SKILLVERSE : Assessing and Enhancing LLMs with Tree Evaluation

Anonymous ACL submission

Abstract

As language models evolve to tackle complex, multifaceted tasks, their evaluation must adapt to capture this intricacy. A granular, skill-specific understanding of model capabilities can empower researchers to make informed model development plans. In this paper, we introduce SKILLVERSE, an unsupervised tree-structured diagnosis framework for understanding model proficiency in specific abilities. With LLM as a judge, SKILLVERSE first critiques the model responses, and then organizes them into a hierarchical structure termed dendrogram. Given proficiency at arbitrary levels of granularity, SKILLVERSE is flexible to produce insights of behaviors of modern large models. We also demonstrate its efficacy in two downstream tasks: 1) improving model in-context learning by 25% using a tree-search algorithm to select more informative few shots, and 2) accurately predicting new model weaknesses with a 55% success rate, 22% higher than the baseline.

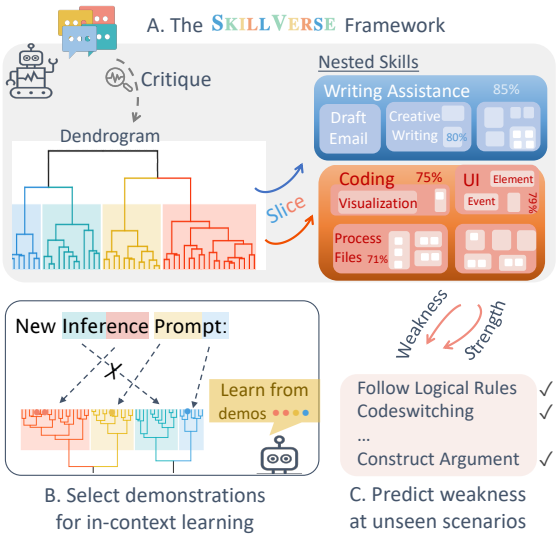


Figure 1: **Up:** The input-output flow of SKILLVERSE. Skill-specific critiques are extracted, structured into a dendrogram, and sliced at varying granularities to reveal nested clusters of skills and model proficiency. **Bottom:** Versatile applications of SKILLVERSE, from selecting informative few-shot demonstrations to uncovering hidden model weaknesses.

1 Introduction

In recent years, leaderboard and benchmark results such as ChatbotArena (Chiang et al., 2024) and MMLU (Hendrycks et al., 2020) have become the dominant practice for evaluating the potency of language models. While these results provide a high-level snapshot of a model's rank, their limited interpretability makes it difficult to identify subtle behavioral traits and derive actionable insights (Murahari et al., 2023; Moayeri et al., 2024).

The limited interpretability of the current evaluation paradigm makes it hard to compare the relative strengths and weaknesses of different models. For instance, does a higher-ranked model consistently outperform lower-ranked counterparts across the entire benchmark? Do comparable scores transfer to equivalent model performance on all subdomains? Addressing such questions typically requires manual inspections, which is both time-consuming and costly. These challenges highlight the need for automatic, granular analyses that provide valuable insights to enrich our understanding of model behavior, which paves the road for targeted model improvement of specific capabilities.

Lately, LLM-based evaluations (Li et al., 2023; Zheng et al., 2023), where LLMs critique and judge model responses, have emerged as a scalable approach to approximate human preferences (Wang et al., 2023; Yuan et al., 2024). These methods enable detailed analysis with rich contextual feedback, forming the foundation of our diagnosis framework, SKILLVERSE. Orthogonal to developing more reliable auto-raters, this paper contributes to **structuring contextual feedback** to generate actionable insights for model model evaluation, comparison, debugging, and improvement.

As shown in Figure 1 and Figure 2, SKILL-

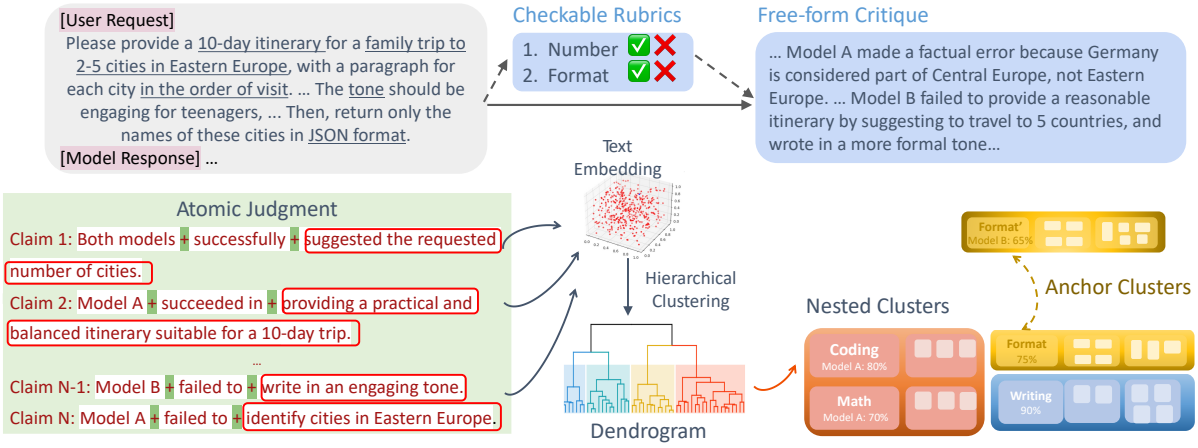


Figure 2: The overall framework of SKILLVERSE. A set of critiques on model responses is parsed into atomic judgments and organized using bottom-up clustering into a dendrogram, which is then unfolded at varying levels of granularity to form nested clusters, allowing for detailed analysis of model proficiency. Thanks to the hierarchical structure, this novel pipeline is highly flexible in interpreting model capabilities.

VERSE generates personalized insights through a tree-structured model assessment, tailored to the level of detail preferred by human scientists. To quantify and extract actionable insights from critiques to diverse real-world data, we introduce *atomic judgment*: an assessment of an indivisible aspect of model capability. Next, we conduct agglomerative clustering on these atomic judgments based on their semantic distance, resulting in a dendrogram. This tree can be chopped at different levels, resulting into clusters of varying sizes or granularities. Each cluster represents a specific skill, for which we calculate success rates to evaluate model performance (§2).

SKILLVERSE produces insights of current model behaviors, such as Gemini, Claude, and GPT-4 (§3). For instance, on Arena-Hard benchmark (Li et al., 2024) where Claude-3.5-Sonnet (Anthropic, 2024) ranks 2nd and Gemini-1.5-Pro (Google, 2024a) ranks 6th,[1] SKILLVERSE finds out Gemini needs improvement in debugging, writing command lines, and business analysis. On the other hand, the higher-ranked Claude, falls short in providing analogical examples, debate, and evaluate arguments. Notably, by comparing different sized models within the same family, we also identify instances of inverse scaling, where larger models underperform smaller ones due to strong parameterized knowledge (McKenzie et al.). Such tasks include handling word inclusion/exclusion, word counts, and adherence to specific formats.

To validate SKILLVERSE's ability to identify true model errors and its potential values for model improvement, we design a few extended improvement explorations that lead to promising gains. In §4, we show that the dendrogram enhances in-context learning by enabling a tree search algorithm that adaptively selects challenging and relevant examples as contrastive few-shot demonstrations. This approach achieves a 25% relative improvement over the standard contrastive in-context learning method (C-ICL, Yan et al. (2021)). In §5, we demonstrate that a strong reasoner such as GPT-4o can digest the model proficiency report generated from SKILLVERSE to predict weaknesses in unseen scenarios. For example, the model proficiency on ten hypothesized tasks is only 55%, 22% lower than uninformed predictions by the same reasoner.

We showcase that SKILLVERSE can serve as a powerful tool for providing fine-grained interpretation of model behaviors and developing targeted improvement of discovered model deficiencies during inference. Future research could also leverage the actionable feedback derived from SKILLVERSE to a wide range tasks: such as model routing, curating targeted training data for specific subdomains where the current model underperforms, and etc.

2 SKILLVERSE : Diagnosis Framework

2.1 Overview

Figure 2 presents the overall framework of SKILLVERSE. Starting with a large set of user prompts and model responses, we collect critiques that evaluate model responses in detail (§2.2). These critiques are then parsed into atomic judgments, enabling efficient organization and large-scale quantification. Using bottom-up clustering algorithms,

[1] Under `Arena-Hard-Auto` on `https://lmarena.ai`, as of Nov 18th, 2024.

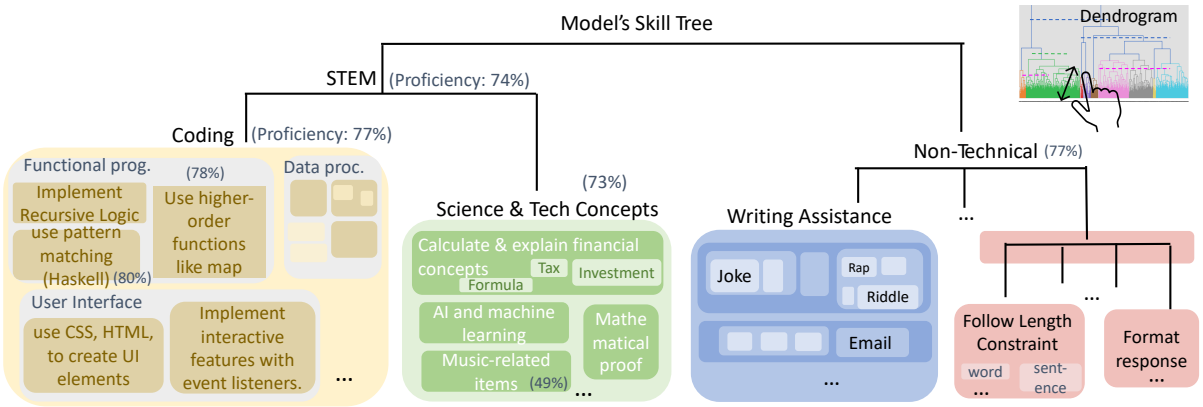


Figure 3: **Upper right**: A dendrogram produced by SKILLVERSE on a combination of two datasets: ChatbotArena (Chiang et al., 2024) and IF Eval (Zhou et al., 2023). **Main figure**: Multiple layers of nested clusters that represent model proficiencies from coarse-grained to fine-grained, by horizontally slicing the dendrogram at different levels. For each group, we can then calculate skill-level model proficiency based on the atomic judgments, as demonstrated in the parenthesis. An LLM summarizes all members in the same cluster and generate the skill-level description.

the atomic judgments are structured into a dendrogram. To interpret the results, the dendrogram is unfolded at varying levels of granularity based on the detail preferred by engineers and researchers (§2.3 and Figure 3). Finally, clusters derived independently are anchored to support multi-party analyses (§2.4).

2.2 Collecting Accurate Critiques

A straightforward way to collect critiques is adopting language models off-the-shelf or finetune them on domain specific data—a reliable approach for evaluating content relevance, style. etc. However, recent studies (Murugadoss et al., 2024; Son et al.; Jing et al., 2024) highlight limitations of LLMs-as-a-Judge in domains like factual verification, format checking, and calculations. Fortunately, aspects like format and calculation are programmatically checkable, eliminating the need to rely solely on language models for feedback.

To enhance accuracy, we first identify the checkable components of a user request and leverage programs to evaluate these metrics. These results are then provided as input to the critique model, alongside the original user prompt and model response, resulting in a more robust evaluation.

2.3 Structuring Diverse Critiques

Converting to Atomic Judgments To efficiently organize thousands of free-form critiques, we introduce the concept of atomic judgments, which serve as act as the building blocks for systematically organizing these critiques. An atomic claim is a statement that addresses a single, non-decomposable aspect of model ability (*e.g.*, "*Model A + failed to + identify cities in Easter Europe.*" as is illustrated in Figure 2). We enforce all atomic judgments to follow a strict syntax with three components: Subject (*i.e.*, the model name) + Verb (*i.e.*, succeed, partially succeed, or fail) + Object (*i.e.*, a specific task), which provides the necessary certainty and precision to quantify large volumes of critiques and calculate model proficiency.

As a preparation step for clustering, we leverage Google's Text Embedding API (Google, 2024b) to vectorize these atomic judgments. Since the both the subject and verb are deterministic, we focus exclusively on embedding and clustering the third component, the specific task.

Hierarchical Clustering We perform agglomerative (*i.e,* bottom-up) clustering on the atomic judgments based on their semantic distance. The algorithm begins by treating each claim as an individual cluster. It then identifies the two closest clusters and merges them. This unsupervised process is repeated until all the clusters are merged into a single one, resulting in a tree of nested clusters, also known as a dendrogram.

Interpreting the Dendrogram To analyze model behavior, we horizontally slice the dendrogram at a preferred level. Next, to obtain a description for each resulting cluster, we prompt an LLM to summarize the group members.

Figure 3 illustrates a dendrogram produced in one of our experiments on the ChatbotArena (Chiang et al., 2024) and the IF Eval dataset (Zhou et al., 2023). This hierarchy captures relations among all data points. A horizontal cut at the highest level yields two primary branches: a left, technical

branch and a right, non-STEM branch. Further slicing these two branches reveals subclusters: 1) *coding*, 2) *calculating formulas and explaining STEM concepts* for the left and 1) *format output*, 2) *providing helpful information*, 3) *writing assistance and other content creation* for the right one. Each cluster is further nested into smaller subclusters (refer to Figure 3 for details). Model proficiency can be computed by calculating the ratio of positive atomic judgments within these clusters.

2.4 Anchoring Clusters

As different models often generate varying responses, the corresponding dendrograms produced by clustering may differ slightly. Additionally, rerunning the clustering process with all existing model responses every time a new model is added is inefficient. To support multi-party analyses, it is essential to merge or anchor clusters derived independently and ensure consistency across models.

Our algorithm merge two clusters only if two key conditions are satisfied. First, the centroids of the clusters must be close to each other in the feature space. Let μ_i and μ_j denote the centroids of clusters C_i and C_j. The clusters can be merged if their similarity exceeds a threshold τ:

$$\text{sim}(\mu_i, \mu_j) = \frac{\mu_i \cdot \mu_j}{\|\mu_i\|\|\mu_j\|} \geq \tau \quad (1)$$

Second, there must be significant overlap between the clusters. For instance, consider a counterexample where cluster C_i is a large circle with 1,000 members, and C_j is a single point at the center, merging would be inappropriate despite centroid proximity due to minimal overlap. Overlap is quantified as the intersection of cluster regions $\text{Area}(C_i \cap C_j)$ relative to their union $\text{Area}(C_i \cup C_j)$, satisfying:

$$\frac{\text{Area}(C_i \cap C_j)}{\text{Area}(C_i \cup C_j)} \geq \epsilon. \quad (2)$$

Both conditions ensure the merged cluster represents the data accurately without adding ambiguity or excessive variance.

3 Results

3.1 Verifying SKILLVERSE Reliability

Assuming the critiques are reliable and given that the success rates are algorithmically calculated, the only potential source of error in our framework arises from the unsupervised clustering process.[2]

	Predicted Neg.	Predicted Pos.
Actual Neg.	TN = 0.883	FP = 0.084
Actual Pos.	FN = 0.117	TP = 0.916

Table 1: The performance of our clustering algorithm.

We judged the accuracy of our unsupervised cluster with human evaluation, as reported below:

Accuracy of Clustering We recruit human annotators to evaluate the similarity between pairs of user requests sampled from atomic claims (*e.g.,* <writing lyrics that are less cliché, calculating RAM occupation>), rating similarity on a scale from 1 (completely different) to 5 (highly similar). Each input is rated by 3 annotators, resulting in 1,590 annotations. Detailed guidelines are shown in Figures 13 and 14 in Appendix B.3. Comparing these human annotations with our embedding similarity produces a Pearson correlation of 0.643 (p<0.0001), indicating substantial agreement.

Next, we converted the human-provided 5-point scale scores into binary categories, where scores of 4 and 5 indicate the same cluster and 1 and 2 indicate the opposite. Ambiguous pairs with a score of 3 were excluded, leaving 993 cases. These remaining cases were then divided into validation (for optimal slicing threshold) and test sets (for evaluation). Table 1 shows that hierarchical clustering achieved a true positive rate of 0.916 and a true negative rate of 0.88.

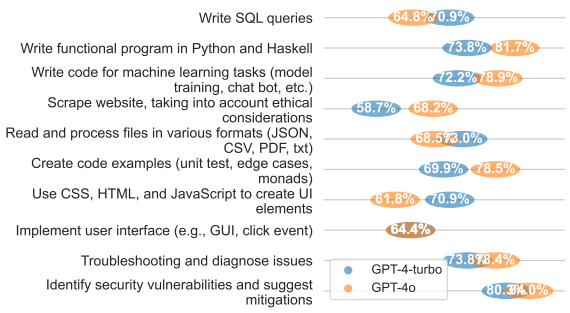
Accuracy of Anchoring We evaluated our anchoring procedure using dendrograms from three independently derived model sets (Llama3, Gemini1.5, Claude3). Slicing these dendrograms at the same threshold yielded 54, 58, and 55 clusters, respectively. Human annotators reviewed 30 random members from each cluster to decide on merging, establishing a gold standard. These clusters were split into validation and test sets. We apply grid search to optimize thresholds τ and ϵ on the validation set. Test results show our merging algorithm achieved a precision of 0.926 and a recall of 0.980.

In summary, both confirm the effectiveness of our approach in constructing and merging clusters.

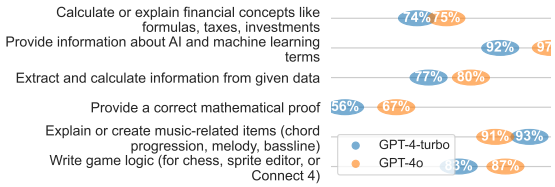
3.2 Insights of Fine-Grained Model Behavior

While our framework is generic and can be applied to any {prompt, response} pairs, we showcase a

[2] SKILLVERSE works with any critiques or rationales, whether human-provided or automatically generated. While improving LLM evaluators is beyond the scope of this paper, advancements in neural evaluators will naturally enhance the framework's reliability.



(a) Performance of GPT-4-turbo and GPT-4o on coding.



(b) Results on other STEM areas.

Figure 4: Comparing the strengths and weaknesses of two proprietary models that were release consecutively.

few exemplars: 1) comparison of two proprietary models that were release consecutively by the same company(*i.e.,* GPT-4o vs GPT-4-turbo), 2) comparisons across different model families, and 3) comparison between large and small models within the same family.

Comparison within the same model family Figure 4 illustrates the performance comparison between GPT-4o and GPT-4-turbo on STEM tasks. Despite GPT-4o being a more recent and ostensibly stronger release, SKILLVERSE reveals that GPT-4-turbo outperforms GPT-4o in specific areas, including writing SQL queries (6.1% improvement), reading and processing files (9.1% improvement), and handling music-related tasks (2% improvement).

Comparison across different families Similarly, we compare the best-performing models (as of November 1, 2024) across three families: Claude 3.5-Sonnet, Gemini-1.5-pro, and GPT-4o. SKILLVERSE reveals that Claude excels in coding and analytical tasks such as visualization (*e.g.,* 85.5% vs 76.8%-79.5%), creating or using AI models, handling edge cases, and writing shell commands; Gemini performs best in developing contents for educational purposes, game creation, and text formatting; while GPT-4o is superior at producing mathematical proofs, and it is exceptional at inferring the user's precise intent from vague instructions (83.7% vs 63.2%). We provide a comprehensive view of the wins and losses of these models in Figures 11 and 12 in the Appendix.

Identified Capabilities that Follow Inverse Scaling
Wrap the entire response in double quotes
Format text using markdown
Output in JSON format
End the response with a specific phrase
Following format of limericks and rhyme scheme
Include/exclude specific phrases
Comply with the word count requirement

Table 2: SKILLVERSE identified capabilities that follow inverse scaling, where increasing model size deteriorates performance.

Are larger models always better than its smaller counterparts? Another interesting finding emerges when comparing large and small models within the same family: Gemini-1.5 (Pro vs. Flash), Llama3.1 (405B, 70B, and 8B), and Claude3 (Opus, Sonnet, Haiku). On average, larger models outperform smaller ones across over 95% of identified capabilities, including STEM, problem-solving, and writing tasks. However, there are a few exceptions that demonstrate inverse scaling, where increasing model size deteriorates performance (McKenzie et al.). SKILLVERSE discovers inverse scaling on tasks with fine-grained constraints, such as keyword inclusion/exclusion and strict formatting, as detailed in Table 2.

4 SKILLVERSE Enhances Model Performance at Inference Time

The remainder of this paper explores two extended tasks. In this section, we illustrate how SKILLVERSE serves as a knowledge base of model proficiency and helps improve inference-time performance by providing better few-shot demonstrations that considers both relevance and challenges posed to the target language model. In Section 5, we demonstrate how the the uncovered model proficiency can serve as a foundation to predict model failures in previously unseen scenarios.

4.1 Approach

Motivation Contrastive in-context learning (C-ICL), which presents an LLM with both correct and incorrect examples as demonstration, have been shown to effectively guide the models in distinguishing between desired and undesired outputs across various tasks such as information extraction (Chao et al., 2024) and reasoning (Chia et al., 2023a; Zhang et al., 2024).

However, a typical method to construct contrastive examples involves synthetically generating

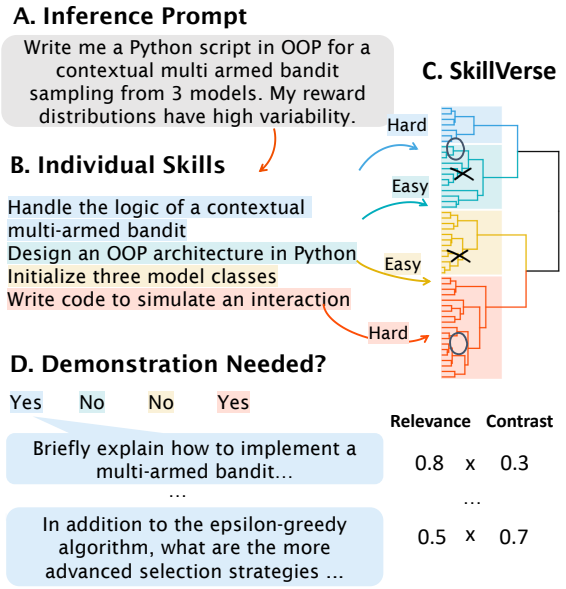


Figure 5: The dendrogram produced by SKILLVERSE helps to selects more informative few-shot demonstrations by considering both relevance and challenges posed to the target model. In contrast, previous methods selected *'...output a Python script in OOP for a bandit-inspired approach to optimize hyper-parameters across 3 models.'* which is semantically most similar but helped less as an in-context example.

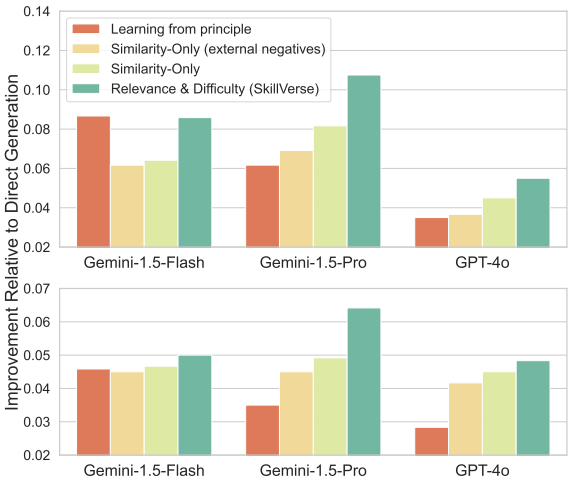


Figure 6: Improvement of different in-context learning approaches, compared to direct generation on IF Eval (**top**) and Chatbot Arena (**bottom**). We posit that the performance gain is smaller for GPT-4o because of the strong performance of direct generation.

negative responses by introducing hand-crafted error types, which may not best reflect a model's own distribution. Moreover, errors in C-ICL may arise from models "over-reflecting" on simple prompts that LLMs already know how to answer. We hypothesize that SKILLVERSE mitigates the first issue by naturally storing pairs of good and bad responses, thereby facilitating an LLM's ability to learn from its own mistakes. Additionally, access to detailed model proficiency helps resolve the second issue, as it allows us to dynamically determine whether—and which part of—an inference prompt poses more challenge to the target model.

Method Figure 5 illustrates the three steps to select few-shot examples with SKILLVERSE:

Step 1: Skill Identification. Given an inference prompt, an LLM analyzes and predicts the individual skills required to solve the task

Step 2: Mapping and Pruning. The identified skills are located within an existing dendrogram, where simpler branches (*e.g.*, those with a success rate $\geq T$) are pruned.

Step 3: Selecting Few-Shot Demonstrations. The remaining candidate pairs are re-ranked based on two factors: (1) content relevance and (2) the benefit provided by the current contrastive pair. Here, the benefit is defined as $C(r_1) - C(r_2)$, where $C(\cdot)$ denotes the scaler score labeled by the critique model, r_1 refers to the correct response generated by another model and r_2 refers to incorrect response generated by the target model.

4.2 Experiments

Compared models We compare with principle learning from mistakes (Zhang et al., 2024), which prompts the model to learn from the distilled principles derived from self-made mistakes. We also ran two ablations for selecting few-shot examples: similarity-only that selects semantically similar instances (Mo et al., 2024), with or without incorporating self-generated errors as negative responses. Additional details about the experimental setup can be found in § B.

Results We present the performance under different ICL settings in Figure 5. Interestingly, we find that 'learning from principles' works well with smaller models such as Gemini-1.5-flash. One possible explanation is that smaller models have limited capacity for reasoning about correct and incorrect answers in long contexts. Therefore, directly providing high-level principles might be a more effective strategy. Overall, SKILLVERSE consistently outperforms or performs on par with all baseline models. This indicates that it successfully serves as a knowledge base of granular model proficiencies, enabling the selection of more informative in-context examples to guide the target model.

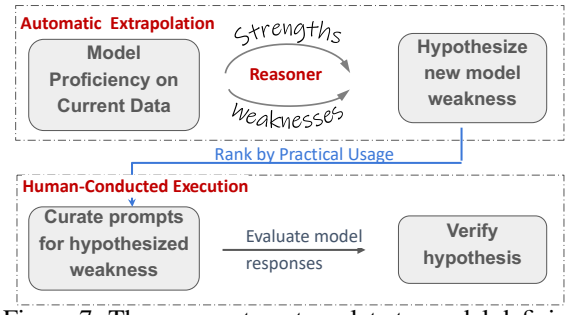


Figure 7: The process to extrapolate to model deficiencies in unseen scenarios. An LLM acts as a reasoning model to generate the hypothesis, and humans execute experiments to verify these AI-generated hypotheses.

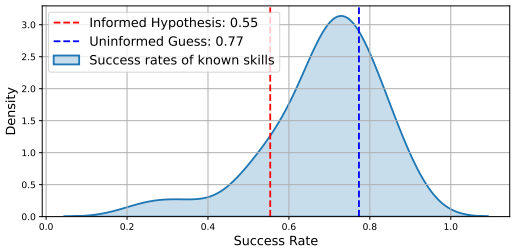


Figure 8: The performance of Gemini-1.5-flash under different settings. The success rate of SKILLVERSE-informed hypothesized weakness is only 55%, 22% lower than the uninformed hypothesis. We also list exemplar hypothesized weakness and the corresponding model performance in Table 3.

5 Auto-Discovery: Extrapolating Model Weakness to Unseen Scenarios

5.1 Approach

We explore the feasibility of automatically extrapolating to unseen error types where models may underperform. As is shown in Figure 7, we first provide the target model's capabilities on existing data and to a reasoning LLM[3] to uncover the underlying connections between areas where models perform well and poorly. Based on this analysis, we ask the reasoning model to hypothesize potential deficiencies of the target model, based on which humans curate prompts to test these hypothesized weaknesses individually.

5.2 Experimental Setup

We conduct experiments under **two settings**: 1) identify new weaknesses in a single model, *e.g.,* Gemini-1.5-flash, and 2) predict inverse scaling,, where a larger model underperforms its smaller counterpart. *e.g.,* Claude-3-Opus underperforming Claude-3-Sonnet on certain skills. We ask the reasoning model to hypothesize 50 tasks for the first setting and 20 for the second. To filter out less significant predicted tasks, such as "writing a paragraph in alternating capital and small letters", we re-ranked them by practical relevance and selected the top half. For each selected task, we then collected 150 user prompts to test model capability solely on this task, gathered model responses, and evaluated model success rates.

As **an uninformed baseline**, we test the reasoning model's ability to predict weaknesses without performance data. Specifically, we give it a random subset of skills and prompt it to propose new tasks where the model may fail. Comparing these uninformed predictions to informed ones reveals whether meaningful extrapolations arise from the reasoning model inherently or from SKILLVERSE.

[3]A different reasoning LLM is deliberately chosen to minimize inherent biases in the target model. Specifically, we use GPT-4o as it had the strongest reasoning capabilities when our experiments were conducted.

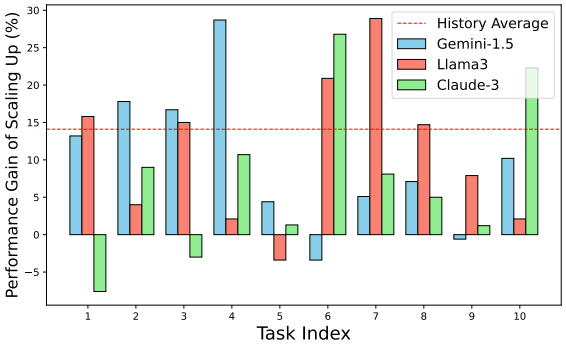


Figure 9: Ten hypothesized tasks that are likely to follow inverse scaling, and the actual performance gain of scaling up. These hypotheses include *formatting a bibliography in APA style*, and *inserting hyperlinks into a document*, as listed in Table 4. Positive values show larger models outperforming smaller ones; negative values show underperformance. We show the average gain from existing data as a horizontal reference line.

5.3 Result

We visualize Gemini-1.5-flash's success rate on SKILLVERSE-informed predicted weaknesses in Figure 8. These predicted tasks are, on average, 14.2% more challenging than existing tasks and 22% more challenging than uninformed predictions. As shown in Table 3, the reasoning model successfully predicted weaknesses in following logical relations (14.8%) and avoiding specific phonemes (27.1%). In contrast, without the contrastive insights provided by SKILLVERSE, the same reasoner wrongly predicted weaknesses in presenting opposing opinions (98.2%) and contradicting physical realities (100%)——tasks where the model actually excelled.

Moreover, as is shown in Figure 9, the reasoner

ID	SKILLVERSE-Informed Hypothesis	Succ.	Uninformed Hypothesis	Succ.
1	Multilingual Code Switching: Seamlessly alternating between two or more languages.	0.607	Opposing Opinions: Present two opposing opinions with equal depth and justification.	0.982
2	Logical Relations: Avoid or include words based on logical rules (e.g., AND, XOR, conditional).	0.148	Encode Hidden Information: Insert a hidden message using techniques like acrostics or word placements.	0.622
3	Avoid Specific Phonemes: Write text excluding words with selected phonemes (e.g., "th").	0.271	Physical Uncommonsense: Write a story where physical laws are broken (e.g., objects floating, time moving backward).	1.00
4	Argument Construction: Develop a three-part argument (premise, reasoning, conclusion).	0.506	Speech Impediments: Create dialogue with a specific impediment or linguistic quirk.	0.704
5	Dynamic Math Puzzles: Create riddles where each solution depends on the previous one.	0.460	Rhyming with Meaning: Write a poem where rhyming words form a meaningful phrase.	0.966

Table 3: Comparison of SKILLVERSE-informed and uninformed predicted model weaknesses, with Gemini-1.5-flash success rates. The Kolmogorov-Smirnov test confirms statistically different distributions (p-value = 0.02).

also succeeds in identifying capabilities where stronger models may underperform their weaker counterparts. Appendix B lists the predicted tasks for inverse scaling, both with (Table 4) and without (Table 5) the findings by SKILLVERSE. Under the informed setting, the average performance gain of scaling up is merely 0.5%, which is statistically different from the 10.6% gain observed in uninformed predictions. Both results highlight the value of SKILLVERSE in predicting unseen model weaknesses, enabling proactive identification of potential limitations before deployment, rather than merely fixing issues after they arise.

6 Related works

Interpreting Model Behaviors. Shifting focus from aggregated leaderboard metrics, researchers have been striving to interpret model losses more effectively. For instance, LLMSys (Chiang et al., 2024) uses BERTopic to embed prompts, reduce dimensionality, and cluster them into a predefined number of groups. QualEval (Murahari et al., 2023) and a concurrent work, SkillIndex (Moayeri et al., 2024), identify attributes like subtasks and domains from evaluation data and then assign them to individual data points. In contrast, SKILLVERSE derives hierarchical clusters entirely unsupervised. Its tree structure enables efficient tracing of semantically similar prompts for downstream tasks while giving users control over granularity—chopping the tree at lower levels provides finer-grained loss categories for model capabilities.

LLM as Evaluator. Recently, LLM-based evaluation (Li et al., 2023) that requires the LLM to provide critiques to responses across a wide range of domains have emerged a scalable method for approximating human preferences (Zheng et al., 2023; Chang et al., 2024). Vu et al. (2024) and Wang et al. (2023, 2024) have demonstrated that critique models improve agreement with human judgment and reduce bias of the assessments when supervised multi-task fine-tuning is used. As a result, LLM-as-a-judge offers a practical alternative to traditional, labor-intensive methods of human preference collection and reward modeling (Wang et al., 2023; Yuan et al., 2024).

Learning from Mistakes. With discovered losses, how can we further improve the model? Extensive research has explored both training-time correction and inference-time improvement from mistakes or feedback (Pan et al., 2024) (LLM-Refine (Xu et al., 2024), SelfRefine (Madaan et al., 2023), etc.). SKILLVERSE enhances model performance during inference. Among prior works, contrastive chain-of-thought prompting (Chia et al., 2023b) and principle learning from mistakes (Zhang et al., 2024) are most relevant as both leverage model mistakes via in-context learning. However, the few-shot demonstration examples are fixed and predefined in these works, whereas SKILLVERSE adaptively selects examples for in-context learning through its dendrogram, balancing semantic relevance and potential benefit.

7 Conclusion

We developed a hierarchical diagnosis framework that distills a tree of fine-grained model capabilities from unstructured traffic data. Our framework offers the following key benefits: 1) it provides flexible insights into nuanced model abilities that are not captured by existing leaderboards or benchmarks, 2) SKILLVERSE serves as a knowledge base of model proficiency and helps enhance the model at inference-time by providing better few-shot demonstrations, and 3) it can be used to predict unseen error types before deployment.

Limitation

As pointed out by Murahari et al. (2023), fine-grained model analysis does not reject the use of benchmark metrics but uses them as one of the parts of a more actionable evaluation.

One limitation of SKILLVERSE is that we use LLMs as judges to generate critiques of model responses to user prompts, which might introduce errors. Although out-of-scope of this work, developing more robust and accurate automatic critique models can definitely improve the utility of SKILLVERSE. Inspired by the conclusion from prior works that large language models are better at evaluating model capabilities from comparison than evaluating the single model's response in isolation (Liusie et al., 2024a,b), we always compare responses from a pair of models and generate critiques. However, as recent work suggested, pairwise comparisons can sometimes amplify biases present in LLM evaluators (Kawabata and Sugawara, 2024). Therefore, it is crucial to be aware of potential biases that may arise during the evaluation process. In addition, as a framework designed to systematically assess model capabilities and enhance performance, we emphasize the importance of preventing the misuse of SKILLVERSE.

References

Anthropic. 2024. Introducing claude 3.5 sonnet. Accessed: 2024-12-14.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology, 15(3):1–45.

Liang Chao, Wei Xiang, and Bang Wang. 2024. In-context contrastive learning for event causality identification. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 868–881.

Yew Ken Chia, Guizhen Chen, Luu Anh Tuan, Soujanya Poria, and Lidong Bing. 2023a. Contrastive chain-of-thought prompting. arXiv preprint arXiv:2311.09277.

Yew Ken Chia, Guizhen Chen, Luu Anh Tuan, Soujanya Poria, and Lidong Bing. 2023b. Contrastive chain-of-thought prompting. Preprint, arXiv:2311.09277.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot arena: An open platform for evaluating llms by human preference. arXiv preprint arXiv:2403.04132.

Google. 2024a. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. arXiv preprint arXiv:2403.05530.

Google. 2024b. Text embeddings api. https://cloud.google.com/vertex-ai/generative-ai/docs/embeddings/get-text-embeddings. Accessed: 2024-12-14.

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

Xiaonan Jing, Srinivas Billa, and Danny Godbout. 2024. On a scale from 1 to 5: Quantifying hallucination in faithfulness evaluation. arXiv preprint arXiv:2410.12222.

Akira Kawabata and Saku Sugawara. 2024. Rationale-aware answer verification by pairwise self-evaluation. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 16178–16196, Miami, Florida, USA. Association for Computational Linguistics.

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

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Adian Liusie, Potsawee Manakul, and Mark Gales. 2024a. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 139–151, St. Julian's, Malta. Association for Computational Linguistics.

Adian Liusie, Vatsal Raina, Yassir Fathullah, and Mark Gales. 2024b. Efficient LLM comparative assessment: A product of experts framework for pairwise comparisons. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 6835–6855, Miami, Florida, USA. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder,

Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In Advances in Neural Information Processing Systems, volume 36, pages 46534–46594. Curran Associates, Inc.

Ian R McKenzie, Alexander Lyzhov, Michael Martin Pieler, Alicia Parrish, Aaron Mueller, Ameya Prabhu, Euan McLean, Xudong Shen, Joe Cavanagh, Andrew George Gritsevskiy, et al. Inverse scaling: When bigger isn't better. Transactions on Machine Learning Research.

Ying Mo, Jiahao Liu, Jian Yang, Qifan Wang, Shun Zhang, Jingang Wang, and Zhoujun Li. 2024. C-icl: contrastive in-context learning for information extraction. arXiv preprint arXiv:2402.11254.

Mazda Moayeri, Vidhisha Balachandran, Varun Chandrasekaran, Safoora Yousefi, Thomas Fel, Soheil Feizi, Besmira Nushi, Neel Joshi, and Vibhav Vineet. 2024. Unearthing skill-level insights for understanding trade-offs of foundation models. arXiv preprint arXiv:2410.13826.

Vishvak Murahari, Ameet Deshpande, Peter Clark, Tanmay Rajpurohit, Ashish Sabharwal, Karthik Narasimhan, and Ashwin Kalyan. 2023. Qualeval: Qualitative evaluation for model improvement. arXiv preprint arXiv:2311.02807.

Bhuvanashree Murugadoss, Christian Poelitz, Ian Drosos, Vu Le, Nick McKenna, Carina Suzana Negreanu, Chris Parnin, and Advait Sarkar. 2024. Evaluating the evaluator: Measuring llms' adherence to task evaluation instructions. arXiv preprint arXiv:2408.08781.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. Transactions of the Association for Computational Linguistics, 12:484–506.

Guijin Son, Hyunwoo Ko, Hoyoung Lee, Yewon Kim, and Seunghyeok Hong. Llm-as-a-judge & reward model: What they can and cannot do.

Tu Vu, Kalpesh Krishna, Salaheddin Alzubi, Chris Tar, Manaal Faruqui, and Yun-Hsuan Sung. 2024. Foundational autoraters: Taming large language models for better automatic evaluation. arXiv preprint arXiv:2407.10817.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. 2024. Self-taught evaluators. arXiv preprint arXiv:2408.02666.

Tianlu Wang, Ping Yu, Xiaoqing Ellen Tan, Sean O'Brien, Ramakanth Pasunuru, Jane Dwivedi-Yu, Olga Golovneva, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. Shepherd: A critic for language model generation. arXiv preprint arXiv:2308.04592.

Wenda Xu, Daniel Deutsch, Mara Finkelstein, Juraj Juraska, Biao Zhang, Zhongtao Liu, William Yang Wang, Lei Li, and Markus Freitag. 2024. LLMRefine: Pinpointing and refining large language models via fine-grained actionable feedback. In Findings of the Association for Computational Linguistics: NAACL 2024, pages 1429–1445, Mexico City, Mexico. Association for Computational Linguistics.

Caixia Yan, Xiao Chang, Minnan Luo, Huan Liu, Xiaoqin Zhang, and Qinghua Zheng. 2021. Semantics-guided contrastive network for zero-shot object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 46:1530–1544.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. arXiv preprint arXiv:2401.10020.

Tianjun Zhang, Aman Madaan, Luyu Gao, Steven Zheng, Swaroop Mishra, Yiming Yang, Niket Tandon, and Uri Alon. 2024. In-context principle learning from mistakes. arXiv preprint arXiv:2402.05403.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. Advances in Neural Information Processing Systems, 36:46595–46623.

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

Appendix

A Full Analysis Result

Figure 10 is an example of the Gemini-1.5-flash's capability report generated by our framework. It consists of a high-level summary, descriptions of fine-grained capabilities along with the model success rates. Overall, it is good at text formatting, and needs improvement in subdomains such as following length constraints, writing riddles, and assisting STEM tasks.

Figures 11 and 12 present a comprehensive capability report comparing Gemini-1.5-pro, Claude3.5-Sonnet, and GPT-4o on the ChatbotArena benchmark (Chiang et al., 2024). The report includes a high-level summary, detailed descriptions of fine-grained capabilities, and the models' success rates. SKILLVERSE reveals that Claude excels in coding and analytical tasks, such as visualization (*e.g.*, 85.5% vs. 76.8%-79.5%), creating or using AI models, handling edge cases, and writing shell commands. Gemini performs best in creating content for educational purposes, game development, and text formatting. Meanwhile, GPT-4o stands out in producing mathematical proofs and is exceptional at inferring user intent from vague instructions (83.7% vs. 63.2%).

B Experimental Details

B.1 Data Used in In-Context Learning

We evaluated our approach to select few-shot demonstrations on GPT-4o, Gemini-1.5-pro, and Gemini-1.5-flash using two datasets: a well-structured IF-Eval dataset that involves instruction following such as format (Zhou et al., 2023), and the less structured ChatbotArena that involves reasoning tasks (Chiang et al., 2024). We conduct the diagnosis process of SKILLVERSE using 450 prompts from the first dataset and 2,500 prompts from the second, with inference performed on 150 held-out prompts for each dataset.

B.2 Detailed Results of Auto-Discovery

Table 4 and Table 5 lists the predicted inverse scaling tasks with or without the insights produced by SKILLVERSE. A positive value indicates that the larger model outperforms its smaller counterpart, while a negative value indicates underperformance. On average, the larger models outperform their smaller siblings by only 0.5%, compared to 10.6% in the uninformed predictions.

High-Level Summary	Fine-Grained Description	Success Rate
Creative writing in multiple forms.	write jokes	87.8%
	write a riddle	**70.0%**
	write a song (rap or jazz)	82.6%
	write a poem (limerick, haiku)	72.9%
follow length constraint	write within word limit	**64.8%**
	follow request on sentence count	**69.1%**
Text formatting	format and use bullet points	73.7%
	format the title (double angular brackets)	90.7%
	separate items with six asterisks	79.4%
	separate paragraphs with special characters	92.9%
	add a postscript starting with e.g., "P.S."	78.6%
	format and highlight sections in the text	85.5%
Code-related tasks, including providing visualization, comments, documentation.	generate visualization	78.3%
	write command-line script (for variable replacement, synchronize folders, etc.)	**71.3%**
	add correct comments for functions	76.7%
	identify and handle errors, troubleshooting	75.0%
	format output in JSON format	**64.2%**
Providing technical assistance	provide technical instructions on a wide range of technologies	76.3%
	calculate or estimate (range, confidence interval, ambiguity, CPU usage)	87.0%
	work with time and date (calculate, provide, explain)	**59.3%**

Figure 10: An example of the Gemini-1.5-flash's capability report generated by our framework. It consists of a high-level summary, descriptions of fine-grained capabilities along with the model success rates. We highlight model weaknesses in red.

B.3 Human Annotation

We present the complete annotation guideline used in Section 3 to verify the accuracy of our clustering algorithm in Figure 13 and 14 The inter-annotator agreement is 0.88 as measured by Pearson correlation.

High-Level Summary	Detailed description	Gemini-1.5-pro	Claude3.5-Sonnet	GPT-4o
Write functional and specific-purpose code, focusing on implementation and problem-solving.	write code for visualization purpose (e.g., scattered plot, bar chart)	79.49%	85.53%	76.83%
	implement a feature or write code for game development (e.g., terrain generation, sprite editor)	71.25%	69.03%	74.04%
	create or use AI/ML model/chatbot	87.61%	94.56%	90.70%
	write SQL queries,containing specific techniques like CTEs, joins, window functions	84.92%	89.92%	85.42%
	improve code quality (organization, efficiency, conciseness, functionality)	62.51%	58.81%	70.44%
	comment on code quality (efficiency, readability)	90.93%	84.87%	92.44%
	translate into first-order logic (FOL) and conjunctive normal form (CNF),	83.85%	78.09%	80.44%
	write test cases, handling edge cases and generating the expected output	88.99%	97.26%	93.86%
	provide detailed instructions for package installation, configuration, and setup	96.31%	89.28%	91.83%
	write code for games/chess/elo, create card decks, solve rubiks cube	76.79%	73.57%	80.78%
	write code or script for shell commands (file operations, automation, configuration)	80.40%	90.11%	96.43%
	write code or function about network, stack, TCP, IP address	74.99%	65.23%	70.44%
	Fixing bugs or troubleshooting	86.14%	77.92%	94.48%
	write code for data processing (e.g., scrape data, convert/generate files, save files, extract content)	79.78%	85.14%	86.43%
Compose clear and comprehensive explanations for algorithms	provide explanations of algorithms including aspects like steps, examples, details, and target audiences	96.97%	90.63%	89.54%
	use examples (analogy, specific, real-world, illustrative)	89.18%	62.06%	81.98%
	compare (pros/cons of) methods/strategies/options	93.90%	82.99%	89.55%
	provide explanation (sentiment analysis, chatbot, LLM, Langchain, AutoGen)	88.05%	88.55%	91.07%
	complexity analysis for algorithm	86.93%	87.89%	82.83%
	provide helpful, correct and relevant suggestions and advice	100.00%	97.96%	98.15%
	write documentation including comments and docstrings	94.87%	90.70%	83.86%
Provide mathematical analysis, calculations, or proofs.	provide or write mathematical proofs	62.00%	75.73%	83.77%
	calculate financial terms (interest rate, future value, present value, profit/loss, etc) or analyze (options, hedging, market behavior, etc)	79.56%	75.63%	82.63%
	solve a (math, physics) problem	82.35%	88.25%	78.46%
	perform calculation or derive formula	77.42%	70.25%	75.46%
Create and analyze technical content with practical details.	create game related content like character sheet or game session outline	89.20%	70.74%	86.51%
	analyze and advise on security vulnerabilities and best practices	89.20%	79.41%	85.68%
	write music-related items (song, melody, chord progression) using formats like ABC notation and MIDI	85.73%	81.59%	71.26%
	provide detailed business analysis like strategy, model, SWOT, market, and finance	92.20%	96.60%	91.86%
	design on a topic like software, schema, or engineering process	87.29%	85.19%	80.65%
	debate or evaluate arguments	90.90%	69.11%	74.95%

Figure 11: The capability report comparing Gemini-1.5-pro, Claude3.5-Sonnet, and GPT-4o on ChatbotArena (Chiang et al., 2024). It consists of a high-level summary, descriptions of fine-grained capabilities along with the model success rates.

High-Level Summary	Detailed description	Gemini-1.5-pro	Claude3.5-Sonnet	GPT-4o
Develop, evaluate, and refine diverse content (creative, educational)	provide comprehensive recipe (ingredient, flavor, nutrition, instruction)	93.29%	92.97%	85.72%
	create a structured and well-organized curriculum or outline for a course (covering topics, levels, titles)	95.22%	97.31%	88.28%
	write creative content like story, script, lyrics, article	79.55%	81.85%	70.26%
	translate or analyze (grammar, meaning, usage) of language and write in a specific style (tone, format)	84.50%	86.47%	82.01%
	create a detailed training plan with time estimates and specific workouts	80.48%	76.45%	66.69%
	categorize, extract, and identify entities, relationships, from long context	78.81%	82.65%	75.01%
Focus on clarity, conciseness, and formatting in writing.	structure text using headings, (numbered) bullet points, and bolding	91.65%	80.00%	88.72%
	write in a clear, well-organized, easy to understand, and readable format (e.g., formatting, structure, clarity)	92.37%	89.20%	88.72%
	provide relevant explanation	96.64%	90.01%	95.81%
	be concise and to the point (instead of overly verbose, repetitive)	95.21%	97.00%	91.32%
	write concisely, balancing the level of detail (using examples, visual aids)	84.85%	72.74%	91.50%
Understand and address user intent with clear, structured responses.	address vague requests and identify the user's actual ntent	63.18%	63.18%	83.68%
	understand and fulfill the prompt, addressing requirements, instructions, and questions	91.53%	89.50%	89.99%
	provide help on ethical related issues	97.81%	92.74%	97.32%
	not refuse too many requests due to ethical concerns	63.34%	83.77%	87.26%
	provide resources (links, references)	72.02%	69.39%	78.28%
	provide truthful, relevant, accurate, and factual information	77.38%	80.97%	80.52%
	write about comprehensive details (scope, limitations, potential issues)	89.29%	83.78%	87.42%

Figure 12: Figure 11 continued.

ID	Description	Gemini-1.5	Llama-3	Claude-3	Avg.
1	Format a bibliography in APA style based on a list of references	-13.2%	7.7%	-13.6%	-6.4%
2	Insert a hyperlink for every occurrence of a specific word	4.7%	0.4%	2.3%	2.5%
3	Write all numbers in words instead of numerals	-3.4%	-3.5%	29.0%	7.4%
4	Write a response where every second sentence starts with the same word	-8.7%	6.5%	7.7%	1.8%
5	Use exactly one comma per sentence, placed in a specified position	2.9%	-1.7%	0.1%	0.4%
6	End each sentence with a specific punctuation mark (e.g., every sentence must end with an exclamation point)	-20.0%	14.6%	-6.3%	-3.9%
7	Format a table using LaTeX syntax	-9.0%	5.2%	5.2%	0.5%
8	Replace every occurrence of the word 'the' with 'a'	13.7%	12.6%	-6.5%	6.6%
9	Enclose the entire response in an HTML and tag	1.8%	1.8%	-8.8%	-1.8%
10	Write a strict-structure sonnet while maintaining a particular rhyme scheme (e.g., iambic pentameter)	-0.2%	5.3%	1.6%	2.2%
11	Wrap conversations and individual sentences in parentheses	-10.9%	5.9%	-7.3%	-4.1%

Table 4: Predicted inverse scaling tasks (SKILLVERSE-informed) and the performance gap between larger and smaller models. We evaluated Gemini-1.5 (pro vs. flash), Llama3 (405b vs. 70b), and Claude3 (Opus vs. Sonnet). A positive value indicates that the larger model outperforms its smaller counterpart, while a negative value indicates underperformance. On average, the larger models outperform their smaller siblings by only 0.5%, compared to 10.6% in the uninformed predictions (Table 5 in the Appendix).

Index	Description	Gemini-1.5	Llama3	Claude-3	Average
1	Respond to a question using only questions	13.2%	15.8%	-7.6%	7.1%
2	Recommend a Playlist Based on Mood	17.8%	4.0%	9.0%	10.3%
3	Describe something as if it were a recipe	16.7%	15.0%	-3.0%	9.6%
4	Suggest a Weekly Meal Plan	28.7%	2.1%	10.7%	13.8%
5	Create a riddle for technical concepts	29.4%	23.8%	16.4%	23.2%
6	Describe a process, but only using the future tense	4.4%	-3.4%	1.3%	0.8%
7	Reverse Word Order	-3.4%	20.9%	26.8%	14.8%
8	Generate Advice for Improving Public Speaking Skills	5.1%	28.9%	8.1%	14.0%
9	Write a song chorus and bridge	7.1%	14.7%	5.0%	8.9%
10	Generate Rhyming Words	-0.6%	7.9%	1.2%	2.8%
11	Alliteration Generation	10.2%	2.1%	22.3%	11.5%

Table 5: Predicted inverse scaling tasks (uninformed predictions) and the performance gap between larger and smaller models. We evaluated Gemini-1.5 (pro vs. flash), Llama3 (405b vs. 70b), and Claude3 (Opus vs. Sonnet). A positive value indicates that the larger model outperforms its smaller counterpart, while a negative value indicates underperformance. On average, the larger models outperform their smaller siblings by 10.6%, with a distribution that is statistically different from those predicted with the performance data as input.

Introduction of Tasks

Welcome to the phrase similarity annotation task! The objective of this task is to **annotate the similarity between two phrases that describe user requests**. As a rater, you will be provided with pairs of short phrases sourced from various user requests or feedback. Some requests are extremely similar, while others are not. Your task is to label their similarity on a scale from 1 to 5.

Rater Instructions

To ensure the quality of the annotation, please follow these detailed instructions:

1. **Review the given phrases carefully**
   a. The phrases you will annotate come from various ***user requests*** or ***feedback to these requests***
   b. **Ignore** the sentiment of the feedback (e.g., positive or negative), and focus solely on the similarity of the user request parts.
   c. For more details, please refer to examples 01 and 03
2. **Choose a similarity rating from a 1-5 scale**
   - 5-Highly similar (same group): The phrases are highly similar and should belong to the same group.
   - 4- Relevant (sister groups): The phrases are relevant and should belong to related groups (sister groups).
   - 3-Neutral: Choose this when the similarity is ambiguous or you are not sure.
   - 2-Weakly related (leaning towards unrelated): The phrases are leaning towards unrelated, but you can see how they might be distantly related.
   - 1-Totally unrelated (never be sister groups): The phrases have no similarity and should never be in the same group

Examples:

Please **ignore the sentiment of the feedback (e.g., succeed or fail)**, and only focus on the **similarity between the user request parts**.

ID	Input Pairs	Expected Annotation
01	A-write a poem about playing video games, B-wrote a haiku about missing class, failing to follow the poem structure	5-Highly similar (same gro…
02	A-used the word "peace" at least 10 times, B-Include the keyword “resume” for 3 times	5-Highly similar (same gro…

Comment: Both pairs involve specific instructions on word usage within text. The two phrases should belong to the same group – 01 on avoiding capitalization, while 02 on repeating specific words.

P.S. Following the instruction to ignore the sentiment of the feedback, *wrote a haiku about missing classes, failing to follow the poem structure* is reduced to *wrote a haiku about missing classes*

ID	Input Pairs	Expected Annotation
03	A-avoid using commas	4- Relevant (sister groups)

Figure 13: Annotation Guideline: Phrase Similarity Annotation Instructions Page 1

ID	Input Pairs	Expected Annotation
	B-exclude the words "can" and "ride" in the written response, but "ride" appeared twice	
04	A-write 3 to 5 paragraphs B - (not) stay within the 100-word limit, going 122 words over	4- Relevant (sister groups)

Comment: Both pairs involve specific instructions to restrict certain elements in writing — 03 avoiding punctuation or keywords and 04 limiting paragraph or word count. These user requests are highly relevant but not identical, hence "sister groups."

P.S. Following the instruction to ignore the sentiment of the feedback, *exclude the words "can" and "ride" in the written response, but "ride" appeared twice in the response* is reduced to *exclude the words "can" and "ride" in the written response*

ID	Input Pairs	Expected Annotation
05	A-wrote the poem without capitalization B-writing lyrics that are less cliche	3-Neutral
06	A-finished writing the email B-wrote the entire response in English	3-Neutral

Comment: Both pairs contain instructions related to style or language use, but lack clear connections. The tasks are broadly related, resulting in a "neutral" rating.

ID	Input Pairs	Expected Annotation
07	A-write an academic proposal, B-explain the calorie content of almonds	2-Weakly related (leaning …
08	A-focus on making the tone of statements sound more like a formal announcement, B-included more information, such as a product image and name	2-Weakly related (leaning …

Comment: Both tasks demand distinct content and presentation approaches. However, they are distantly related because ID 07 involves the concept of preparation, and ID 08 focuses on enhancing information delivery.

ID	Input Pairs	Expected Annotation
09	A-exclude commas from the output, B-explain the calorie content of almonds	1-Totally unrelated (never …
10	A-writing quality that is repetitive and clunky B-use the proper markdown syntax for italics	1-Totally unrelated (never …

Comment: Both pairs involve completely different writing elements—one addresses content or style, while the other focuses on format or syntax. These tasks are unrelated in their objectives and techniques.

Figure 14: Annotation Guideline: Phrase Similarity Annotation Instructions Page 2