity for semantic reasoning and clustering instruction alignment. Mastering the syntax of clustering outputs is necessary but not sufficient. The model must also internalize the underlying reasoning process that drives coherent and instruction-faithful cluster assignment.

J.2 INPUT LENGTH SCALING ANALYSIS

Tables 6, 7, and 8 report detailed results across three input complexity dimensions: the number of items per instance (n), the true cluster count (K^*), and the input prompt length (L). For each bin, we report overall V-measure, V-measure conditioned on valid outputs, and format failure rate. Across all three dimensions, a consistent pattern emerges. As input complexity increases, both models show lower clustering performance and higher format failure rates. However, the degradation is substantially more severe for GPT-4.1 than for C1-Qwen.

Effect of number of items per instance. As n increases, GPT-4.1 degrades sharply, with V-measure dropping from 0.62 to 0.21 and failure rate rising from about 9% to nearly 40%. In contrast, C1-Qwen maintains substantially stronger performance, with V-measure decreasing more gradually from 0.77 to 0.52. Its failure rate remains close to zero for most bins and increases only in the most difficult setting. Importantly, even when conditioning on valid outputs, C1-Qwen consistently outperforms GPT-4.1, indicating that the advantage cannot be explained solely by better format compliance.

Effect of cluster count. Increasing K^* also makes the task more difficult for both models. GPT-4.1 shows unstable performance and rising failure rates as the number of clusters grows, especially when $K^* \geq 8$. C1-Qwen remains comparatively stable for moderate values of K^* and degrades noticeably only in the largest-cluster setting. Since larger K^* does not necessarily imply longer input sequences, this result suggests that the difficulty is not purely driven by context length, but also by the combinatorial challenge of assigning items across more clusters.

Effect of prompt length. Longer prompts further amplify the difference between the two models. GPT-4.1 shows a steady decline in V-measure together with a substantial rise in format failures, exceeding 40% in the longest-length bin. By contrast, C1-Qwen maintains strong performance and

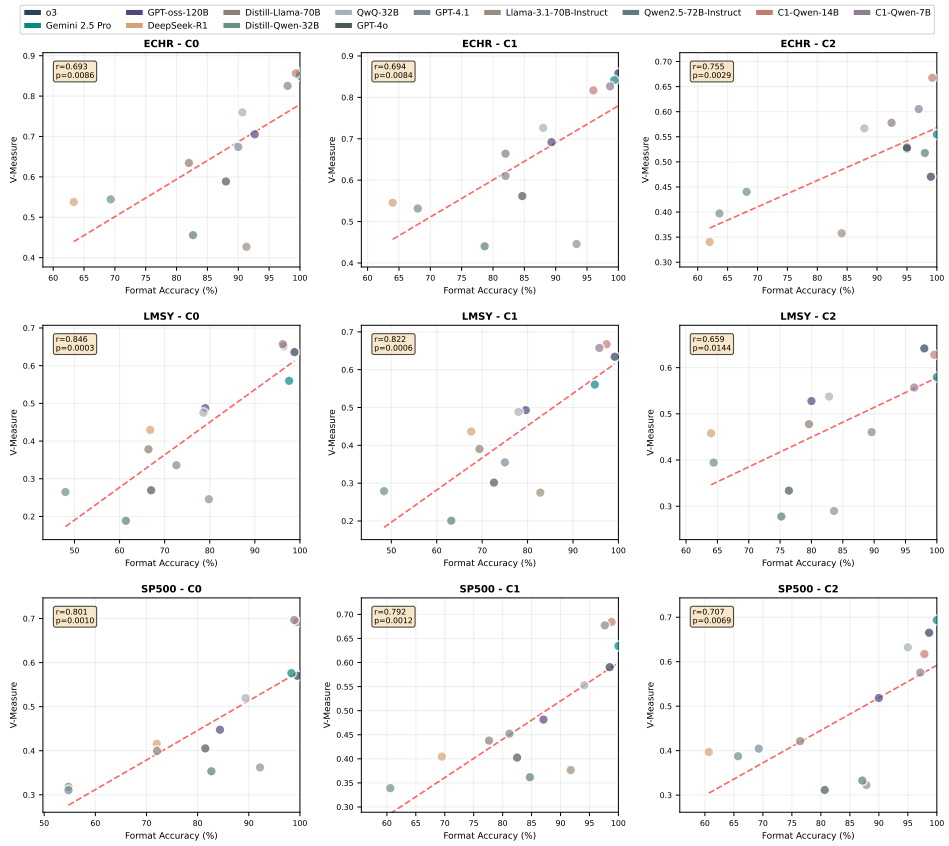


Figure 17: Relationship between format accuracy and V-measure score. Models with higher format accuracy consistently achieve higher V-measure scores, highlighting format correctness as a fundamental prerequisite for clustering quality. Nevertheless, high format accuracy alone does not guarantee superior clustering performance. For instance, models such as Llama-3.1-70B-Instruct exhibit high format accuracy but still attain low V-measure scores. Effective clustering further depends on a model's capacity for semantic reasoning and clustering instruction alignment. Mastering the syntax of clustering outputs is necessary but not sufficient.

near-zero failure rates up to moderately long prompts, with degradation appearing only at the largest lengths.

J.3 Failure Case

Figure 19 illustrates a representative failure case where the clustering agent exhibits brittle reasoning and unstable self-revision. The instruction is to group ECHR cases by the primary vulnerable population. The agent initially proposes sensible, mutually exclusive categories and begins assigning texts accordingly. During a later "reflection" pass, however, it merges Children with a broader bucket (Victims of discrimination based on social status) and then continues to use Children as a standalone label in assignments. This revision creates overlapping categories and internal inconsistency: Children is both a subtype inside a broad "Vulnerable individuals" bucket and a separate top-level cluster used for item assignment. One possible reason is our current reinforcement objective optimizes final-format validity and end-state clustering quality, but it does not explicitly reward the process of reasoning. Concretely, the reward is the sum of a format term and a clustering term with count accuracy and V-measure, without intermediate checks on reasoning steps or on constraints like mutual exclusivity and consistent granularity. In this case, the agent's "reflection" collapses distinct categories, introduces overlap, and then assigns items inconsistently.

J.4 CASE STUDY

Figure 18 illustrates a representative success case of our clustering agent on the ECHR dataset, highlighting the model's ability to perform interpretable reasoning while maintaining a structural output format. In this example, the model first infers an appropriate number of clusters and articulates a clear rationale for each, corresponding to distinct categories of human rights violations such as Right to Privacy, Protection of Property, Freedom of Expression, and Right to a Fair Trial. The reasoning sequence reveals a deliberate, hierarchical decision process in which the model identifies the underlying legal principle in each text segment before assigning it to a cluster. The resulting clusters are both logically coherent and legally interpretable, demonstrating that the model not only adheres to output format conventions but also internalizes domain-specific reasoning patterns. This example shows reasoning supervision enables models to construct clusters that are semantically meaningful, structurally consistent, and faithful to the task instruction.

Clustering Agent Response Example (Full Reasoning and Answer)

First, I will read through all texts to understand the core legal issue in each case, keeping the goal in mind: "Cluster these legal cases by the violation articles invoked in the Court's judgment." I need to identify the specific right or principle of the European Convention on Human Rights that is at stake. Let's break down each text:

- **Text 1 (JAEGER v. ESTONIA):** The applicant was searched in a public area, and the door was transparent. This is a clear issue of privacy and dignity under Article 8 (Right to respect for private and family life).
- **Text 2 (ALTIN AND OTHERS v. TURKEY):** The applicants were unable to get a judgment enforced. This is a problem with the right to a fair trial and the right to peaceful enjoyment of possessions (the judgment award). It falls under Article 1 of Protocol No. 1 (Protection of property) and Article 13 (Right to an effective remedy).
- **Text 3 (Massimo ROSA v. ITALY):** The applicant was unable to get a court order enforced to evict a tenant. This is another case about the right to peaceful enjoyment of possessions under Article 1 of Protocol No. 1.
- **Text 4 (STREHAR v. SLOVENIA):** The case concerns the length of pre-trial detention. This is a classic Article 5 (Right to liberty) issue.
- **Text 5 (GAWEDA v. POLAND):** The government refused to register a periodical based on its title. This is a restriction on freedom of the press under Article 10 (Freedom of expression).
- **Text 6 (UNSAL OZTURK v. TURKEY):** The applicant's books were confiscated based on their content. This is another case concerning freedom of the press under Article 10.
- **Text 7 (UNION EUROPEENNE DES DROITS DE L'HOMME AND JOSEPHIDES v. TURKEY):** Security forces seized the applicants' flag and confiscated their materials during a peaceful demonstration. This is an interference with the right to peaceful assembly under Article 11.
- **Text 8 (SIVERIN v. RUSSIA):** The applicants' judgment for benefits was not enforced. Like text 2, this is another Article 1 of Protocol No. 1 (Protection of property) and Article 13 issue.
- **Text 9 (KHALILOVA AND AYYUBZADE v. AZERBAIJAN):** The applicants were arrested and prosecuted for participating in a peaceful protest. This is another Article 11 (Freedom of assembly) issue.
- **Text 10 (SZEMELY ES VAGYONOROK FUGGETLEN SZAKSZERVEZETI SZOVETSEGE v. HUNGARY):** A trade union was refused registration. This is another case concerning freedom of association under Article 11.
- **Text 11 (DURMUS v. TURKEY):** A teacher was sanctioned for distributing a union notice. This is another Article 11 issue, concerning the freedom of association.
- **Text 12 (CLOOTH v. BELGIUM):** The applicant's claim for compensation was dismissed. The issue is the right to an effective remedy under Article 13.
- **Text 13 (T.M. AND OTHERS v. RUSSIA):** The applicants face removal to Uzbekistan where they fear ill-treatment. This is an Article 3 (Prohibition of inhuman or degrading treatment) issue.
- **Text 14 (PEREE v. THE NETHERLANDS):** The applicant was prosecuted for statements made in a political interview. This is another Article 10 (Freedom of expression) issue.
- **Text 15 (NIEMINEN v. FINLAND):** The police entered the applicant's home to investigate a crime. This is another Article 8 issue concerning the right to privacy.

- **Text 16 (ALBAYRAK v. TURKEY):** The applicant was transferred to a lower court for trial. The justification was his "undermine[d] the honour and dignity of the judiciary." This concerns the right to a fair trial under Article 2 (Right to a fair trial).
- **Text 17 (DEMIR v. FINLAND):** The applicant was expelled, separating him from his family. This is another Article 8 issue concerning the right to respect for his private life and family life.
- **Text 18 (NIYITEGURE v. SWEDEN):** The applicant fears persecution if deported to Burundi and seeks asylum. This is another Article 3 issue concerning the risk of inhuman or degrading treatment.

Based on this analysis, I can identify several distinct categories of legal issues:

1. **Privacy and Dignity:** Cases involving searches or invasions of personal space.
2. **Property and Compensation:** Cases where the state fails to protect property or enforce judgments.
3. **Freedom of Expression:** Cases concerning restrictions on the press or speech.
4. **Freedom of Assembly:** Cases concerning the right to protest.
5. **Freedom of Association:** Cases concerning the right to form or join organizations.
6. **Right to a Fair Trial:** Cases concerning procedural fairness in criminal proceedings.
7. **Right to Prohibition of Inhuman or Degrading Treatment:** Cases concerning the risk of ill-treatment.

This suggests a total of 7 clusters. I will define them as follows:

- **Cluster 1: Right to Privacy and Dignity (Article 8).** Cases involving searches or other state intrusions into personal space.
- **Cluster 2: Protection of Property and Effective Remedy (Article 1 of P1-1 & 13).** Cases about the state's failure to protect assets or provide a proper remedy.
- **Cluster 3: Freedom of Expression (Article 10).** Cases concerning restrictions on media or speech.
- **Cluster 4: Freedom of Assembly (Article 11).** Cases concerning the right to protest.
- **Cluster 5: Freedom of Association (Article 11).** Cases concerning the right to join organizations.
- **Cluster 6: Right to a Fair Trial (Article 2).** Cases concerning the fairness of legal proceedings.
- **Cluster 7: Prohibition of Inhuman or Degrading Treatment (Article 3).** Cases concerning the risk of ill-treatment.

Now, I will go through each text and assign it to one of these clusters based on my reasoning above.

- Text 1 → Issue is privacy/dignity → **Cluster 1**.
- Text 2 → Issue is property/compensation enforcement → **Cluster 2**.
- Text 3 → Issue is property/compensation enforcement → **Cluster 2**.
- Text 4 → Issue is liberty/pre-trial detention → **Cluster 6**.
- Text 5 → Issue is freedom of expression (press) → **Cluster 3**.
- Text 6 → Issue is freedom of expression (press) → **Cluster 3**.
- Text 7 → Issue is freedom of assembly → **Cluster 4**.
- Text 8 → Issue is property/compensation enforcement → **Cluster 2**.
- Text 9 → Issue is freedom of assembly → **Cluster 4**.
- Text 10 → Issue is freedom of association → **Cluster 5**.
- Text 11 → Issue is freedom of association → **Cluster 5**.
- Text 12 → Issue is lack of an effective remedy → **Cluster 2**.
- Text 13 → Issue is prohibition of inhuman/degrading treatment → **Cluster 7**.
- Text 14 → Issue is freedom of expression (speech) → **Cluster 3**.
- Text 15 → Issue is privacy/home search → **Cluster 1**.
- Text 16 → Issue is fairness of trial (judicial integrity) → **Cluster 6**.

- Text 17 → Issue is prohibition of inhuman/degrading treatment (expulsion) → **Cluster 7**.
- Text 18 → Issue is prohibition of inhuman/degrading treatment (deportation) → **Cluster 7**.

The assignments are complete. I will now format the final answer.

```
<answer>
Total clusters:  [7]
cluster1:  [1, 15]
cluster2:  [2, 3, 8, 12]
cluster3:  [5, 6, 14]
cluster4:  [7, 9]
cluster5:  [10, 11]
cluster6:  [4, 16]
cluster7:  [13, 17, 18]
</answer>
```

Figure 18: An illustrative example of the clustering agent's full reasoning and final structured output for ECHR legal cases clustered by violation articles.

Clustering Agent Failure Case

First, I will read through all texts to understand the nature of the cases and identify potential patterns or themes. The goal is to cluster them by the **"primary vulnerable population involved."** This means I need to identify the specific demographic or social group that faces heightened human rights risks in each case. After reading through the texts, I can identify several potential clusters:

- **Children:** Cases where the vulnerable population is a child.
- **Journalists:** Cases where the vulnerable population is a journalist.
- **People with disabilities:** Cases where the vulnerable population is a person with a disability.
- **Elderly people:** Cases where the vulnerable population is an elderly person.
- **Victims of domestic violence or sexual abuse:** Cases where the vulnerable population is a victim of domestic violence or sexual abuse.
- **Victims of discrimination based on social status:** Cases where the vulnerable population is a person facing discrimination based on social or economic status (e.g., victims of domestic conflict, low-income individuals).

Next, I will go through each text and assign it to one of these clusters.

- **Text 1:** The applicant is a 19-year-old woman seeking social welfare assistance. → This case involves a young person (a child) who is vulnerable due to her age and economic situation.
- **Text 2:** The applicant is a journalist convicted for publishing a newspaper. → This case involves a journalist.
- **...**

Based on this analysis, I can define the following clusters:

1. **Children:** Cases where the vulnerable population is a child.
2. **Journalists:** Cases where the vulnerable population is a journalist.
3. **People with disabilities:** Cases where the vulnerable population is a person with a disability.
4. **Elderly people:** Cases where the vulnerable population is an elderly person.
5. **Victims of domestic violence or sexual abuse:** Cases where the vulnerable population is a victim of domestic violence or sexual abuse.
6. **Victims of discrimination based on social status:** Cases where the vulnerable population is a person facing discrimination based on social status (e.g., victims of domestic conflict or poverty).

However, after re-evaluating the texts, I realize that text 4 (a woman who receives welfare payments for her child) and text 11 (a woman living with her daughter in indecent and insalubrious conditions) fit better into the cluster of "Victims of discrimination based on social status" rather than "Children." Therefore, I will merge the "Children" cluster and the "Victims of discrimination based on social status" cluster into a single cluster called "Vulnerable individuals (children, elderly, or victims of discrimination based on social status)."

This gives me five clusters:

- **Children:** Cases where the vulnerable population is a child.
- **Journalists:** Cases where the vulnerable population is a journalist.
- **People with disabilities:** Cases where the vulnerable population is a person with a disability.
- **Elderly people:** Cases where the vulnerable population is an elderly person.
- **Vulnerable individuals (children, elderly, or victims of discrimination based on social status):** Cases where the vulnerable population is a child, an elderly person, or someone discriminated against due to social or economic status.

(Note: During the "reflection" process, the model merges Children with a broader bucket (Victims of discrimination based on social status) and then continues to use Children as a standalone label in assignments.)

Figure 19: Example of a failure case where the clustering agent exhibits brittle reasoning and unstable self-revision.