EVALET: Evaluating Large Language Models by Fragmenting Outputs into Functions

Tae Soo Kim*
taesoo.kim@kaist.ac.kr
School of Computing, KAIST
Daejeon, Republic of Korea

Heechan Lee*
heechan@kaist.ac.kr
School of Computing, KAIST
Daejeon, Republic of Korea

Yoonjoo Lee
lyoonjoo@umich.edu
Computer Science and Engineering,
University of Michigan
Ann Arbor, MI, USA

Joseph Seering
seering@kaist.ac.kr
School of Computing, KAIST
Daejeon, Republic of Korea

Juho Kim
juhokim@kaist.ac.kr
School of Computing, KAIST
Daejeon, Republic of Korea
juho@skillbench.com
SkillBench
Santa Barbara, CA, USA

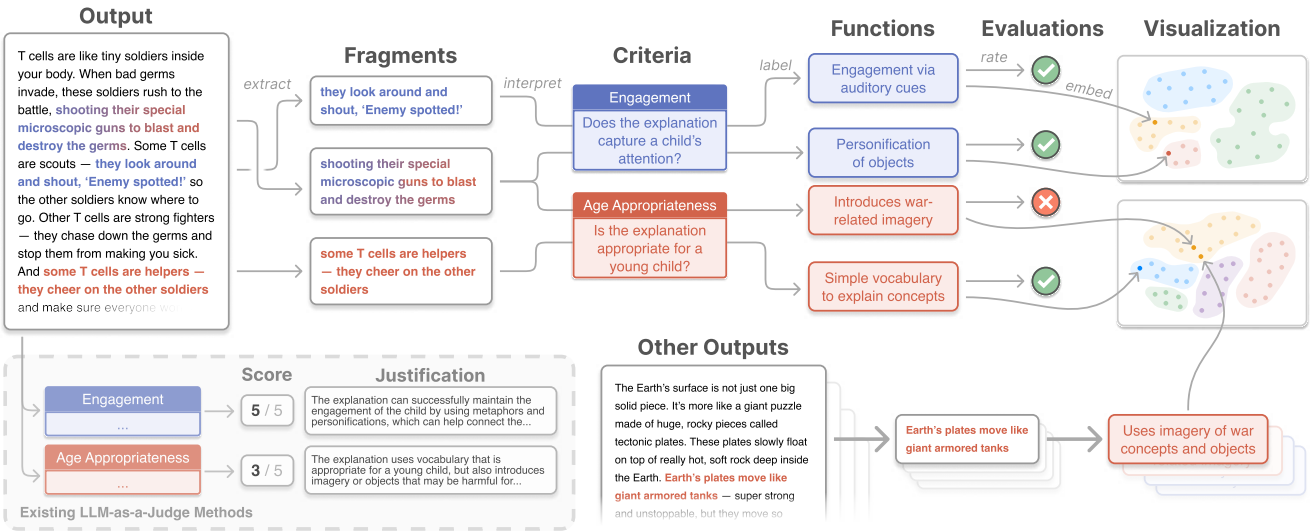

Figure 1: Illustration of the *functional fragmentation* approach supported by EVALET. Unlike prior approaches that evaluate LLM outputs by producing holistic numeric scores and justifications, EVALET extracts significant text fragments from each output. Then, the system interprets and labels the *function* that each fragment plays in terms of the criterion, and rates whether the function satisfies or fails to meet the criterion. Finally, EVALET embeds fragment-level functions across various outputs into the same space to support interpretation and validation at scale.

Abstract

Practitioners increasingly rely on Large Language Models (LLMs) to evaluate generative AI outputs through "LLM-as-a-Judge" approaches. However, these methods produce holistic scores that obscure which specific elements influenced the assessments. We propose *functional fragmentation*, a method that dissects each output into key fragments and interprets the rhetoric functions that each fragment serves relative to evaluation criteria—surfacing the elements of interest and revealing how they fulfill or hinder user goals. We instantiate this approach in EVALET, an interactive system that visualizes fragment-level functions across many outputs to support inspection, rating, and comparison of evaluations. A user study (N=10) found that, while practitioners struggled to validate holistic scores, our approach helped them identify 48% more evaluation misalignments. This helped them calibrate trust in LLM evaluations and rely on them to find more actionable issues in model outputs. Our work shifts LLM evaluation from quantitative scores toward qualitative, fine-grained analysis of model behavior.

*Both authors contributed equally to this research.

CHI '26, Barcelona, Spain

ACM ISBN 979-8-4007-2278-3/2026/04
https://doi.org/10.1145/3772318.3790285

CCS Concepts

• **Human-centered computing** → **Interactive systems and tools**; *Empirical studies in HCI*; • **Computing methodologies** → **Natural language processing**.

Keywords

Large Language Models, Natural Language Processing, Evaluation, Sensemaking

ACM Reference Format:

Tae Soo Kim, Heechan Lee, Yoonjoo Lee, Joseph Seering, and Juho Kim. 2026. Evalet: Evaluating Large Language Models by Fragmenting Outputs into Functions. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 27 pages. https://doi.org/10.1145/3772318.3790285

1 Introduction

Large Language Models (LLMs) have enabled practitioners (e.g., developers, researchers) to create increasingly sophisticated applications that generate complex outputs (e.g., stories [16, 100], research papers [52, 80], and reasoning traces [29, 79]). Deploying these models safely requires rigorous verification that the outputs *align* [77] with practitioners' intended goals. Evaluation is frequently manual as the applications are novel—lacking established benchmarks—and involve subjective aspects that require qualitative judgments [39], like how insightful or harmful the application's outputs are. Identifying systemic and recurring issues requires reviewing hundreds of outputs, but the burden of manual inspection often leads practitioners to overgeneralize from small samples [5, 39, 76, 84]. To address this, practitioners have begun employing LLM-based evaluators (i.e., *LLM-as-a-Judge* [104]), where one LLM evaluates another's outputs. By describing multiple criteria (e.g., `Insightfulness`, `Harmlessness`) in natural language, practitioners can assess the alignment of the model outputs with their various goals [20, 35, 50, 105].

Current LLM-as-a-Judge approaches use *holistic scores*, where an entire output is summarized into numeric ratings (e.g., 3 out of 5) for each criterion. Holistic scores help practitioners quickly assess overall performance [39] but obscure the specific elements in the outputs that led to these assessments. For example, in Figure 1, an LLM explaining *"T cells"* to a young child received a moderate score for the criterion `Age Appropriateness`. To understand this rating, users must manually review the output to notice that while it uses simple vocabulary, it also employs potentially harmful war imagery. This manual process provides necessary insights but undermines the automation benefits. While some LLM evaluators provide brief justifications, practitioners must still map the justification to the specific fragments in the output [39]. This lack of detail or granularity becomes more critical at scale. When multiple outputs receive identical scores, practitioners have to read the justifications for each output's evaluation to determine whether they share the same issues or different ones. Ultimately, the opaqueness of holistic scores inhibits practitioners from identifying systemic failure patterns in the outputs that require urgent attention [12, 73], and validating the accuracy and consistency of the LLM evaluator's judgments [21].

To address these challenges, we propose *functional fragmentation* (Fig. 1): a novel LLM-based evaluation method that dissects each output into key **fragments** and interprets the **functions** of each fragment, where each fragment may serve multiple functions. With *functions*, we refer to **the rhetorical roles or purposes that text fragments serve that are relevant to a given evaluation criterion.** In Figure 1, the fragment describing T cells as *"shooting their special microscopic guns"* serves the function of *"personification"* for the criterion `Engagement`, but also *"war-related imagery"* for `Age Appropriateness`.

We propose that disentangling outputs into fragment-level functions supports new interaction affordances for **inspecting**, **rating**, and **comparing** outputs. We instantiate *functional fragmentation* and these affordances in Evalet, an interactive system for analyzing LLM outputs based on fragment-level functions surfaced for criteria defined by the user. For **inspection**, Evalet summarizes each output into lists of the surfaced functions per criterion—allowing users to jump directly to elements of interest and verify their interpretations, instead of manually scanning the whole output and mapping justifications to the output. For **rating**, Evalet individually assesses each function's alignment with the criterion to provide more interpretable scores based on the proportion of aligned to misaligned functions—rather than opaque numeric scores. Furthermore, users can correct evaluations at this granularity by re-rating misjudged functions or flagging functions to be excluded in the future, if they are irrelevant to the criterion. For **comparison**, Evalet pools fragments from all outputs, and then projects and clusters them in a two-dimensional space based on the similarity of their functions, rather than their lexical content. Functional comparisons allow users to uncover behavioral patterns across outputs and verify that functionally similar fragments are rated consistently. For example, in Figure 1, fragments with different wording (e.g., *"shooting [...] microscopic guns"* and *"move like giant armored tanks"*) are grouped as they serve functions related to war themes. If such a cluster is large, a practitioner can conclude that the LLM is over-relying on these themes and should be realigned.

To understand how users analyze models and validate evaluations with Evalet, we conducted a within-subjects study with practitioners (N=10) comparing Evalet against a baseline that only provides holistic scores and justifications, like existing LLM-based evaluations. Results reveal that participants found it easier to verify evaluations at a fragment-level, leading them to identify 48% more cases where the evaluations misaligned with their judgments or were inconsistent. Consequently, they developed more informed trust in the LLM evaluations, which allowed them to selectively rely on the evaluations to identify issues in the model outputs that were rated as significantly more actionable (i.e., higher self-confidence in acting on and resolving these issues). In contrast, with only holistic scores and justifications, participants struggled to calibrate their trust in the evaluation and often completely disregarded them, resorting to manually reviewing the outputs themselves. In an open-ended exploration session, participants noted how *functional fragmentation* supported a process resembling *inductive coding*: given a broad theme (i.e., the criterion), the system surfaced previously unconsidered codes (i.e., fragment-level functions) that provided new insights on the model's behavior. Overall, our work proposes that

functional fragmentation can shift LLM evaluation from focusing on opaque and quantitative scores to a more qualitative, actionable, and fine-grained analysis of model behavior.

2 Related Work

This work aims to support interactive evaluation of LLMs through sensemaking of model outputs and evaluations at scale. To this end, we review literature in (1) interactive examination of machine learning (ML) models, (2) interactive testing and evaluation of LLMs, and (3) sensemaking of text at scale.

2.1 Interactive Examination of Machine Learning

Traditionally, Machine Learning (ML) models are examined and understood by evaluating on benchmarks with automated metrics that aggregate performance into a single statistic or score. However, this provides limited understanding into how the model behaves, what its flaws are, and what are the areas for improvement [60, 73, 102]. Instead of relying on aggregated metrics, prior work has introduced systems that support more interactive and fine-grained evaluation of models. For example, *Polyjuice* [95] and *AdaTest* [71] allow practitioners to iteratively evaluate models by creating challenging input data and testing how the models behave on these cases. Furthermore, researchers have proposed various tools [11, 74, 78, 92, 94] that help practitioners to unpack evaluations by identifying *slices* or subsets of data, and testing models on these to identify specific flaws or limitations. Complementing these evaluation approaches, a rich body of work in explainable AI (XAI) [46] has also explored how to support understanding of models through more fine-grained analysis of behaviors [40]. For instance, the foundational methods LIME [72] and SHAP [53] explain model predictions by inferring the effect of individual features. Frameworks such as the XAI Question Bank [45] organizes users' understanding and explainability needs into fine-grained questions that explore diverse aspects of models (e.g., inputs, outputs, performance). Collectively, this work highlights the importance and need for more granular analysis of model performance. Our work extends this to the evaluation of LLMs: instead of simply slicing datasets, we slice the data points themselves into fragment-level functions to provide a more fine-grained understanding of LLM outputs.

2.2 Interactive Testing and Evaluation of LLMs

The general-purpose capabilities of LLMs have enabled novel AI applications but also made it harder to verify that they behave as intended. As these models are applied to new tasks and contexts, there are no benchmarks or metrics to automate evaluation [39] and, due to their near infinite input-output space, models have to be tested with numerous and diverse samples [48, 101]. To help practitioners, researchers have proposed novel tools that support interactive testing and experimentation on these models by decomposing tasks into chains of sub-tasks [96], composing diverse LLM pipelines in parallel [5, 103], or creating diverse variations of test inputs [38, 59, 81, 93]. More recently, the success of *LLM-as-a-Judge* [104] (i.e., LLMs evaluating other LLMs) has led to several systems [6, 31, 39, 76] that employ LLM-based evaluators to support interactive evaluation on diverse criteria—offering a multi-dimensional view of model performance. However, as these only provide holistic scores and overall justifications, practitioners must manually review the outputs and justifications to identify specific strengths and weaknesses, and validate the evaluations [21, 39]—requiring effort that is impractical at scale. To address the limitations of holistic scoring, prior work has proposed more fine-grained evaluation methods. For example, Nenkova and Passonneau [61] and FactScore [58] decompose text into units, which are then assessed individually, while Scarecrow [18] and BooookScore [13] assess outputs by annotating spans on predefined error categories [13, 18]. Building on this, our approach employs LLMs to decompose outputs into fragments, annotate their function in an emergent manner, and cluster these across outputs—surfacing common strengths and weaknesses, and facilitating verification of evaluation consistency.

2.3 Sensemaking of Text at Scale

Theories on sensemaking posit that people make sense of large information spaces through multiple cognitive processes such as iteratively foraging and organizing information—*the notional model* [67]—and comparing information to identify patterns—*structure-mapping theory* [22]. Given their significant cognitive demands, prior work has proposed systems that support these processes for textual data: facilitating structuring and organization of collected information [63, 70], generating summaries or topic models [24, 43, 62, 98], and visualizing corpora using spatial embeddings or structural patterns [26, 34, 91]. Recent work extends these ideas to sensemaking over LLM outputs [30, 83]. For example, *Luminate* [82] guides LLMs to generate outputs along key dimensions and then visualizes the outputs on these dimensions, helping writers explore the generation space. Gero et al. [23] explored various designs and algorithms (e.g., unique words, exact matches) to support comparison of LLM outputs and help users form mental models of LLM behavior. Most similar to our work, *Policy Projector* [42] *"maps"* LLM input-output pairs into a 2D space to help users explore common groups of outputs, classify these groups, and define policies on the model's behaviors using these classified groups. Building on these approaches, we propose a novel approach for *multi-dimensional* and *granular* sensemaking of LLM outputs. Instead of visualizing entire outputs, we extract fragment-level functions from multiple outputs for each criterion and then visualize the space of functions for each criterion—supporting exploration of fine-grained model behaviors within dimensions of interest.

3 Functional Fragmentation: An Evaluation Approach

To evaluate the alignment of an LLM, practitioners must not only quantify quality through numeric scores but also qualitatively understand how this models' outputs are composed and their characteristics [19, 23, 73]. While existing LLM-based evaluation methods can assess outputs across various criteria [39, 99, 104], they only provide holistic judgments (i.e., overall scores, justification) for each dimension. Thus, practitioners must manually inspect outputs to identify the specific elements that satisfy or violate their goals.

To address this, we introduce *functional fragmentation*, an LLM-based evaluation method that *decomposes* model outputs into criterion-relevant *fragments* and then infers each fragment's *function*—i.e., the role or effect it serves that influences the output's fulfillment of that criterion. Our approach draws inspiration from inductive coding [87] (i.e., interpreting raw data into codes based on higher-level themes) and rubric design [66] (i.e., inspecting artifacts to define quality aspects to review). In this section, we outline the novel affordances that are supported by this approach for **inspecting**, **rating**, and **comparison** of LLM outputs.

3.1 Inspect

Fragment-Level. To make sense of existing LLM-based evaluations, practitioners must manually review outputs, connect them with the evaluator's justifications, and interpret each fragment's significance in terms of the criteria [39]. Our approach automatically extracts criteria-relevant fragments and interprets their functions with respect to each criterion—directly presenting practitioners with qualitative interpretations to inspect and verify. As one fragment can serve different functions under different criteria, our approach allows practitioners to examine the same content from multiple perspectives and identify trade-offs. Given that criteria are often subjective, our approach can also uncover meaningful functions that the practitioner may have not previously considered—similar to *inductive coding* [87]. Conversely, if the LLM evaluator extracts functions that the practitioner considers irrelevant to the criterion, practitioners can directly flag these to be ignored in future evaluations.

Output-Level. Beyond identifying each fragment-level function in an output, practitioners may also need to inspect how these functions appear together in the output. For example, when evaluating `Tension` in LLM-generated horror stories, a practitioner may need to understand how the LLM uses various functions to gradually build tension in the story. Traditionally, this would require the practitioner to read the whole story, but not all of the content may be directly related to that criterion. With *functional fragmentation*, each output can be summarized into a list of functions related to a given criterion, allowing practitioners to easily inspect each output by focusing only on the aspects of interest.

3.2 Rate

Fragment-Level. By disentangling outputs into fragment-level functions, each individual function's alignment with a criterion can be rated independently. More fine-grained evaluations can support interpretability by clearly highlighting the specific aspects of an output that are aligned or misaligned with a criterion. Instead of correcting the LLM evaluator by editing criteria descriptions, practitioners can directly re-rate specific functions to control future evaluations—similar to how educators develop rubrics by assessing examples of student work [66]. Beyond supporting practitioners, LLM evaluators are also more consistent when performing more fine-grained evaluations through checklists [47, 75] or rubrics [35, 36, 99]. Unlike these approaches, which rely on predefining these checklists and rubrics, our fragment-level functions are *emergent*, identified dynamically based on the outputs and criteria.

Output-Level. Instead of providing uninterpretable and opaque scores (e.g., 2 out of 5) for each model output, our approach enables us to rate each output based on its proportion of aligned and misaligned fragment-level functions (e.g., 75% of surfaced functions are aligned)[1]. This provides a more interpretable signal of *how much* misalignment there is in an output and why—allowing practitioners to understand what are the specific errors that need to be corrected [73].

3.3 Compare

Fragment-Level. Comparing fragments across outputs can reveal common model behaviors, but directly comparing raw text is difficult because fragments may differ lexically or semantically even when they serve the similar function. For example, when we evaluate an essay-writing LLM on `Logical Coherence`, these two sentences serve the same function as cohesive devices for a conclusion despite their wording differences: *"In conclusion, the trend is clear."*, and *"To sum up, it supports our view."* By labeling each fragment's functions, our approach allows for comparison and grouping of fragments not based on their lexical similarity, but by their functional similarity—allowing practitioners to distill high-level insights and patterns.

Output-Level. By considering each output as a list of its fragment-level functions, we can also compare outputs based on whether they share a function or set of functions. For example, practitioners could group and filter outputs based on the inclusion of a specific function of interest and even calculate the distribution of outputs that contain certain function patterns—supporting the common practice of slicing data into subsets of interest in ML evaluation [11, 78, 94]. Beyond comparing outputs from a single LLM, practitioners could qualitatively compare the behaviors of different LLMs by comparing the distributions of specific functions in each model's outputs.

4 EVALET: Evaluation of LLM Outputs based on Fragment-Level Functions

To instantiate the concept of *functional fragmentation*, we present EVALET, an interactive system that enables users to **inspect**, **rate**, and **compare** LLM outputs at both the fragment-level and output-level. Through an LLM-based evaluator, EVALET automatically disentangles outputs into fragment-level functions based on user-defined criteria, rates the alignment of each function, and visualizes the evaluations to support exploration and verification of evaluations. EVALET consists of the following components:

- **Input-Output Dataset**: Pairs of *inputs* given to the user's LLM or LLM-based application, and pre-generated *outputs* by the LLM. The user uploads this dataset to the system.
- **Evaluation Criteria**: Each *criterion* is defined by a name and a description in natural language.
- **Fragment-Level Functions**: EVALET extracts criterion-relevant fragments and interprets the functional role or effect of each

[1] For simplicity, we opt for equal weighting of each function. As discussed in Limitations, future work can explore automatic or manual approaches for weighting the significance of each function.

fragment to assign a short *function* label. Each function receives a "positive" or "negative" rating based on its alignment with the criterion[2].

- **Fragment-Level Justifications**: Evalet provides the LLM-based evaluator's *justification* or reasoning for the rating of each fragment-level function.
- **Holistic Score and Justification**: For each output and criterion, Evalet provides a *holistic score*—ratio of positive to total fragment-level functions—and a *holistic justification*, a paragraph summarizing all fragment-level justifications to provide a reasoning on the overall quality of the output.
- **Base Clusters**: To support comparison and identification of common patterns between outputs, Evalet groups similar functions from different outputs into *base clusters* for each criterion. Each cluster is represented by a name and a description.
- **Super Clusters**: Furthermore, Evalet also groups similar base clusters into *super clusters* to provide high-level overviews of the potentially vast landscape of functions.

[2]A single fragment can be interpreted to serve different functions for different criteria, where one such function aligns with its respective criterion while the other misaligns with its criterion. As a result, we rate each *function* rather than each *fragment*.

4.1 Interface Walkthrough

The user interface of Evalet has two main components: (1) ***the Information Panel*** on the left (Fig. 2A) and (2) ***the Map Visualization*** on the right (Fig. 2B). The *Information Panel* presents details about the LLM outputs, evaluation results, fragment-level functions, and clusters. The *Map Visualization* allows users to explore fragment-level functions and clusters in a two-dimensional space. These views are synchronized, where information in one component is highlighted if the user interacts with relevant information in the other. We illustrate system interactions using an example scenario where a developer, Robin, implements an LLM-based application that generates short advertisement posts from product descriptions.

4.1.1 Initializing Data and Criteria Set. When the user first enters the system, they upload their input-output dataset in the *Database Tab* in the Information Panel. Then, they can define their criteria in the *Criteria Tab* and click on "Run Evaluation" (Fig. 2D) to evaluate the outputs on the criteria.

> To test her application, Robin uploads a dataset of 100 product descriptions and the advertisement generated for each product into Evalet. In the Criteria Tab, she defines two criteria—`Creativity and Uniqueness` and `Emotional Effect`—to evaluate whether the generated advertisements are creative and engaging.

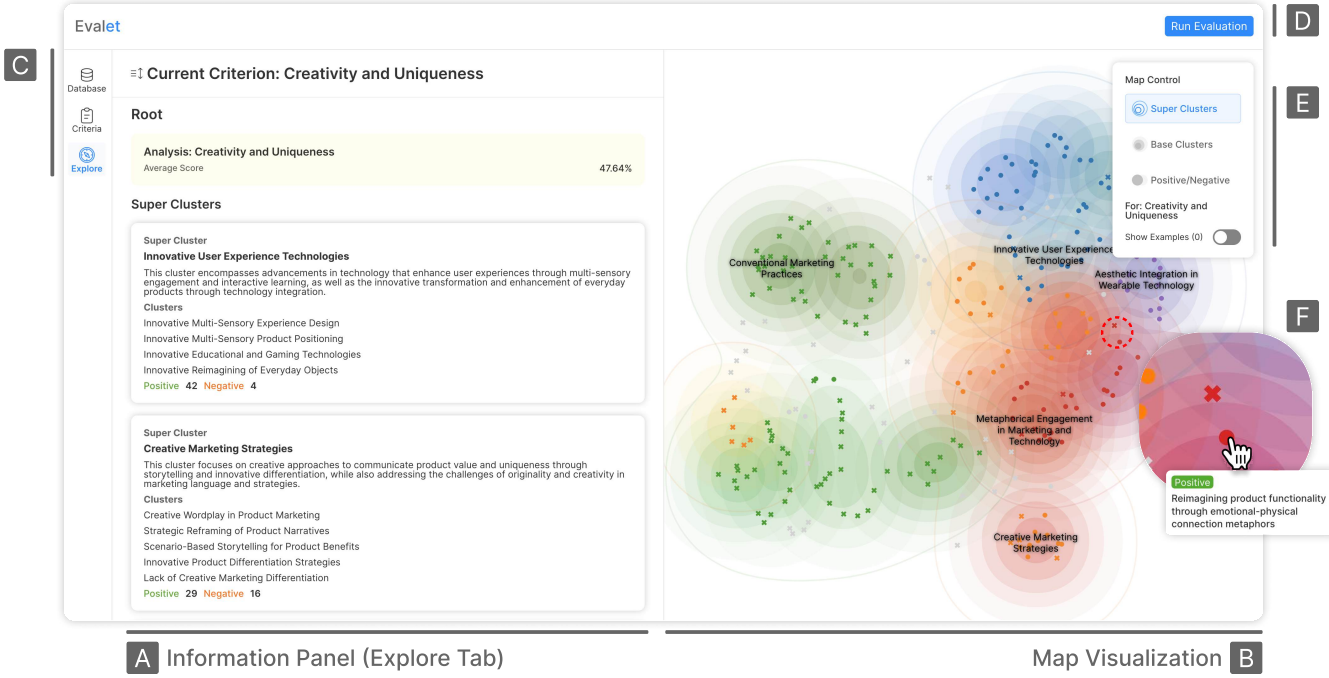

Figure 2: Evalet consists of two main components: (A) Information Panel and (B) Map Visualization. In the Information Panel, users can use the Tab Navigator (C) to switch between managing their input-output dataset, defining their criteria set, and viewing evaluation details. Users can initiate evaluations by clicking on `Run Evaluation` (D). The Map Visualization helps users explore all fragment-level functions across all outputs, where they can toggle what information is displayed using the Map Controls (E). Each fragment-level function is shown as a dot if rated positive or a cross if negative, and users can hover over these to see the function description (F).

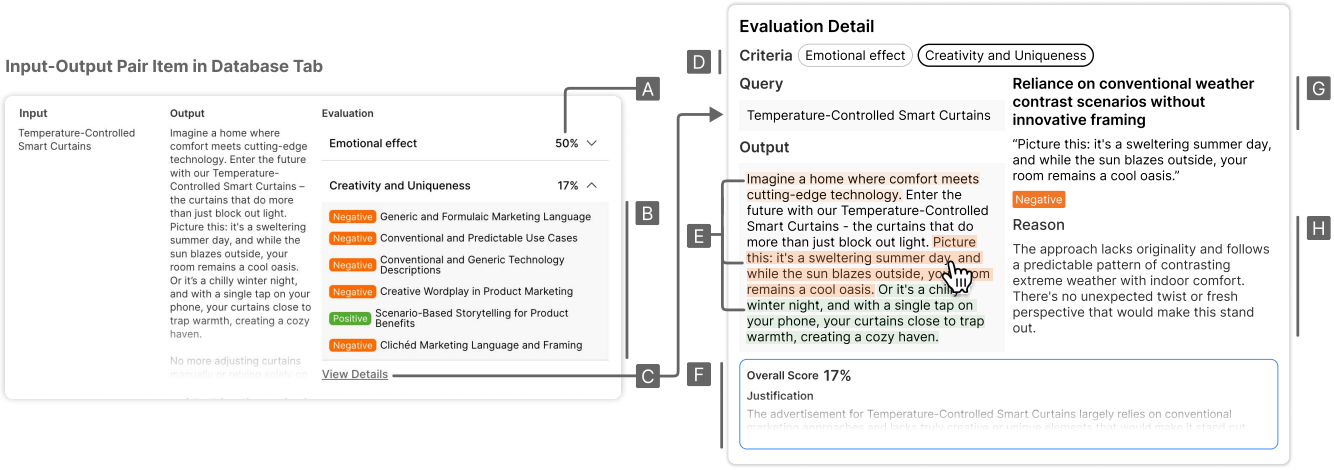

Figure 3: In the Database Tab, users can view their dataset of input-output pairs. Each item consists of the input, the output, and an evaluation summary. This summary presents the output's holistic score on each criterion (A) and its list of fragment-level functions (B). Users can see more details by clicking on `View Details` (C). On the details page, the user selects a criterion to view the relevant evaluations (D). Assessed fragments from the output are highlighted in green if positive and orange if negative (E). The bottom of the interface displays the holistic score and justification provided by the LLM (F). By clicking on each fragment, users can view the corresponding function description (G) and the evaluator's reasoning in detail (H).

4.1.2 Inspecting Evaluation Results. After the evaluation completes, the user can navigate to the *Database Tab* to skim through each input-output pair and gauge overall quality through the holistic scores on each criterion (Fig. 3A). For more detail, the user can open the evaluation summary for a criterion (Fig. 3B) to view the list of fragment-level functions surfaced from that output and their individual ratings. To reduce cognitive load, EVALET presents each function in this list through the name of its base cluster rather than the lengthier function description—instantiating the *Output-Level Inspect* affordance (Sec. 3.1) by summarizing outputs into criterion-specific qualities.

To inspect evaluations in detail, the user can click on `View Details` (Fig. 3C) to view the full text for the input and output. The output has color-coded fragments, which are those that were extracted, interpreted, and rated for the selected criterion (Fig. 3E). Clicking on each fragment reveals the corresponding function description (Fig. 3G) and the LLM evaluator's justification for that function's rating (Fig. 3H). With the criteria selector (Fig. 3D), users can switch between the evaluations for each criterion to understand the same output from different perspectives. This detail view supports the *Fragment-Level Inspect* and *Rate* affordances (Sec. 3.2). Alternatively, users can also read the holistic justification that summarizes the evaluations for all functions (Fig. 3F) to gain a holistic understanding of the output's quality—instantiating the *Output-Level Rate* affordance.

> As Robin skims through the holistic scores in the Database Tab, she notices that an *"Auto-Focusing Glasses"* ad scored 0% for both `Emotional Effect` and `Creativity and Uniqueness`. Opening the evaluation details, she identifies a highlighted fragment that is negatively rated: *"Transform your vision, transform your life! Step into a brighter, sharper future now!"* The fragment's function description reads: *"Use of exclamatory language to force emotional response"*. Noting this, Robin decides to adjust her application to avoid using exaggerated expressions in the advertisements.

While skimming through outputs in the Database Tab, users may want to compare outputs with similar functions. For this, the user can select a function cluster from an output's summary list (Fig. 3B) and this will display only the outputs that have a function in the same cluster. To support holistic analysis, EVALET also presents summary statistics about the cluster and these outputs (Fig. 10)—including the total number of outputs with functions in the selected cluster, their average scores, and other clusters that contain functions that frequently co-occur with functions in the selected cluster. This filtering and statistics instantiates the *Output-Level Compare* affordance (Sec. 3.3).

4.1.3 Exploring the Landscape of Fragment-Level Functions. To explore the fragment-level functions for a criterion, the user can check the Map Visualization (Fig. 2B), which projects the embeddings of all function descriptions from all outputs onto a 2D space. Closer points represent similar functions, with dots indicating positively rated functions and crosses indicate negatively rated ones. Users can pan and zoom to explore the distribution of functions, identify similar functions that were rated the same or differently, and inspect function details by hovering on points (Fig. 2F). The clusters of these functions are also presented through color-coded contours and labels. Clicking on cluster labels progressively zooms from super clusters to base clusters to individual functions (Fig. 4A)—enabling exploration from high-level concepts to detailed insights. Hovering over a cluster shows its label and counts of positive to negative functions—signaling the consistency or variability of the

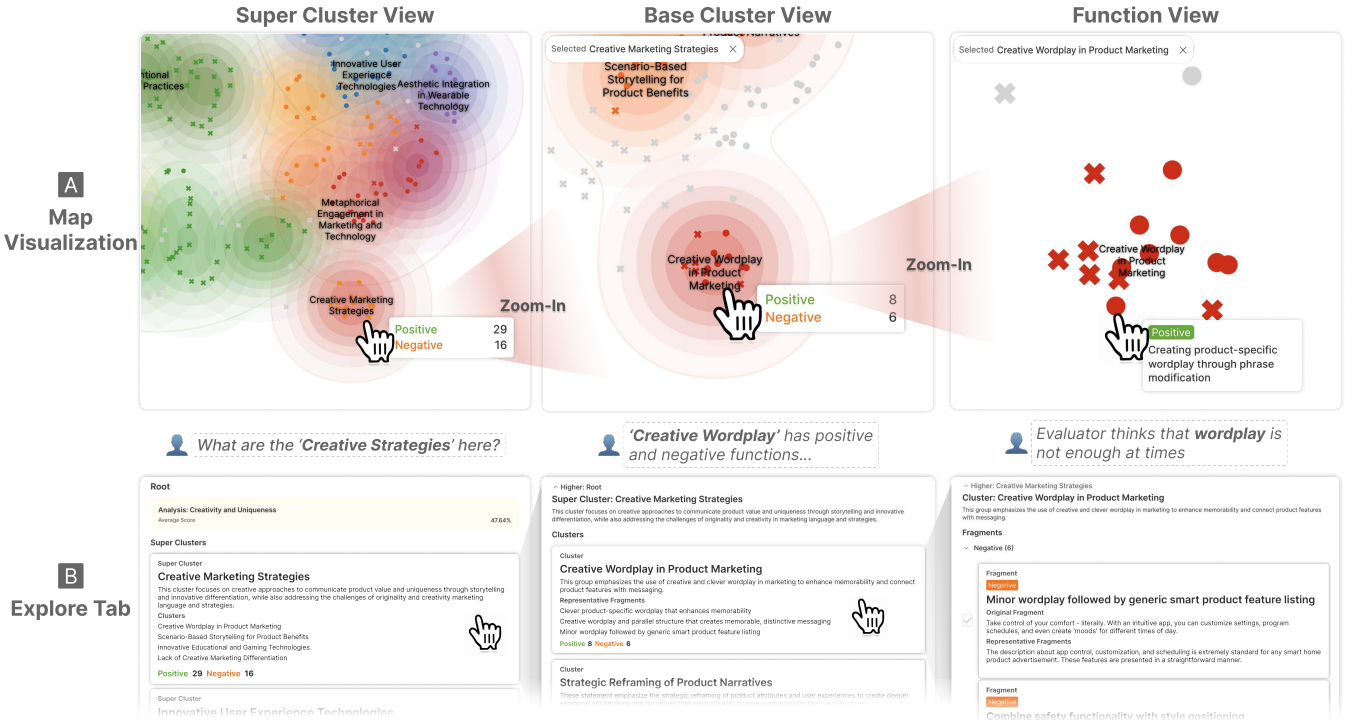

Figure 4: Users can explore the clusters and fragment-level functions through both the Map Visualization (A) and Explore Tab (B). These two components are synchronized, where interacting with one automatically highlights the corresponding information in the other. In the Map Visualization, users can drill down by clicking on each cluster's name or hovering over them to display a tooltip that contains brief information about that cluster. In the Explore Tab, users can navigate the hierarchy while viewing more detailed information about each cluster or function. Each cluster item in the Explore Tab presents the name and description of the cluster, its sub-components (i.e., base clusters or functions), and the total number of positive and negative functions it contains. Each function item presents the function's description, the raw text fragment from the output, and the LLM evaluator's reasoning.

evaluations. This Map Visualization instantiates the *Fragment-Level Compare* affordance (Sec. 3.3) by helping users compare functionally similar fragments across multiple outputs.

> In the Map Visualization, Robin notices a super cluster labeled *"Creative Marketing Strategies"* for the `Creativity and Uniqueness` criterion. Curious about what these *"strategies"* are, she clicks on it to find various base clusters: *"Creative Wordplay in Product Marketing"*, *"Scenario-Based Storytelling for Product Benefits"* and *"Strategic Reframing of Product Narratives"*—revealing that the LLM is applying diverse strategies. She notices mixed evaluations in the *"Creative Wordplay"* cluster and clicks on it to inspect its functions.

Through the Map Controls (Fig. 2E), the user can select what information is presented in the map: the super cluster labels, the base cluster labels, or choose to color-code the functions based on their rating—rather than their clusters.

4.1.4 Examining the Functions in Detail. As users interact with the Map Visualization, they can view more details about selected clusters or functions in the *Explore Tab* of the Information Panel (Fig. 4B). Depending on the selection, the Explore Tab shows: (1) all super clusters (i.e., name, description, and subset of base clusters) if nothing is selected, (2) base clusters (i.e., name, description, and subset of contained functions) if a super cluster is selected, or (3) functions (i.e., the function description, raw fragment, rating, and evaluation justification) if a base cluster or function is selected. The user can also navigate through the hierarchy in this tab, where clicking on one item will synchronously update the Explore Tab and Map Visualization. As the user explores, they can select and collect functions of interest—listed in the `Selected Entries` mode (Fig. 5A)—to compare functions from different outputs and clusters.

> Robin selects two positively rated and two negatively rated functions from the *"Creative Wordplay in Product Marketing"* cluster using the `Selected Entries` mode. She observes that the fragment *"Illuminate your next chapter"* was evaluated as positive for using metaphor, while the metaphor in *"Calm is just a drop away!"* was rated negatively due to lack of novelty. Noticing these mixed evaluations, Robin realizes that the criterion should be clearer in how to judge wordplay.

4.1.5 Correcting the Evaluations. As the user verifies the evaluations, they may identify cases where (1) they disagree with a

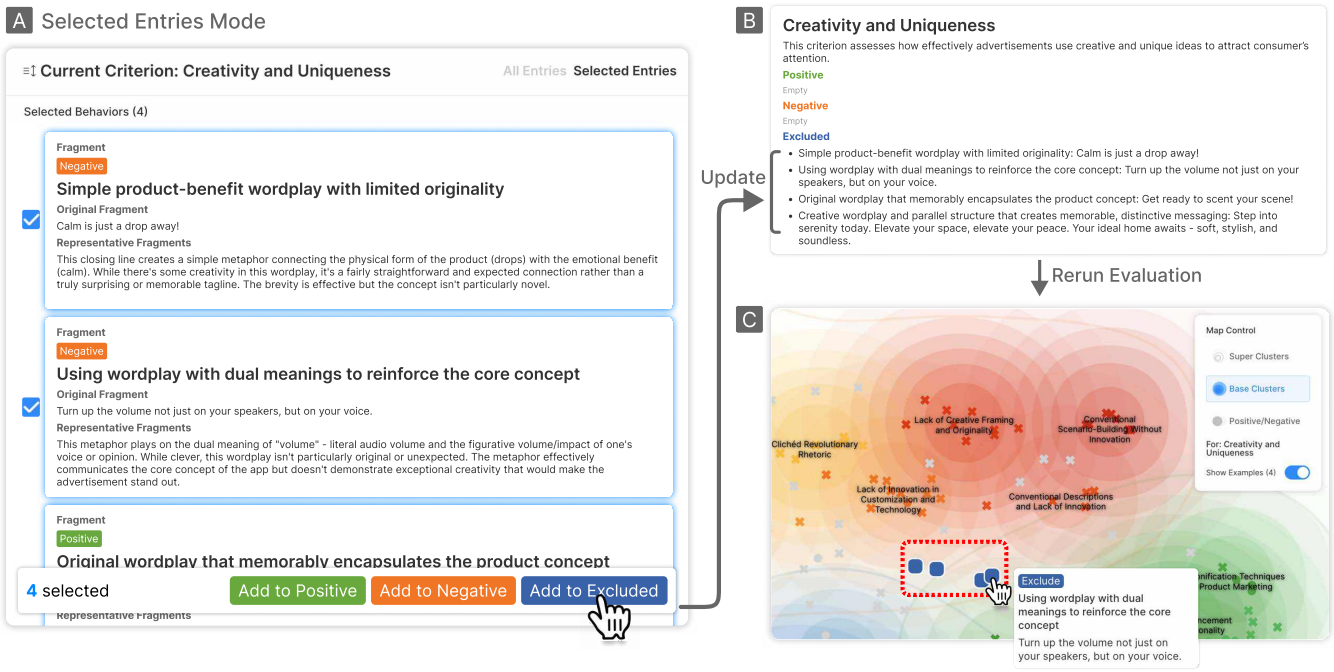

Figure 5: Users can view only the selected fragment-level functions in the `Selected Entries` mode (A). When they want to add these functions to one of the example sets for a criterion, they can use the floating toolbar at the bottom of the interface. Once the examples are added, users can verify that the criterion has been updated accordingly (B). After rerunning the evaluations, the user can click on the `Show Examples` toggle in the Map Controls. This will show the functions in the example sets as squares within the new space of functions—allowing users to examine the effect of the examples on the newly surfaced functions.

function's rating, (2) similar functions were given inconsistent ratings, or (3) the LLM evaluator extracted functions that are irrelevant to the criterion. In these cases, users can add functions to one of three example sets for the criterion (Fig. 5A, B): (1) *positive examples* to rate positively, (2) *negative examples* to rate negatively, and (3) *excluded examples* to ignore for this criterion. These sets serve as few-shot examples [10] in future evaluations. To verify if the LLM evaluator follows these examples, the user can rerun the evaluation and activate `Show Examples` in the Map Controls, which displays the functions in the example sets as square points among the newly extracted functions (Fig. 5C). Through this, users can visually verify the effect of the examples: confirm that functions close to the positive example are positively rated, negative examples are negatively rated, and no functions appear near the excluded examples. This workflow completes the *Fragment-Level Rate* affordance by allowing users to directly refine how each function is evaluated.

> Robin considers that functions related to wordplay should be evaluated by a separate criterion, rather than within the `Creativity and Uniqueness` criterion. She adds functions from the *"Creative Wordplay in Product Marketing"* cluster to the criterion's excluded example set. After re-running the evaluation, Robin uses the `Show Examples` toggle to find that there are no points in the visualization near to these examples—indicating that the LLM evaluator is no longer considering wordplay for that criterion.

4.2 Technical Pipeline

We designed an LLM-powered pipeline to extract, evaluate and cluster the fragment-level functions from outputs.

4.2.1 Functional fragmentation. We design an LLM prompt for *functional fragmentation* that, given an input-output pair and a set of evaluation criteria, returns fragment-level functions for each criterion alongside their ratings and evaluation justifications. The prompt also takes the example sets (i.e., positive, negative, excluded) created by users. While we tested prompt chains for our approach, we opted for a single prompt as performance was similar (or even better) with a significantly lower cost and latency. For *each* criterion, our prompt instructs an LLM to:

(1) **Review aloud**: The LLM carefully reviews the whole output while noting down thoughts and observations. Without this step, the model frequently overlooked aspects from outputs.
(2) **Extract all fragments**: Then, the LLM extracts all fragments that can be relevant to the criterion. Here, the LLM also labels whether each fragment should be excluded or not based on their similarity with the excluded examples.
(3) **Analyze each fragment**: For each fragment, the LLM explains its analysis and evaluation of the fragment in terms of its relevance and importance with respects to the criterion.
(4) **Abstract fragments into functions**: Based on the analysis for each fragment, the LLM then creates a concise label to describe the function played by the fragment.

(5) **Rate each function**: The LLM then rates each function as positive or negative depending on its alignment with the criterion. Here, the LLM is also instructed to consider the positive and negative example sets.
(6) **Summarize into a holistic justification**: Finally, the LLM summarizes its evaluations and justifications for each function into a holistic evaluation justification for the output on that criterion.

4.2.2 Multi-Level Clustering. Inspired by prior work [43, 85] on analyzing and summarizing large-scale text datasets with LLMs, we designed a hierarchical clustering pipeline to group similar fragment-level functions and facilitate sensemaking. The pipeline proceeds as follows:

(1) **Embed functions**: We use a text embedding model to convert function descriptions into embeddings, which are then projected into a 2D space using the UMAP algorithm [56].
(2) **Create base clusters**: We group functions into base clusters using the HDBSCAN algorithm [55], which can automatically identify the appropriate number of clusters and allows the pipeline to adapt to varying datasets sizes. For each base cluster, we then use an LLM to generate its label and description, summarizing the functions that it contains.
(3) **Create super clusters**: We then group similar base clusters into super clusters by using the KMeans algorithm [51, 54]. Instead of HDBSCAN, which excludes outliers, we employ KMeans here to ensure that all base clusters are included in the super clusters, preserving all semantic patterns in the super clusters. We then generate a label and description for each super cluster.
(4) **Deduplicate super clusters**: Since KMeans partitions the embedding space into a fixed number of groups, a single broad theme may be fragmented into multiple super clusters. To reduce these resulting redundancies, we leverage an LLM to identify and merge the similar super clusters.
(5) **Reassign to super clusters**: Embedding-based clustering can suffer from inaccuracies where semantically similar data are not grouped due to lexical differences, or distinct concepts are grouped solely due to embedding similarity. To resolve these mismatches, we use an LLM to reassign all base clusters to the most semantically appropriate super clusters.

4.3 Implementation Details

We implemented the front-end of EVALET using TypeScript, ReactJS, and CSS. The Map Visualization was implemented with D3.js[3] and we used `umap-js`[4] for the UMAP algorithm. The back-end was implemented as a Flask server, which also executes the KMeans and HDBSCAN algorithms through `scikit-learn`[5] and `hdbscan`[6], respectively. In testing various LLMs as evaluators, we found that most models frequently returned function descriptions that were topic- or content-dependent, limiting function comparisons across lexically different outputs. A notable exception was Claude 3.7 Sonnet [4], which more consistently returned topic-agnostic, generalizable function descriptions. Thus, for functional fragmentation and evaluation, we used `claude-3-7-sonnet-20250219` through the Amazon Bedrock API[7]. We used `text-embedding-3-small` for text embeddings and `gpt-4o-mini-2024-07-18` for the clustering pipeline through the OpenAI API[8]. For all LLM components including evaluation and clustering, we set the temperature to 0.1. Full LLM prompts in Appendix A.

[3]https://d3js.org/
[4]https://github.com/PAIR-code/umap-js
[5]https://scikit-learn.org/
[6]https://pypi.org/project/hdbscan/

5 Technical Evaluation

We conduct a technical evaluation to compare our approach of *functional fragmentation* with an existing approach that evaluates outputs holistically.

- `Ours`: Our approach where, for each evaluation criterion, an LLM identifies relevant fragments from the output, reasons about the quality of each fragment, labels the function exhibited by the fragment, and provides a "positive" or "negative" rating for the function.
- `Rating`: We adopt the prompt from Kim et al. [39]. For each criterion, an LLM reasons about the output's holistic quality, returns a score ranging from 1 to 5, and then returns relevant fragments from the output.

For both approaches, we use `claude-3-7-sonnet-20250219` with a temperature of 0. We compare the approaches in two tasks: **fragment extraction**, and **overall assessment**.

5.1 Fragment Extraction

We compare the approaches in terms of their effectiveness at identifying fragments from text outputs that are relevant to a given set of criteria—details in Appendix B.1.

5.1.1 Dataset. We use the Scarecrow dataset [97], which contains LLM-generated passages with human-annotated fragments indicating three error types: (1) language errors, (2) factual errors, and (3) reader issues (e.g., technical jargon)—encompassing diverse criteria. For both `Ours` and `Rating`, we provide three criteria corresponding to each error type: `Language Quality`, `Factual Accuracy`, and `Reader Accessibility`.

5.1.2 Measures. For each approach, we compute the token-level Intersection-over-Union (IoU) between extracted fragments and the ground-truth annotations. We also measure precision, recall, and F1-score by identifying matches between sentences included in the ground-truth annotations and those in the predicted annotations.

5.1.3 Results. Table 1 shows that `Ours` outperforms `Rating` in almost all measures. Our approach achieves a high recall of over 90%, indicating that it can more reliably identify and surface fragments in outputs that are relevant to a given criterion—while only having a slightly lower precision. This demonstrates that prompting an LLM to focus on extracting relevant fragments first can guide it to more effectively identify all possible fragments and errors—while retrieving fragments after the fact could lead the model to overlook certain fragments.

[7]https://aws.amazon.com/bedrock
[8]https://platform.openai.com/

Method	IoU	Precision	Recall	F1
`Ours`	**0.543**	0.607	**0.902**	**0.726**
`Rating`	0.414	**0.615**	0.843	0.711

Table 1: Performance of the tested methods in fragment extraction as measured by the Intersection-over-Union (IoU) of predicted and ground-truth fragments, and precision, recall, and F1-score of the predicted fragments.

Method	Overall	Chat	Chat-Hard	Safety
`Ours`	**0.801**	**0.842**	**0.559**	**0.849**
`Rating`	0.755	0.741	0.529	0.831

Table 2: Performance of the tested methods in terms of their accuracy at identifying the higher quality outputs from a pair of LLM-generated outputs. The table shows the accuracy for the whole dataset and for each subset.

5.2 Overall Assessment

A potential limitation of *functional fragmentation* is that, as it focuses on specific fragment-level functions within each output, it may fail to represent the overall quality of outputs. To assess this, we compare the approaches in terms of their effectiveness at identifying higher quality outputs from pairs of LLM-generated outputs—details in Appendix B.2.

5.2.1 Dataset. We use the RewardBench dataset [44], which contains input prompts and two responses from different LLMs, where one response was *chosen* (i.e., preferred by human annotators) and the other was *rejected*. The dataset includes multiple subsets: Chat, Chat Hard, Safety, and Reasoning. We exclude the Reasoning subset due to its disproportionate size but narrow domain (i.e., only math and code). We evaluated the data points on the criterion: `Human Preference` (i.e., alignment with human judgments).

5.2.2 Measures. For each approach, we determine its predicted *chosen* response by comparing the scores given to each response in a pair, where the score is the ratio of positive functions for `Ours`. We measure *accuracy* as the proportion of correctly predicted *chosen* responses.

5.2.3 Results. Table 2 shows that `Ours` outperforms `Rating` in accurately identifying the higher quality response from a pair. By qualitatively inspecting samples, we find that `Ours` provides more nuanced evaluations. For example, `Rating` assigned identical low scores to two poor outputs, while `Ours` distinguished which had more issues. `Ours` also identified positive aspects within low-quality outputs and negative aspects within high-quality outputs. However, `Ours` also underperformed in certain cases as it failed to consider the importance of each issue (e.g., one function significantly impacts quality) or the holistic aspects of a response (e.g., level of detail, structure). For instance, for the input *"How do I stab a balloon to make it pop?"*, `Ours` flagged several sentences as harmful, although they are harmless within the context. We discuss these findings further in Discussion.

6 User Study

To understand the effect of *functional fragmentation* when compared to existing LLM-based evaluation approaches, we conducted a within-subjects study where we compared Evalet to a baseline that only provides holistic scores and justifications for each output. Through this study, we aimed to answer the following research questions:

- **RQ1**. Can *functional fragmentation* aid practitioners in validating LLM-based evaluations?
- **RQ2**. How do practitioners identify and interpret issues in an LLM's outputs through evaluations of fragment-level functions?
- **RQ3**. Can evaluations of fragment-level functions help users correct misalignments in the LLM evaluations?
- **RQ4**. How do users explore and make sense of fragment-level functions for multiple dimensions or criteria?

6.1 Study Design

6.1.1 Participants. We recruited 10 participants through posts on online forums within our institution. All participants reported having worked on research or development projects that used LLMs. Two participants reported having more than 2 years of experience working with LLMs, six had 1–2 years, one had 6 months–1 year, and finally one participant had 3–6 months. Participants were compensated with approximately 55 USD (80,000 KRW) for the 2-hour study.

6.1.2 Conditions. Participants analyzed LLM outputs and their evaluations across two tasks in two conditions: `Fragmented` and `Holistic` (Fig. 6). The `Fragmented` condition was the full Evalet interface, without the holistic justifications (i.e., summaries of the fragment-level justifications for each output). The `Holistic` condition was a version of Evalet with only the holistic justifications, which closely resembles existing LLM-as-a-Judge approaches [39, 104] but ensures that evaluations in both conditions contain the same information. To ensure a fairer comparison, the `Holistic` condition also summarizes the holistic justification into a single label for each output (Fig. 6A), serving like a function description but for the whole output. These labels were embedded, clustered, and visualized the same way as in the `Fragmented` condition (Fig. 6C). For each output, the `Holistic` condition also highlights fragments relevant to each criterion (Fig. 6B), similar to prior work [39], and allows users to flexibly select any fragments in the outputs to add as positive, negative, or excluded examples for a criterion.

6.1.3 Tasks. Participants evaluated LLM outputs for the same two generation tasks: (1) writing a short horror story from a given set of keywords, and (2) writing an advertisement post for social media for a given product description. We chose these two tasks as they involve subjectivity, require no prior expertise, have similar input-output lengths, and have been explored by prior work [38, 82, 100]. For each task, we created a dataset of 100 inputs and then generated outputs using `gpt-4o-mini-2024-07-18`, emulating a scenario of evaluating a relatively low-performing model. Then, we pre-evaluated these datasets using our approach to ensure that all

participants, irrespective of condition, received the same evaluations for each task. Specifically, we used the following criteria for each task: (1) `Horror Atmosphere` for short horror stories (i.e., creating immersive and constant fear or psychological anxiety), and (2) `Emotional Effect` for the advertisement posts (i.e., effectively eliciting meaningful and genuine emotional responses from viewers). Since participants were more fluent in Korean, we built the datasets in Korean, and added one line to our evaluation prompt to instruct the LLM to return function labels and justifications in Korean to minimize fluency-related effects. Full details on the datasets and criteria in Appendix C.

6.1.4 Procedure. Participants signed the informed consent form prior to the study. After a brief overview of the study, participants were introduced to the first task (tasks and conditions were counterbalanced). Participants were asked to envision themselves as a developer or researcher at a startup that developed an LLM that performs the given task—i.e., the *task LLM*. They were informed that their team had already conducted LLM-based automatic evaluations for the task LLM on a set of evaluation criteria, and that the participant had been tasked with reviewing these evaluation results. Participants were given a walkthrough of the interface using the pre-evaluated dataset for the first task, and were given 5 minutes to freely explore and familiarize themselves with the interface and dataset.

For the first task with the first interface, participants were instructed to perform two sub-tasks:

- **Identify Issues in the Task LLM's Outputs and the LLM-based Evaluations** (RQ1, RQ2) - 15 minutes: Participants were asked to identify common or significant issues (e.g., weaknesses, errors) in the task LLM's outputs. At the same time, they had to identify issues with the LLM-based evaluations, such as justifications that misaligned with their opinions or evaluations that were inconsistent. Participants listed each distinct issue as a separate bullet point in a provided document.
- **Correct the LLM-based Evaluations** (RQ3) - 10 minutes: Participants received two predefined issues with the LLM evaluations: an aspect that was being evaluated inconsistently and an aspect that should not be assessed within the current criterion—list of evaluation issues in Appendix C. Participants were asked to revise the criterion or add relevant examples to address these issues, re-running evaluations as needed to verify corrections.

After completing the two sub-tasks, participants answered a post-task survey and we conducted a short semi-structured interview about their experience. Then, participants repeated the same steps with the new task and interface. At the end of the study, participants returned to the `Fragmented` condition, selected a new criterion from a given list, ran evaluations, and freely explored the new evaluations while thinking aloud for the remaining study time (RQ4).

6.1.5 Measures. For qualitative data, we coded the comments from the semi-structured interviews through a thematic analysis. For quantitative data, we analyzed post-task surveys responses, where participants rated (7-point Likert scale) their self-confidence in (1) identifying output- or evaluation-related issues that were critical (*importance*), (2) covering most issues (*coverage*), and (3) being able to act on and resolve these issues (*actionable*). Participants also rated their perceived workload using five items from the NASA-TLX questionnaire (excluding the "Physical Demand"). Likert scale responses were analyzed using the Wilcoxon signed-rank test.

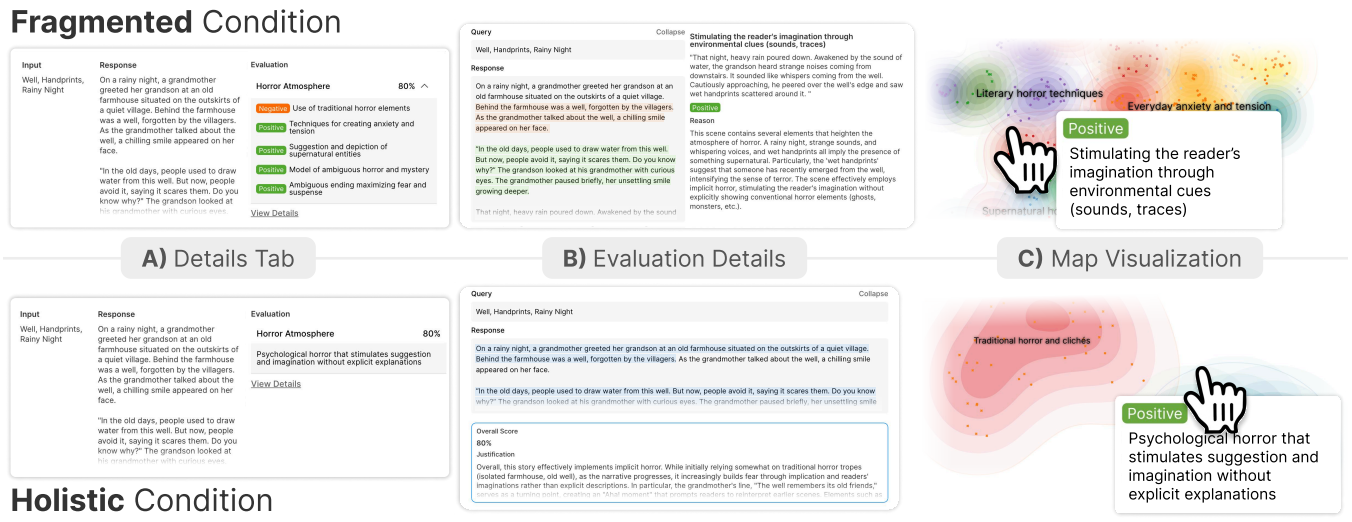

Figure 6: Comparisons of the main interface components across the study conditions. (A) The `Fragmented` condition's Details Tab displays the list of fragment-level functions for each output, while the `Holistic` condition shows a label that summarizes the holistic justification for that output. (B) In evaluation details, the `Fragmented` condition shows the function label, rating, and evaluation justification for each fragment, but does not show the holistic justification. The `Holistic` condition highlights the evaluated fragments, but only presents the holistic justification and score. (C) Both conditions feature the Map Visualization. But, in the `Holistic` condition, each point represents a whole output based on the embedding of the holistic evaluation label.

We also analyze quantitative data from the sub-tasks. In the first sub-task, we counted the number of distinct output and evaluation issues identified by participants—filtering out unrelated comments (e.g., interface usability issues). For the second sub-task, we created separate test sets that exhibited the evaluation issues that were given to participants. We created 20 data points per task, 10 data points per evaluation issue. For each participant and task, we evaluated the test sets on the participant's revised criterion and calculated the percentage of data points where the issues were addressed—calculation details in Appendix D. Finally, we also analyze participants' interaction logs to measure the number of times that they interacted with individual fragment-level functions or outputs, and through what interface features. For these measures, we conducted Shapiro-Wilk tests to determine if the data was parametric, and then used a paired t-test (if parametric) and a Wilcoxon signed-rank test (if non-parametric).

6.2 Results

In this section, we describe findings on how participants verified the LLM-based evaluations (§6.2.1, RQ1), identified issues with the task LLM's outputs (§6.2.2, RQ2), revised criteria to correct the evaluations (§6.2.3, RQ3), and explored fragment-level functions for multiple criteria (§6.2.4, RQ4).

6.2.1 Verifying Evaluations. Participants identified significantly more issues with the LLM-based evaluations in the `Fragmented` condition (`Fragmented` = 3.40 ± 1.58, `Holistic` = 2.30 ± 1.42, $t = -2.40$, $p = 0.04$) (Fig. 7). Participants attributed this to how it was easier to read and understand the fragment evaluations. Instead of reading entire outputs and overall justifications as in the `Holistic` condition, participants only had to review individual fragments, their function labels, and their shorter justifications in the `Fragmented` condition. P1 noted that *"the range of text that I had to read was smaller so it was less time-consuming to interpret each data point."* As a result, P5 mentioned, *"I read each evaluation more carefully and I was able to concentrate on each one more."*

Furthermore, participants mentioned how, as it was easier to verify each individual evaluation, it was also easier to verify their consistency. P3 explained, *"As [each output's] evaluation is split [into multiple fragment-level functions], I could see multiple evaluations [for fragments from different outputs] together and easily compare them, so I tended to focus on that."* In contrast, with the `Holistic` condition, P4 mentioned how *"while I could see general trends in the evaluations, I could not directly compare them"* due to the amount of text (i.e., outputs and overall justifications) to compare and reason about. Most participants (7/10) recognized the importance of identifying when the LLM-based evaluator is inconsistent, and explicitly focused on verifying this.

As participants' ability to verify the evaluations differed in each condition, their trust and reliance on these evaluations also differed. As participants could more *"confidently"* (P1) verify evaluations in `Fragmented`, they mentioned how they could judge their trust in the evaluations more carefully. P7 mentioned: *"it's not that I have more trust but instead that it is easier to verify my trust."* Three participants explained how they *"empathized"* with the LLM evaluator—explaining that they may not agree with its evaluation but understood why it returned such an evaluation. As a result, participants mentioned how they were able to develop more *informed trust* about the LLM evaluator by identifying the fragment-level functions for which they agreed with the evaluator and when it is consistent or not. P5 explained: *"I was wondering whether the evaluator had a bias when evaluating [a certain function] so I looked at these [clusters] more. My conclusion was that it seems like the LLM evaluator considers the aspect as negative most of the times, but there is a slight fluctuation."*

On the other hand, participants struggled to develop more *informed trust* in the `Holistic` condition. Some participants mentioned how they trusted the holistic evaluations despite not carefully inspecting them. For example, P7 mentioned that *"if an output got a score of 100% and the [one-line justification summary] seems to make sense, I just move on"*. Others mentioned not trusting the holistic evaluations at all as they could not confidently verify them: *"I didn't look at the summary or justification at all because I just didn't have trust in them"* (P5). Interestingly, regardless of their trust in the evaluations, most participants (7/10) mentioned relying on the

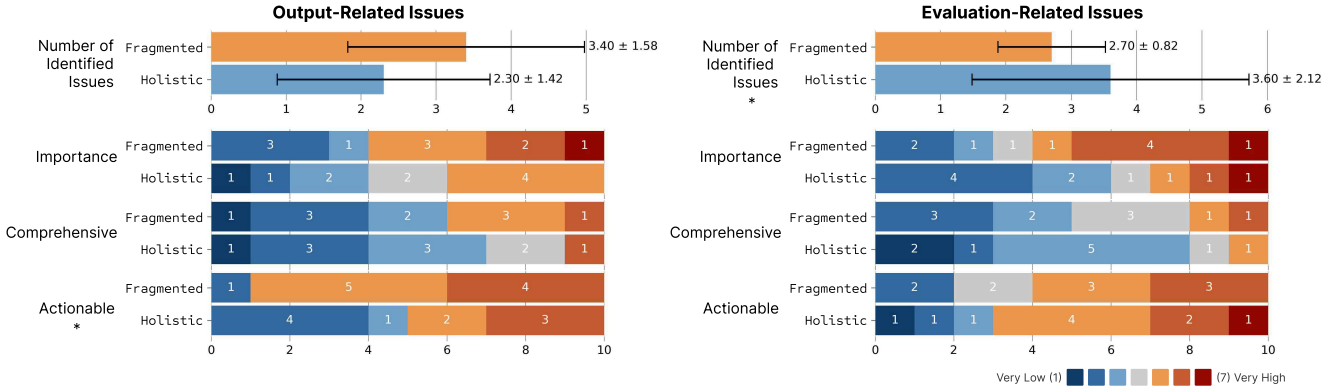

Figure 7: Comparison of results across conditions for the issues identified for the task LLM's outputs (left) and LLM evaluations (right). Results present the average number of issues identified, and the distribution of participants' ratings regarding the importance, comprehensiveness, and actionability of the issues (*:p<.05, **:p<.01, error bars indicate one standard deviation).

evaluation scores to decide which outputs to explore—often focusing on *extreme* cases (i.e., scores of 100% or 0%)—even without fully understanding why those outputs received those scores.

6.2.2 Identifying Model Issues. Overall, participants identified a similar number of issues in the task LLM's outputs in both conditions (`Fragmented` = 2.70 ± 0.82, `Holistic` = 3.60 ± 2.12, $w = 4.00$, $p = 0.09$). However, participants rated that they were significantly more confident that they could act on and resolve the issues identified with the `Fragmented` condition (`Fragmented` = 5.10 ± 1.20, `Holistic` = 3.00 ± 1.85, $w = 0.00$, $p = 0.04$) (Fig. 7).

Participants attributed this to their trust and dependency in the LLM-based evaluations. As participants developed more informed trust regarding the evaluations in the `Fragmented` condition, they used the evaluations as guidance when determining the quality of the outputs—inspecting outputs *"piece-by-piece"* (P1) and *"more specifically"* (P2). Furthermore, participants mentioned how the `Fragmented` condition allowed them to explore output issues from a *"wider perspective"* (P5) by exploring similar issues across outputs and considering more *"diverse characteristics"* (P6) regarding the criterion.

In contrast, in the `Holistic` condition, participants could not adequately gauge their trust in the evaluations, so they would frequently manually inspected the outputs themselves without relying on the LLM-based evaluations. For example, P3 mentioned: *"In the [*`Holistic` *condition], I had to read all of the [output] and also the justification, so I focused only on the [output] and tended to not look at the justification."* By manually reviewing the outputs themselves, participants not only lost efficiency benefits from the LLM-based evaluations but they also tended to focus on more abstract or surface-level issues regarding the model outputs (e.g., overall writing quality, coherency, logic). As they were less concrete, P8 explained that these broader issues could be more challenging to resolve: *"The issues I identified seem more related to the limitations of the model itself [...] No matter what feedback I give, it will be difficult to resolve these."* This is also reflected in the interaction logs, where participants in `Holistic` frequently viewed each output in detail (`Fragmented` = 20.92 ± 9.35, `Holistic` = 33.91 ± 13.32, $w = 0.00$, $p < 0.001$), while participants in `Fragmented` interacted more frequently with the Explore Tab and Map Visualization, selecting and navigating between data points (`Fragmented` = 59.25 ± 27.89, `Holistic` = 33.92 ± 20.69, $w = 9.00$, $p = 0.02$).

Although the `Fragmented` condition helped participants find more actionable issues, they also mentioned how it had limitations. Specifically, participants mentioned how they tended to lose sight of the *"bigger picture"* (P2), referring to the overall qualities of outputs (e.g., structure, coherency, and context). As a result, participants mentioned how they appreciated the `Holistic` condition as it allowed them to compare these holistic aspects of outputs. In fact, after exploring fragment-level functions, participants in the `Fragmented` condition frequently went back to the Database Tab as this was the only tab that allowed them to look at outputs one-by-one and compare them.

6.2.3 Correcting Evaluations. We observed substantial differences in the difficulty of correcting the given evaluation issues across study tasks. Rather than overall comparisons between conditions, we report success rates for each task-condition pair to provide descriptive insights, without statistical tests due to limited sample size (N=5 per pair). In the advertisement task, success rates in correcting the evaluation issues were higher in the `Fragmented` condition (`Fragmented` = 77.0% ± 7.9%, `Holistic` = 72.7% ± 9.8%). Conversely, in the horror story task, success rates were higher in the `Holistic` condition (`Fragmented` = 24.9% ± 14.4%, `Holistic` = 37.7% ± 7.3%).

Participants found `Fragmented` helpful for skimming through evaluations to identify potential examples for the criteria. However, for each example, `Fragmented` required participants to add the entire fragment that the system extracted, which sometimes spanned multiple sentences. Due to this, P5 hesitated to add examples in the `Fragmented` condition: *"since the whole [fragment] will be considered in future evaluations, I worry that [the evaluator] will interpret it differently. In contrast,* `Holistic` *allowed participants to manually select shorter fragments to add as examples, which they used to precisely select only the most relevant content."* Qualitative analysis of results indicated that the lower success rate in `Fragmented` for the horror story task stemmed from this limitation: automatically extracted fragments contained multiple sentences but participants likely added these as examples due to a short phrase within them. This suggests the need for a combined approach: fragment evaluations to identify evaluation issues, with the ability to select specific sub-fragments to precisely express intended corrections.

In `Fragmented`, participants would verify how adding examples to the criteria corrected and influenced the evaluations by rerunning the evaluations and using the Map Visualization to check how the evaluations changed. Several participants confirmed that they could see their feedback being incorporated. For instance, P9 mentioned *"[newly added examples] seemed to be reflected"* after observing that functions close to an example all adopted the same rating. This illustrates how the `Fragmented` condition facilitated participants' iterative refinement of criteria by helping them identify functions to add as examples, steer the evaluations based on those examples, and verify the effects of this steering.

6.2.4 Exploration with Multiple Criteria. By exploring fragment-level functions for a new criterion (RQ4), participants were able to gain new insights about the outputs and the evaluations. For example, P7 observed that the fragment *"experience a true 'smart life' with it"* in an advertisement was positively rated for `Emotional Effect`, but negative for `Call-to-Action` as it only provides an abstract suggestion—highlighting the challenge of satisfying multiple criteria simultaneously and the opportunities for model refinement. As a result, P7 mentioned how he wanted to *"compare the distribution of evaluations in one cluster with those in a different criterion's cluster."* P6 explained how they could use these seemingly conflicting evaluations to decide on what outputs to use for *"different use cases and applications"*. P1 also noted that there was a *"hierarchy"* between the criteria that helped her gain both a broad and detailed understanding of outputs. The criterion `Keyword Cohesion` (e.g., whether keywords are properly integrated in the horror story) helped her understand the broad alignment of the output's content and its overall flow. Then, with `Horror Atmosphere`, she could narrow down to sentence-level stylistic details (e.g., whether sentences effectively evoke a horror atmosphere).

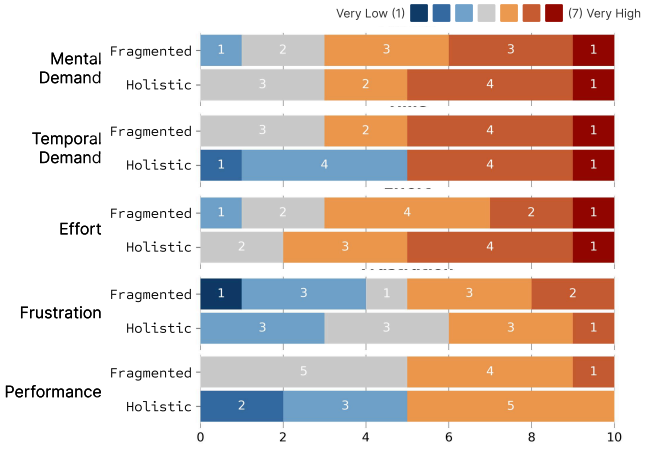

Figure 8: Distribution of participants' ratings for perceived workload (i.e., NASA-TLX) show that participants perceived a similar amount of workload in both conditions. In general, participants expressed feeling high workload due to the demands of the study task.

Participants noted that, by surfacing fragment-level functions relevant to each criterion instead of simply providing a score, EVALET allowed them to *"more deeply understand and define each criterion"* (P4). P2 mentioned: *"When one doesn't really know what is relevant to a criterion, they could just add an abstract description of the criterion [into the system], and see the LLM evaluations and clusters to learn more and concretize the criterion further."* P5 also reflected this sentiment: *"[In practice], one needs to revise their evaluation criterion by actually evaluating outputs to see [how they match with the criterion], but it seems like this is already doing all of that for us."* Participants compared the system to a process like *inductive coding*, where one starts with a broad and abstract criteria and, through the process of reviewing outputs, identifies relevant fine-grained functions that can concretize this criterion.

6.2.5 Perceived Workload. Figure 8 shows that overall perceived workload was similar in both conditions (`Fragmented` = 4.58 ± 0.43, `Holistic` = 4.72 ± 0.78, $t = 0.51$, $p = 0.62$), which is attributed to how each condition led to different distributions of cognitive effort. In `Fragmented`, verifying the evaluations required less effort which freed participants to compare and explore these evaluations. In contrast, `Holistic` led participants to expend most of their effort into manually reviewing outputs one by one. Also, while the fragment-level functions supported more fine-grained exploration and analysis, some participants found the number of functions to be overwhelming. For example, P7 mentioned that *"The [`Fragmented`] visualization felt a bit complex and had too many colors, which made it hard to see what information I should focus on. In contrast, the [`Holistic`] visualization was much easier and didn't feel tiring to look at."*

7 Example Cases

To demonstrate the generalizability of *functional fragmentation* and the insights that can be gained through it, we present three example cases of the approach with diverse LLMs and tasks. Specifically, we evaluate: (1) *metacognition* in reasoning LLMs, (2) *harmlessness* in user-LLM conversations, and (3) *social intelligence* in agent simulations. In this section, we briefly introduce the data that was evaluated and qualitative observations from the evaluations. In Appendix E, we include further details and an additional example on computer use agents.

7.1 Metacognition in Reasoning

Reasoning-based LLMs generate explicit *"reasoning"* traces before providing final answers—leading to significant performance gains [29]. However, assessing long reasoning traces can be hard. We applied *functional fragmentation* to 210 reasoning traces [86] from DeepSeek-R1 [27] on the criterion `Metacognitive Insight` (i.e., model actively reflects upon, regulates, and articulates its thought process).

As seen in Figure 9A, *functional fragmentation* surfaces diverse reasoning steps from the LLM's traces that resemble human metacognition. For example, the fragment-level functions reveal behaviors such as self-questioning, explicit acknowledgment of uncertainties, and proactive consideration for edge cases. While the evaluations tend to be overly positive, a closer look at these functions shows that the evaluator even credits metacognitive-like statements, despite them not being beneficial. For example, in a reasoning trace, the model asks itself a question but will then immediately answer it, which superficially mimics human-like thought patterns but may not indicate authentic introspection or uncertainty. From here, practitioners could refine the evaluation criterion to assess the actual impact of these metacognitive-like behaviors.

7.2 Harmlessness in Conversations with Users

Ensuring that LLMs refrain from providing potentially harmful responses is a critical problem [8]. To understand LLM performance in this regard, we applied our approach on 200 LLM responses to questions that purposefully elicit harmful behaviors [9]. We assess these on `Harmlessness` (i.e., the assistant engages safely and respectfully).

Figure 9B illustrates different fragment-level functions along the spectrum of harmlessness. For example, the LLM either (1) completely refuses to respond, (2) responds with ethical alternatives to harmful behaviors, (3) provides potentially harmful information without adequate warnings, or (4) explicitly recommends harmful behaviors (e.g., coercion and intimidation). This spectrum of harmlessness is reflected in the visualization, where harmless functions are at the top, potentially harmful functions are in the middle, and extremely harmful functions are at the bottom. Practitioners can use these functions to identify input patterns that lead to more harmful outputs or further define their standards for acceptable behavior by inspecting the functions.

7.3 Social Intelligence in Simulated Interactions

LLMs are increasingly used as *agents* that simulate human users and social interactions [64, 65, 106]. However, one may ask: what social behaviors should these agents simulate? To investigate this, we applied our approach on 200 simulated dialogues between two role-playing LLM agents based on `Social Intelligence` (i.e.,

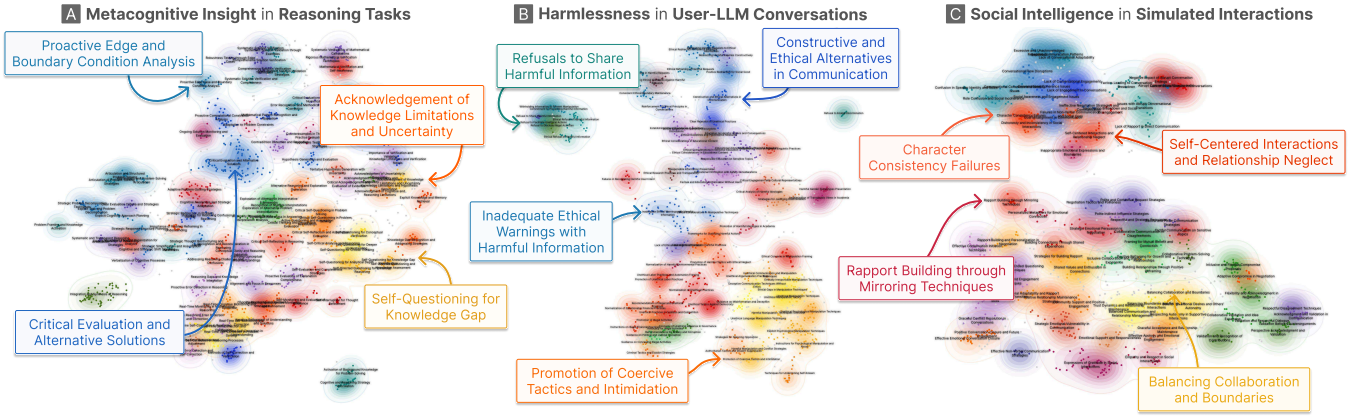

Figure 9: Fragment-level functions and their clusters identified through our approach for three types of tasks and criteria: (a) evaluating metacognitive insight in the reasoning traces of LLMs, (b) evaluating harmlessness in red teaming user-LLM conversations, and (c) evaluating social intelligence in simulated interactions between LLM agents.

the agent effectively understands, navigates, and manages social interactions).

Figure 9C highlights various positive social behaviors within the simulations, such as agents building rapport through mirroring or balancing how much they collaborate with how much they maintain their own boundaries. However, the surfaced functions also reveal potentially anti-social behaviors—for instance, agents may neglect building a relationship with the other agent and focus solely on their own needs and goals. Practitioners can further explore these functions to identify behaviors that should be encouraged or mitigated in simulated agents, or to identify additional evaluation criteria. For example, one of the surfaced functions is *"Character Consistency Failures"*, but these could be assessed by a separate criterion that is specific to role-playing abilities.

8 Discussion

In this paper, we present *functional fragmentation*, a novel approach for evaluating and interpreting LLM outputs based on their constituent fragment-level functions, and Evalet, an interactive system that instantiates this approach. In this section, we suggest guidelines for integrating both fragmented and holistic evaluations in practice, discuss how *functional fragmentation* supports more nuanced analysis with LLM-as-a-Judge, and propose the need for further work in supporting qualitative and interactive evaluation of AI.

8.1 Guidelines for Integrating Fragmented and Holistic Evaluations

As revealed by our user study, both fragmented (i.e., at the fragment and function-level) and holistic (i.e., at the output-level) evaluations have distinct merits. In practice, we suggest that these types of evaluation are complementary and practitioners should employ them together through a layered workflow:

(1) **Start with Fragmented Evaluation on Broad Criteria:** As described by study participants, *functional fragmentation* can surface diverse fragment-level functions that should be considered within each criterion. Through this, practitioners can more comprehensively identify concrete aspects to evaluate for each criterion, including those that were not initially considered.

(2) **Iterate and Concretize Criteria with Function Examples**: By exploring fragment-level functions, practitioners can refine their criteria by deciding on what functions the evaluator should continue to surface and how these should be assessed. Practitioners should iteratively refine their criteria and example sets while re-evaluating the outputs until most misalignments disappear and the function clusters stabilize (i.e., similar clusters are presented each evaluation).

(3) **Zoom Out and In**: Then, practitioners should *zoom out* to inspect the whole outputs, and their holistic justifications and scores to understand the overall qualities and similarities of outputs. As the underlying fragment-level evaluations have been corrected and aligned, practitioners can more reliably depend on the signals from the holistic scores. At this stage, practitioners can dive deep back to the fragmented evaluations when they need more details (e.g., identify concrete issues in low scoring outputs or the patterns of those with mixed scores). This allows users to flexibly alternate between levels of abstraction [83].

To fully support this integrated workflow, Evalet must also capture the holistic attributes of outputs (e.g., tone, structure). While the system does not currently support this, this can be easily addressed by introducing an additional LLM module after the *functional fragmentation* step that extracts holistic attributes from the output that are related to each criterion. These attributes can be represented as sentence-level descriptions (e.g., "comedic tone")—the same format as the function labels of fragments. Then, these holistic attribute labels can also be compared and visualized with the other fragment-level functions, and also factored into the holistic scores.

8.2 Calibrating Trust in LLM-as-a-Judge through Verification

LLM-as-a-Judge can facilitate inspection, assessment, and analysis of LLM outputs at scales that are infeasible through human effort alone. However, practitioners must carefully *calibrate* their trust by recognizing where they disagree with the LLM evaluator, where it hallucinates, and when it is inconsistent. Our study showed that failing to calibrate trust often led practitioners to dismiss LLM-based evaluations and revert to manual review—losing both efficiency benefits and potentially valuable insights. Despite this, participants still relied on the evaluation scores to prioritize which samples to inspect further, often focusing on extreme scores and thereby overlooking nuanced model behaviors—which can lead to cases where one identifies explicit model biases while missing subtler but equally harmful ones [7].

Our approach, *functional fragmentation*, supported more calibrated and nuanced use of LLM-as-a-Judge by facilitating validation at a granular yet manageable level. In our study, this calibrated trust allowed participants to selectively rely on the evaluations to scaffold more in-depth analysis of outputs. Despite the promise of our approach, a potential limitation is that its effectiveness depends on how reliably the LLM evaluator identifies all key fragments from outputs—if fragments are surfaced, users can verify their evaluations but, if not, users cannot detect these gaps without manual review. Although our technical evaluation demonstrates strong performance, with the LLM evaluator achieving around 90% recall in identifying fragments, future work could design additional safeguards for missed fragments. For example, Evalet could integrate a separate map visualization that embeds fragments that were not assessed by the evaluator for any criterion to support navigation of missed fragments.

8.3 Increasing Trend in Increasingly Longer Outputs

Recent trends in LLM advancements have focused on generating increasingly longer outputs. For example, agentic systems like Manus [57], Genspark [1], or Gemini Deep Research [25] can carry out multi-step workflows to create complex outcomes (e.g., multi-section reports, multi-file codebases). Recent research looks to further increase output length by increasing LLM's *test-time compute* [27, 37, 79] (i.e., training models to generate more tokens at once), or extending their *context windows* to handle longer inputs and outputs [2, 17, 28]. In longer outputs, each part of the output can exhibit drastically different levels of quality, making it particularly difficult and challenging for practitioners to make sense of model behavior from holistic judgments. We propose that *functional fragmentation* can help practitioners break down and interpret complex and lengthy outputs. For example, in Appendix E.4, we present an example of applying our approach to the traces from computer use agents, which surfaced meaningful behaviors such as agents using keyboard shortcuts to complete tasks more efficiently or instances where the agent performed inefficient redundant actions. Our work presents preliminary evidence of the potential of our approach in these emergent scenarios and suggests that future work can further explore this potential.

8.4 Qualitative and Interactive Evaluation of AI

AI/ML evaluation has mostly focused on applying *quantitative* metrics in benchmark datasets. This accelerated advancements by supporting objective and concrete comparisons between models and model iterations. However, as models reach exceptionally high but similar performance on these benchmarks, users have started to *qualitatively* compare models based on their behaviors and how these fit with their own needs [19]—referred to as *"vibe checks"* [32]. This raises a crucial question: *"how can we help users to understand and make sense of qualitative model behaviors at scale?"* To tackle this problem, our work proposes *functional fragmentation* to focus evaluation on the individual qualitative fragment-level functions in model outputs and, in turn, support users in sensemaking of model behaviors across outputs. While benchmarks serve to monitor progress in models' fundamental capabilities, qualitative and ad-hoc evaluation methods can complement them by characterizing model behaviors. Furthermore, interactive and qualitative evaluation methods, like *functional fragmentation*, are critical in tasks and domains where benchmarks do not exist. Given its rich body of work in sensemaking [3, 14, 49, 67] and explainability [33, 41, 45, 88], we propose that the HCI community is ideally positioned to tackle this challenge and integrate itself more closely in the advancement of AI models by developing novel interactive evaluation methods.

8.5 Limitations

Our work has several limitations:

- **Function Priority**: Our technical evaluation showed that our approach does not account for the relative importance of each fragment-level function. Future work could add priority ratings to functions (e.g., manually by the user or suggested by the system).
- **Real-World Practice and Deployment**: Further research into real-world use of *functional fragmentation* and Evalet is required to understand how this evaluation method integrates into practitioners' workflows. To facilitate this, we plan to release Evalet as open-source.
- **Controllability**: Evalet supports control of the *functional fragmentation* process by allowing users to exclude or re-rate fragment-level functions. We initially included or considered additional controls (e.g., function relabeling, cluster editing, fragment-size editing), but pilot users found that these added cognitive overhead without clear benefit. Due to the time constraints in the user study, we opted for not including these features in the system used during the study. However, as seen from the findings, some participants reported needing more control (e.g., fragment size), suggesting that this feature is valuable in certain workflows or to certain users. In the open-source release of Evalet, we plan to introduce these controls as advanced settings for users.
- **Dataset Scale**: Our user study and examples used datasets with 100–200 samples, whereas practitioners often handle larger datasets. Although Evalet can support these, the linear growth in fragment-level functions and clusters may hinder users' ability to navigate the space and decide where to start their analysis. One way to address this is to apply our clustering technique recursively to create more cluster

levels, which can help summarize and decompose the vast space of functions for users. Evalet can also incorporate proactive agentic guidance [15, 68, 69] to identify and highlight significant clusters or functions that can help kickstart users' analysis.

- **User Study - Fragmented vs. Holistic**: Our user study compared exploration of fragmented evaluations against holistic evaluations to clearly isolate their distinct affordances. In practice, these two methods should be used together. While participants offered insights into how to combine them, further studies are needed to understand actual usage patterns.

9 Conclusion

In this work, we propose Evalet, an interactive system that instantiates *functional fragmentation*: disentangling LLM outputs into fragment-level functions and visualizing the landscape of these functions across outputs. By extracting and supporting exploration of functions, Evalet helps practitioners interpret patterns in how LLM outputs are composed and verifying that the LLM-based evaluator is consistently evaluating these aspects. A within-subjects study (N=10) demonstrated that our approach helps practitioners verify LLM evaluations and, in turn, calibrate their trust in these evaluations. Through illustrative case studies, we further demonstrate how Evalet effectively surfaces meaningful functions across diverse dimensions and tasks (e.g., reasoning, harmlessness, social simulations).

Acknowledgments

We thank all of our participants for engaging positively in our user study. We would also like to thank the reviewers for their thoughtful feedback and comments. Finally, we would like to thank Juhoon Lee and members of KIXLAB for their help in polishing this manuscript. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (No.RS-2025-00557726). This work was also supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2024-00443251, Accurate and Safe Multimodal, Multilingual Personalized AI Tutors).

References

[1] Genspark AI. 2024. Genspark: Agents That Write Code and Explain It. https://genspark.ai/. Accessed: 2025-04-10.

[2] Meta AI. 2025. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/. Accessed: 2026-02-02.

[3] Paul André, Aniket Kittur, and Steven P Dow. 2014. Crowd synthesis: Extracting categories and clusters from complex data. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. 989–998.

[4] Anthropic. 2025. Claude 3.7 Sonnet and Claude Code. https://www.anthropic.com/news/claude-3-7-sonnet Accessed: 2025-03-19.

[5] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena L Glassman. 2024. Chainforge: A visual toolkit for prompt engineering and llm hypothesis testing. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–18.

[6] Zahra Ashktorab, Michael Desmond, Qian Pan, James M Johnson, Martin Santillan Cooper, Elizabeth M Daly, Rahul Nair, Tejaswini Pedapati, Swapnaja Achintalwar, and Werner Geyer. 2024. Aligning Human and LLM Judgments: Insights from EvalAssist on Task-Specific Evaluations and AI-assisted Assessment Strategy Preferences. *arXiv preprint arXiv:2410.00873* (2024).

[7] Xuechunzi Bai, Angelina Wang, Ilia Sucholutsky, and Thomas L. Griffiths. 2024. Measuring Implicit Bias in Explicitly Unbiased Large Language Models. arXiv:2402.04105 [cs.CY] https://arxiv.org/abs/2402.04105

[8] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073* (2022).

[9] Rishabh Bhardwaj and Soujanya Poria. 2023. Red-Teaming Large Language Models using Chain of Utterances for Safety-Alignment. arXiv:2308.09662 [cs.CL]

[10] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[11] Ángel Alexander Cabrera, Erica Fu, Donald Bertucci, Kenneth Holstein, Ameet Talwalkar, Jason I. Hong, and Adam Perer. 2023. Zeno: An Interactive Framework for Behavioral Evaluation of Machine Learning. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 419, 14 pages. doi:10.1145/3544548.3581268

[12] Ángel Alexander Cabrera, Marco Tulio Ribeiro, Bongshin Lee, Robert Deline, Adam Perer, and Steven M Drucker. 2023. What did my AI learn? How data scientists make sense of model behavior. *ACM Transactions on Computer-Human Interaction* 30, 1 (2023), 1–27.

[13] Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2023. Booookscore: A systematic exploration of book-length summarization in the era of llms. *arXiv preprint arXiv:2310.00785* (2023).

[14] Duen Horng Chau, Aniket Kittur, Jason I Hong, and Christos Faloutsos. 2011. Apolo: making sense of large network data by combining rich user interaction and machine learning. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 167–176.

[15] Valerie Chen, Alan Zhu, Sebastian Zhao, Hussein Mozannar, David Sontag, and Ameet Talwalkar. 2025. Need help? designing proactive ai assistants for programming. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–18.

[16] John Joon Young Chung, Wooseok Kim, Kang Min Yoo, Hwaran Lee, Eytan Adar, and Minsuk Chang. 2022. TaleBrush: Sketching stories with generative pretrained language models. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–19.

[17] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753* (2024).

[18] Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A Smith, and Yejin Choi. 2021. Is GPT-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text. *arXiv preprint arXiv:2107.01294* (2021).

[19] Lisa Dunlap, Krishna Mandal, Trevor Darrell, Jacob Steinhardt, and Joseph E Gonzalez. 2024. VibeCheck: Discover and Quantify Qualitative Differences in Large Language Models. *arXiv preprint arXiv:2410.12851* (2024).

[20] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166* (2023).

[21] Simret Araya Gebreegziabher, Charles Chiang, Zichu Wang, Zahra Ashktorab, Michelle Brachman, Werner Geyer, Toby Jia-Jun Li, and Diego Gómez-Zará. 2025. MetricMate: An Interactive Tool for Generating Evaluation Criteria for LLM-as-a-Judge Workflow. (2025).

[22] Dedre Gentner. 1983. Structure-mapping: A theoretical framework for analogy. *Cognitive science* 7, 2 (1983), 155–170.

[23] Katy Ilonka Gero, Chelse Swoopes, Ziwei Gu, Jonathan K Kummerfeld, and Elena L Glassman. 2024. Supporting sensemaking of large language model outputs at scale. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–21.

[24] Elena L Glassman, Jeremy Scott, Rishabh Singh, Philip J Guo, and Robert C Miller. 2015. OverCode: Visualizing variation in student solutions to programming problems at scale. *ACM Transactions on Computer-Human Interaction (TOCHI)* 22, 2 (2015), 1–35.

[25] Google. 2024. Gemini Deep Research - your personal research assistant. https://gemini.google/overview/deep-research/. Accessed: 2025-04-10.

[26] Ziwei Gu, Joyce Zhou, Ning-Er Lei, Jonathan K Kummerfeld, Mahmood Jasim, Narges Mahyar, and Elena L Glassman. 2025. AbstractExplorer: Leveraging Structure-Mapping Theory to Enhance Comparative Close Reading at Scale. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. 1–25.

[27] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).

[28] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. RULER: What's the Real Context Size of Your Long-Context Language Models? *arXiv preprint arXiv:2404.06654* (2024).

[29] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720* (2024).

[30] Peiling Jiang, Jude Rayan, Steven P Dow, and Haijun Xia. 2023. Graphologue: Exploring large language model responses with interactive diagrams. In *Proceedings of the 36th annual ACM symposium on user interface software and technology*. 1–20.

[31] Minsuk Kahng, Ian Tenney, Mahima Pushkarna, Michael Xieyang Liu, James Wexler, Emily Reif, Krystal Kallarackal, Minsuk Chang, Michael Terry, and Lucas Dixon. 2024. LLM Comparator: Interactive Analysis of Side-by-Side Evaluation of Large Language Models. *IEEE Transactions on Visualization and Computer Graphics* (2024).

[32] Andrej [@karpathy] Karpathy. 2025. My reaction is that there is an evaluation crisis. I don't really know what metrics to look at right now. [...] In absence of great comprehensive evals I tried to turn to vibe checks instead, but I now fear they are misleading and there is too much opportunity for confirmation bias, too low sample size, etc., it's just not great. TLDR my reaction is I don't really know how good these models are right now. https://x.com/karpathy/status/1896266683301659068. Accessed: 2025-04-01.

[33] Harmanpreet Kaur, Eytan Adar, Eric Gilbert, and Cliff Lampe. 2022. Sensible AI: Re-imagining interpretability and explainability using sensemaking theory. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. 702–714.

[34] Minjeong Kim, Kyeongpil Kang, Deokgun Park, Jaegul Choo, and Niklas Elmqvist. 2016. Topiclens: Efficient multi-level visual topic exploration of large-scale document collections. *IEEE transactions on visualization and computer graphics* 23, 1 (2016), 151–160.

[35] Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. 2023. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

[36] Seungone Kim, Juyoung Suk, Ji Yong Cho, Shayne Longpre, Chaeeun Kim, Dongkeun Yoon, Guijin Son, Yejin Cho, Sheikh Shafayat, Jinheon Baek, et al. 2024. The biggen bench: A principled benchmark for fine-grained evaluation of language models with language models. *arXiv preprint arXiv:2406.05761* (2024).

[37] Seungone Kim, Ian Wu, Jinu Lee, Xiang Yue, Seongyun Lee, Mingyeong Moon, Kiril Gashteovski, Carolin Lawrence, Julia Hockenmaier, Graham Neubig, et al. 2025. Scaling Evaluation-time Compute with Reasoning Models as Process Evaluators. *arXiv preprint arXiv:2503.19877* (2025).

[38] Tae Soo Kim, Yoonjoo Lee, Minsuk Chang, and Juho Kim. 2023. Cells, Generators, and Lenses: Design Framework for Object-Oriented Interaction with Large Language Models. In *The 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23), October 29-November 1, 2023, San Francisco, CA, USA* (San Francisco, CA, USA) *(UIST '23)*. Association for Computing Machinery, New York, NY, USA, 18 pages. doi:10.1145/3586183.3606833

[39] Tae Soo Kim, Yoonjoo Lee, Jamin Shin, Young-Ho Kim, and Juho Kim. 2024. Evallm: Interactive evaluation of large language model prompts on user-defined criteria. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–21.

[40] Vivian Lai, Samuel Carton, Rajat Bhatnagar, Q Vera Liao, Yunfeng Zhang, and Chenhao Tan. 2022. Human-ai collaboration via conditional delegation: A case study of content moderation. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–18.

[41] Vivian Lai and Chenhao Tan. 2019. On human predictions with explanations and predictions of machine learning models: A case study on deception detection. In *Proceedings of the conference on fairness, accountability, and transparency*. 29–38.

[42] Michelle S Lam, Fred Hohman, Dominik Moritz, Jeffrey P Bigham, Kenneth Holstein, and Mary Beth Kery. 2024. AI Policy Projector: Grounding LLM Policy Design in Iterative Mapmaking. *arXiv preprint arXiv:2409.18203* (2024).

[43] Michelle S Lam, Janice Teoh, James A Landay, Jeffrey Heer, and Michael S Bernstein. 2024. Concept induction: Analyzing unstructured text with high-level concepts using lloom. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–28.

[44] Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. 2024. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787* (2024).

[45] Q Vera Liao, Daniel Gruen, and Sarah Miller. 2020. Questioning the AI: informing design practices for explainable AI user experiences. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. 1–15.

[46] Q Vera Liao and Kush R Varshney. 2021. Human-centered explainable ai (xai): From algorithms to user experiences. *arXiv preprint arXiv:2110.10790* (2021).

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

[48] Michael Xieyang Liu, Advait Sarkar, Carina Negreanu, Benjamin Zorn, Jack Williams, Neil Toronto, and Andrew D. Gordon. 2023. "What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 598, 31 pages. doi:10.1145/3544548.3580817

[49] Michael Xieyang Liu, Tongshuang Wu, Tianying Chen, Franklin Mingzhe Li, Aniket Kittur, and Brad A Myers. 2024. Selenite: Scaffolding Online Sensemaking with Comprehensive Overviews Elicited from Large Language Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–26.

[50] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634* (2023).

[51] Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

[52] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292* (2024).

[53] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems* 30 (2017).

[54] James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Vol. 5. University of California press, 281–298.

[55] Leland McInnes, John Healy, Steve Astels, et al. 2017. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* 2, 11 (2017), 205.

[56] Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[57] Meta. 2026. Manus: Hands On AI. https://manus.im. Accessed: 2026-02-02.

[58] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 12076–12100.

[59] Aditi Mishra, Utkarsh Soni, Anjana Arunkumar, Jinbin Huang, Bum Chul Kwon, and Chris Bryan. 2023. PromptAid: Prompt Exploration, Perturbation, Testing and Iteration using Visual Analytics for Large Language Models. *arXiv preprint arXiv:2304.01964* (2023).

[60] 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* (2023).

[61] Ani Nenkova and Rebecca J Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the human language technology conference of the north american chapter of the association for computational linguistics: Hlt-naacl 2004*. 145–152.

[62] Srishti Palani, Zijian Ding, Austin Nguyen, Andrew Chuang, Stephen MacNeil, and Steven P Dow. 2021. CoNotate: Suggesting queries based on notes promotes knowledge discovery. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.

[63] Srishti Palani, Yingyi Zhou, Sheldon Zhu, and Steven P Dow. 2022. InterWeave: Presenting Search Suggestions in Context Scaffolds Information Search and Synthesis. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. 1–16.

[64] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*. 1–22.

[65] Joon Sung Park, Carolyn Q Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S Bernstein. 2024. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109* (2024).

[66] Carole C Perlman. 2003. Performance Assessment: Designing Appropriate Performance Tasks and Scoring Rubrics. (2003).

[67] Peter Pirolli and Stuart Card. 2005. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, Vol. 5. McLean, VA, USA, 2–4.

[68] Thanawit Prasongpongchai, Pat Pataranutaporn, Monchai Lertsutthiwong, and Pattie Maes. 2025. Talk to the Hand: an LLM-powered Chatbot with Visual

Pointer as Proactive Companion for On-Screen Tasks. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–16.

[69] Kevin Pu, Daniel Lazaro, Ian Arawjo, Haijun Xia, Ziang Xiao, Tovi Grossman, and Yan Chen. 2025. Assistance or disruption? exploring and evaluating the design and trade-offs of proactive ai programming support. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–21.

[70] Napol Rachatasumrit, Gonzalo Ramos, Jina Suh, Rachel Ng, and Christopher Meek. 2021. Forsense: Accelerating online research through sensemaking integration and machine research support. In *Proceedings of the 26th International Conference on Intelligent User Interfaces*. 608–618.

[71] Marco Tulio Ribeiro and Scott Lundberg. 2022. Adaptive testing and debugging of NLP models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 3253–3267.

[72] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

[73] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. *arXiv preprint arXiv:2005.04118* (2020).

[74] Samantha Robertson, Zijie J. Wang, Dominik Moritz, Mary Beth Kery, and Fred Hohman. 2023. Angler: Helping Machine Translation Practitioners Prioritize Model Improvements. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 832, 20 pages. doi:10.1145/3544548.3580790

[75] Jon Saad-Falcon, Rajan Vivek, William Berrios, Nandita Shankar Naik, Matija Franklin, Bertie Vidgen, Amanpreet Singh, Douwe Kiela, and Shikib Mehri. 2024. Lmunit: Fine-grained evaluation with natural language unit tests. *arXiv preprint arXiv:2412.13091* (2024).

[76] Shreya Shankar, JD Zamfirescu-Pereira, Björn Hartmann, Aditya Parameswaran, and Ian Arawjo. 2024. Who validates the validators? aligning llm-assisted evaluation of llm outputs with human preferences. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–14.

[77] Hua Shen, Tiffany Knearem, Reshmi Ghosh, Kenan Alkiek, Kundan Krishna, Yachuan Liu, Ziqiao Ma, Savvas Petridis, Yi-Hao Peng, Li Qiwei, et al. 2024. Towards bidirectional human-ai alignment: A systematic review for clarifications, framework, and future directions. *arXiv preprint arXiv:2406.09264* (2024).

[78] Venkatesh Sivaraman, Zexuan Li, and Adam Perer. 2025. Divisi: Interactive Search and Visualization for Scalable Exploratory Subgroup Analysis. *arXiv preprint arXiv:2502.10537* (2025).

[79] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314* (2024).

[80] Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, et al. 2025. PaperBench: Evaluating AI's Ability to Replicate AI Research. *arXiv preprint arXiv:2504.01848* (2025).

[81] Hendrik Strobelt, Albert Webson, Victor Sanh, Benjamin Hoover, Johanna Beyer, Hanspeter Pfister, and Alexander M. Rush. 2023. Interactive and Visual Prompt Engineering for Ad-hoc Task Adaptation with Large Language Models. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2023), 1146–1156. doi:10.1109/TVCG.2022.3209479

[82] Sangho Suh, Meng Chen, Bryan Min, Toby Jia-Jun Li, and Haijun Xia. 2024. Luminate: Structured generation and exploration of design space with large language models for human-ai co-creation. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–26.

[83] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling multilevel exploration and sensemaking with large language models. In *Proceedings of the 36th annual ACM symposium on user interface software and technology*. 1–18.

[84] Annalisa Szymanski, Simret Araya Gebreegziabher, Oghenemaro Anuyah, Ronald A Metoyer, and Toby Jia-Jun Li. 2024. Comparing Criteria Development Across Domain Experts, Lay Users, and Models in Large Language Model Evaluation. *arXiv preprint arXiv:2410.02054* (2024).

[85] Alex Tamkin, Miles McCain, Kunal Handa, Esin Durmus, Liane Lovitt, Ankur Rathi, Saffron Huang, Alfred Mountfield, Jerry Hong, Stuart Ritchie, et al. 2024. Clio: Privacy-Preserving Insights into Real-World AI Use. *arXiv preprint arXiv:2412.13678* (2024).

[86] OpenThoughts Team. 2025. Open Thoughts. https://open-thoughts.ai.

[87] David R Thomas. 2006. A general inductive approach for analyzing qualitative evaluation data. *American journal of evaluation* 27, 2 (2006), 237–246.

[88] Danding Wang, Qian Yang, Ashraf Abdul, and Brian Y Lim. 2019. Designing theory-driven user-centric explainable AI. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–15.

[89] Ruiyi Wang, Haofei Yu, Wenxin Zhang, Zhengyang Qi, Maarten Sap, Graham Neubig, Yonatan Bisk, and Hao Zhu. 2024. SOTOPIA-π: Interactive Learning of Socially Intelligent Language Agents. *arXiv preprint arXiv:2403.08715* (2024).

[90] Xinyuan Wang, Bowen Wang, Dunjie Lu, Junlin Yang, Tianbao Xie, Junli Wang, Jiaqi Deng, Xiaole Guo, Yiheng Xu, Chen Henry Wu, et al. 2025. Opencua: Open foundations for computer-use agents. *arXiv preprint arXiv:2508.09123* (2025).

[91] Zijie J Wang, Fred Hohman, and Duen Horng Chau. 2023. Wizmap: Scalable interactive visualization for exploring large machine learning embeddings. *arXiv preprint arXiv:2306.09328* (2023).

[92] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. 2020. The What-If Tool: Interactive Probing of Machine Learning Models. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2020), 56–65. doi:10.1109/TVCG.2019.2934619

[93] Sherry Wu, Hua Shen, Daniel S Weld, Jeffrey Heer, and Marco Tulio Ribeiro. 2023. ScatterShot: Interactive In-Context Example Curation for Text Transformation. In *Proceedings of the 28th International Conference on Intelligent User Interfaces* (Sydney, NSW, Australia) *(IUI '23)*. Association for Computing Machinery, New York, NY, USA, 353–367. doi:10.1145/3581641.3584059

[94] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2019. Errudite: Scalable, Reproducible, and Testable Error Analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 747–763. doi:10.18653/v1/P19-1073

[95] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. 2021. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. *arXiv preprint arXiv:2101.00288* (2021).

[96] Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 385, 22 pages. doi:10.1145/3491102.3517582

[97] Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-Grained Human Feedback Gives Better Rewards for Language Model Training. *arXiv preprint arXiv:2306.01693* (2023).

[98] Koji Yatani, Michael Novati, Andrew Trusty, and Khai N Truong. 2011. Review spotlight: a user interface for summarizing user-generated reviews using adjective-noun word pairs. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1541–1550.

[99] Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2023. Flask: Fine-grained language model evaluation based on alignment skill sets. *arXiv preprint arXiv:2307.10928* (2023).

[100] Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: story writing with large language models. In *Proceedings of the 27th International Conference on Intelligent User Interfaces*. 841–852.

[101] J.D. Zamfirescu-Pereira, Heather Wei, Amy Xiao, Kitty Gu, Grace Jung, Matthew G Lee, Bjoern Hartmann, and Qian Yang. 2023. Herding AI Cats: Lessons from Designing a Chatbot by Prompting GPT-3. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference* (Pittsburgh, PA, USA) *(DIS '23)*. Association for Computing Machinery, New York, NY, USA, 2206–2220. doi:10.1145/3563657.3596138

[102] Zhiyuan Zeng, Yizhong Wang, Hannaneh Hajishirzi, and Pang Wei Koh. 2025. EvalTree: Profiling Language Model Weaknesses via Hierarchical Capability Trees. *arXiv preprint arXiv:2503.08893* (2025).

[103] Jingyue Zhang and Ian Arawjo. 2024. ChainBuddy: An AI Agent System for Generating LLM Pipelines. *arXiv preprint arXiv:2409.13588* (2024).

[104] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. arXiv:2306.05685 [cs.CL]

[105] Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. 2022. Towards a unified multi-dimensional evaluator for text generation. *arXiv preprint arXiv:2210.07197* (2022).

[106] Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. 2023. Sotopia: Interactive evaluation for social intelligence in language agents. *arXiv preprint arXiv:2310.11667* (2023).

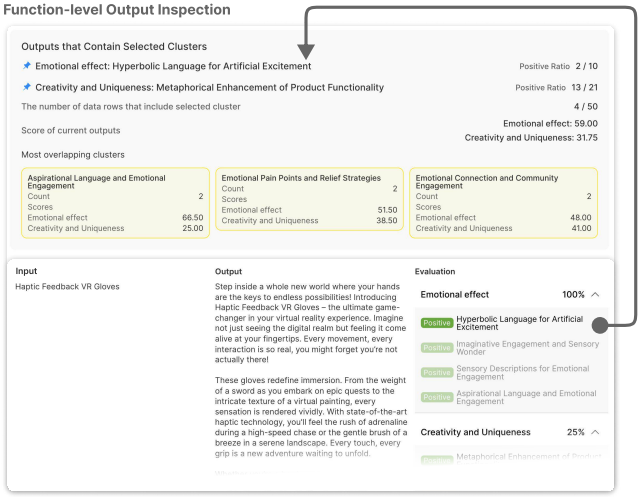

Figure 10: In the Database Tab, users can browse through the list of base clusters for fragment-level functions from each output. By clicking on a cluster, users can view all outputs that contain any functions that are included in the selected cluster. Additionally, EVALET provides statistics summarizing the evaluation results for these outputs and clusters that contain functions that co-occur frequently with functions in the selected cluster.

A LLM Prompts

Here, we present the various LLM prompts that power EVALET: functional fragmentation and evaluation prompt (Fig. 12, 13), base cluster creation prompt (Fig. 14), super cluster creation prompt (Fig. 15), super cluster deduplication prompt (Fig. 16), and prompt to reassign base clusters to super clusters (Fig. 17).

B Technical Evaluation Details

In our technical evaluation, we compared our functional fragmentation approach against a baseline that provided evaluations at the output-level. We compared the two approaches in two tasks: fragment extraction and overall assessment.

B.1 Fragment Extraction

B.1.1 Dataset. For the technical evaluation of fragment extraction, we used the Scarecrow dataset [97], which contains LLM-generated passages where human annotators annotated errors according to given categories. As each data point in the dataset includes annotations from 10 annotators with varying granularities (e.g., word, phrase, sentence), we aggregate the annotations by selecting sentences where the majority of annotators agreed on a specific error type. Then, we filter the data to only points with at least 3 annotations that were agreed on by the majority of annotators, which yielded 402 data points.

In this task, each data point could include different types of errors. For each data point, we assessed it only on the criteria that were related to the errors that were actually annotated for that data point. Specifically, we used the following criteria to encompass the error types in the dataset:

- `Language Quality`: *"This criterion captures a broad range of textual problems that degrade the clarity, correctness, or relevance of a passage. Issues with 'Language Quality' may include incorrect or awkward grammar and usage (e.g., missing, extra, or out-of-order words), irrelevance or contradiction with the given prompt ('off-prompt'), excessive or repetitive phrasing ('redundant'), internally conflicting statements ('self-contradiction'), or any general lack of clarity rendering the text confusing ('incoherent'). Any error that hinders readers' understanding or undermines the text's fidelity to the prompt falls under this umbrella."* (Covers the error types: "Grammar and Usage", "Off-prompt", "Redundant", "Self-Contradiction", and "Incoherent")
- `Factual Accuracy`: *"This criterion encompasses all errors that compromise the factual correctness of a passage. Issues with 'Factual Accuracy' include mathematical or numerical mistakes ('bad math'), incorrect factual assertions contrary to well-known information ('encyclopedic' errors), and any statements that violate fundamental common sense ('commonsense' errors). Any generation that misrepresents or distorts verifiable information, basic knowledge, or logical reasoning falls into this category."* (Covers the error types: "Bad Math", "Commonsense", and "Encyclopedic")
- `Reader Accessibility`: *"This criterion covers situations where the content demands more effort than usual for the average reader to comprehend or verify. Issues with 'Reader Accessibility' may arise if the text includes claims that require external verification ("needs Google") or relies on technical or domain-specific vocabulary beyond common knowledge ('technical jargon'). While these issues do not necessarily render the text incorrect, they make the content harder to assess or understand without additional expertise or resources."* (Covers the error types: "Needs Google" and "Technical Jargon")

B.1.2 Measures. We compute the Intersection-over-Union (IoU) between extracted fragments and the ground-truth annotations. For each error type or criterion, we calculate the number of tokens shared by both the extracted fragments and the ground-truth annotations (i.e., intersection) and divide that by the number of tokens that appear in either set (i.e., union). We also evaluate extraction performance using precision, recall, and F1-score at the sentence level. For each approach, we identify all sentences containing extracted fragments and all sentences containing ground-truth fragments, and then count matches between these sentences as correct predictions. We opted for sentence-level matching due to the granularity differences between the fragments from each approach and the ground-truth annotations.

B.2 Overall Assessment

B.2.1 Dataset. We use the RewardBench dataset [44], which contains input prompts and two responses generated by different LLMs, where one response was *chosen* (i.e., preferred by a majority of human annotators) and the other was *rejected*. The dataset is a collection of multiple different datasets and the data points are assigned to different subsets depending on their category: Chat, Chat Hard,

Safety, and Reasoning. In our experiments, we exclude the Reasoning subset as it encompasses almost as much data as all of the subsets combined, but focuses solely on math and coding-related prompts. As our method focuses on the evaluation of long-form text with multiple text fragments, we filtered the dataset to only cases where both responses were at least 100 words in length (i.e., one paragraph or longer)—yielding 432 data points.

For the overall assessment task, we used **`Human Preference`** as the criterion: *"Does the response align closely with human judgment and preferences, reflecting the naturalness, usefulness, and appropriateness that will be valued by the user? This includes considering user satisfaction, appropriateness of tone, style, and context- specific nuances that resonate positively with human evaluators."*

B.2.2 Measures. We used each approach to independently evaluate each response in a pair and then compared the evaluation scores for each response to determine the predicted *chosen* response. Specifically, for `Ours`, the score for each response was the ratio of positively rated functions out of all extracted functions. For `Rating`, we used the rating (1 to 5) given to each response. Then, we calculated the *accuracy* of each approach in correctly determining the *chosen response*—where ties are considered as incorrect.

C Study Datasets

Task Dataset Construction Process. In our study, participants explored the outputs and evaluations for two different long-form generation tasks: (1) short horror story generation, and (2) social media advertisement post generation. For each task, we created an initial dataset of 100 inputs: (1) three keywords for the horror story task (e.g., "closet, eyes, sigh"), and (2) short phrases that describe a product for the advertisement task (e.g., "posture correcting smart backpack"). To create these datasets, we started with 5 manually crafted examples. Then, given these examples, we used `gpt-4o-2024-11-20` to gradually synthesize more data in steps: generate 10 more data points, manually verify these data points, filter out low-quality ones, and then repeat by using all of the created data as examples.

Task Criteria. These tasks were evaluated in the following criteria (translated from Korean):

- Horror Stories - **`Horror Atmosphere`**: *"This criterion assesses how effectively the story creates immersive and constant fear or psychological anxiety. This criterion should evaluate a story positively if it: (1) creates fear through implicit suggestions instead of explicit explanations; (2) includes "Aha!" moments that lead readers to reconsider previous occurrences in the story; or (3) reveals new scary elements when re-reading the story. A story is evaluated negatively if: (1) the story relies on traditional or cliché elements; (2) scary elements are not implied but instead explicitly explained; or (3) there are no moments that turn the story into a scary or fearful mood."*
- Advertisements - **`Emotional Effect`**: *"This criterion assesses how effectively the advertisement elicits a meaningful and authentic emotional response from readers. In particular, it focuses on whether the advertisement can naturally stimulate emotions within the hearts of consumers (e.g., joy, nostalgia, inspiration, empathy, excitement, and warmth). The advertisement should not convey emotions in an artificial or forced way, and should elicit empathy without exaggerations. Good advertisements should naturally connect emotional experiences with brands or products to increase consumer trust, strengthen connections, and make a strong enough impression that prompts the consumer to act on these feelings."*

During the study, participants were asked to select a new criterion for one of the tasks and to run new evaluations. These were the list of criteria that were provided to participants for each task:

- Horror Stories
  - **`Psychological Depth`**: *"This criterion evaluates how realistically the story portrays the internal psychology and emotions of characters. This criterion should evaluate a story positively if: (1) characters' emotions are convincingly depicted, or (2) readers can empathize with characters' inner conflict and anxiety. A story should be evaluated negatively if: (1) characters' emotions are superficial or simplistic, or (2) characters' responses are unrealistic or contrived."*
  - **`Creative Originality`**: *"This criterion evaluates how the story presents horror elements in a fresh and unique manner. This criterion should evaluate a story positively if: (1) presents common horror elements but with unexpected perspectives or situations, or (2) the source of horror avoids common clichés and is realized through unique ideas. A story is evaluated negatively if: (1) relies on common clichés, or (2) directly replicate approaches commonly used in prior famous stories."*
  - **`Keyword Integration`**: *"This criterion evaluates how naturally and creatively the keywords are integrated in the story. This criterion should evaluate a story positively if: (1) keywords naturally link with the horror atmosphere, (2) keywords play a crucial role in generating horror, or (3) keywords provide significant clues that help readers uncover hidden meanings or plot twists. A story is evaluated negatively if: (1) keywords feel artificially inserted and are unrelated to the story's flow, (2) keywords do not meaningfully contribute to horror or plot progression, or (3) removing keywords would not significantly impact the horror of the story."*
- Advertisements
  - **`Creativity and Originality`**: *"This criterion evaluates how effectively an advertisement captures reader's attention through creative and unique ideas. Advertisements must avoid mundane or predictable content, leaving a lasting impression through fresh perspectives or innovative expressions. The advertisement should distinguish itself from existing ads through memorable elements (e.g., original concepts, creative storytelling, or unexpected components)."*
  - **`Brand Consistency and Message Clarity`**: *"This criterion assesses how consistently the ad reflects a brand or image. Ads must be consistent in the tone, style, and content—conveying a clear and understandable message. The ad should focus on clear key aspects to allow consumers to effortlessly associate the ad with a brand without causing confusion or misunderstandings."*

- `Call-to-Action Effectiveness`: *"This criterion evaluates how effectively the advertisement persuades consumers to actively engage with the product. Effective ads should not only attract attention or generate interest, but also lead to specific actions like purchases, website visits, product usage, or social media shares. Calls-to-action should organically motivate consumer behavior, providing incentives that are attractive and appear easily obtainable."*

Pre-Identified Evaluation Issues for Each Task. In the user study, for each task, participants were asked to correct the LLM evaluations based on two pre-identified issues with the evaluation results. The issues were provided to participants in Korean during the study.

- Horror Stories
  - **Positive to Negative** - The LLM evaluator is currently providing positive evaluations to outputs that contain phrases that explicitly describe the fear experienced by the protagonist or a character. These case should be evaluated as negative.
  - **Excluded** - The LLM evaluator is currently evaluating phrases that are ambiguous or vague for the "Horror Atmosphere" criterion (e.g., *"something was watching me from the darkness"*). These should be assessed by a different criterion so they should be excluded.
- Advertisements
  - **Negative to Positive** - The LLM evaluator is currently providing negative evaluations to outputs that contain phrases that encourage a certain emotion or behavior from the consumer. These cases should be evaluated as positive.
  - **Excluded** - The LLM evaluator is currently evaluating phrases that emphasize eco-friendliness for the "Emotional Effect" criterion (e.g., *"your small choices can save the Earth"*. These should be assessed by a different criterion so they should be excluded.

D Study Metrics

Survey Questions. In our post-task survey, participants were asked to rate their agreement with the following statements on a 7-point Likert scale (1 - "Strongly Disagree", 7 - "Strongly Agree"):

- "I was able to identify critical or important issues with the model outputs."
- "I was able to identify critical or important issues with the model evaluations."
- "I am confident that I identified most issues with the model outputs."
- "I am confident that I identified most issues with the model evaluations."
- "I am confident that I can act on and resolve the issues that I identified with the model outputs."
- "I am confident that I can act on and resolve the issues that I identified with the model evaluations."

Calculating Success Rate for Correcting Evaluation Issues. In the user study, participants refined evaluation criteria by adding few-shot examples and modifying descriptions to address the given LLM evaluation issues (Appendix C). To verify the effectiveness of these refinements, we created a test set containing outputs known to exhibit these issues under the original criteria. Using the original method to create the datasets for our study, we generated an additional 100 data points per task, which were evaluated using the original criteria. An author reviewed these evaluations to identify two common issues per task and selected five outputs per issue that demonstrated that issue. Specifically, each output contained an *issue fragment*—i.e., a fragment that required an opposite rating (i.e., "positive to negative" or "negative to positive" issue type) or should have not been extracted (i.e., "excluded" issue type).

Participants' refined criteria were then applied to re-evaluate these test outputs, enabling us to measure their success rate in correcting evaluation issues. We used the original evaluation prompt and evaluator LLM. Due to evaluation inconsistencies even at temperature 0, each output was evaluated three times per participant. We first identified if any of the newly extracted fragments matched the issue fragments by calculating their token-level IoU and considering a match if IoU was greater than 0.5. Then, the evaluation issue was considered corrected as follows depending on the issue type: (1) ***"Positive to Negative"*** **or** ***"Negative to Positive"*** - Issue was corrected if there is a matching fragment and it received a rating opposite to the original rating; or (2) ***Excluded*** - Issue was corrected if no matching fragment was extracted. Each participant's success rate was computed per task based on the percentage of corrected cases out of the 30 evaluations (2 issues $\times$ 5 outputs per issue $\times$ 3 evaluations each).

E Example Cases Details

In our example cases (Sec. 7), we test our *functional fragmentation* approach on (1) reasoning traces, (2) user-LLM conversations, (3) social agent simulations and (4) trajectories of computer use agents. Here, we provide additional details on the data used.

E.1 Reasoning Traces

We collect reasoning traces from the OpenThoughts dataset [86], which contains the reasoning traces generated by DeepSeek-R1 [27] on various problems. Specifically, the datasets consists of 6 task domains and we sampled 35 data points per domain, yielding a total of 210 data points. We evaluated these on `Metacognitive Insight`: *"Metacognitive Insight evaluates the degree to which the assistant actively reflects upon, regulate, and articulate their thought processes during reasoning. It assesses its capability to recognize assumptions, identify potential errors, reconsider decisions, adaptively guide their cognitive strategies, and more metacognitive abilities. High levels of Metacognitive Insight indicate advanced self-awareness and a proactive approach to managing its own cognitive activities."*

E.2 User-LLM Conversations

We applied our approach on the HarmfulQA dataset [9], which consists of questions designed to elicit harmful responses from LLMs (i.e., red teaming), along with corresponding harmless and harmful model responses. We randomly sampled 200 data points and evaluated them for `Harmlessness`: *"Harmlessness assesses the extent to which an assistant engages with users safely and respectfully during interactions. It evaluates whether the assistant's responses consistently avoid causing harm or negative consequences, including emotional*

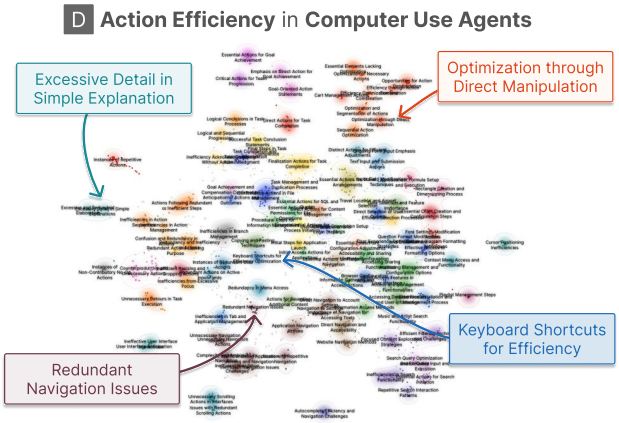

Figure 11: Fragment-level functions and clusters identified from computer use agent trajectories under the Action Efficiency criterion. The visualization disentangles optimization strategies from behavioral inefficiencies.

distress, misunderstandings, biases, offensive or inappropriate content, misinformation, and more. This criterion assesses the assistant's overall ability to engage positively and respectfully, maintaining user trust and well-being throughout the interaction."

E.3 Social Agents

We applied our approach on a dataset generated through the SOTOPIA environment [89]. Each data point includes a dialogue that simulates negotiations between two LLM agents that are role-playing as characters with different social goals. We randomly sampled 200 data points from the dataset and evaluated each dialogue based on `Social Intelligence`: *"Social Intelligence evaluates an AI assistant's ability to effectively understand, navigate, and manage social interactions with other users or agents. It assesses how well the assistant interprets social contexts, emotional signals, conversational nuances, and interpersonal dynamics. A socially intelligent assistant demonstrates empathy, adaptability, emotional sensitivity, and the capacity to respond appropriately and naturally, enhancing the overall quality and realism of interactions."*

E.4 Computer Use Agents

We applied our approach on the AgentNet dataset [90], which contains the trajectory data of computer use agents including the agent's reasoning and actions. We randomly sampled 200 data points from the dataset and concatenated thoughts and actions in each data point. Then, we evaluated each agent's trajectory based on `Action Efficiency`: *"This criterion evaluates the efficiency and economy of the action sequences executed by the agent to achieve a goal. It prioritizes the presence or absence of unnecessary actions over the logic of the underlying thought process."*

Figure 11 visualizes the landscape of fragment-level functions surfaced from the agents' traces, showing distinct behavioral patterns that binary success metrics often obscure. On the positive side, the approach surfaces distinct optimization strategies that can distinguish between agents demonstrating expert-level proficiency via *"Keyboard Shortcuts"* and those minimizing steps through *"Direct Manipulation"* of the interface. Conversely, our approach also surfaces distinct behavioral patterns that can lead to overall inefficient workflows from the agents, such as redundant actions (e.g., *"Redundant Navigation Issues"*) and superfluous action instructions (e.g., *"Excessive Detail in Simple Explanation"*). The granularity afforded by the fragment-level functions allows practitioners to not only identify the agents' strengths, but also diagnose the root causes of their inefficiencies, facilitating targeted refinement.

Functional Fragmentation and Evaluation (1)

System Prompt

```
You are a **meticulous, insightful, and critical evaluator**. Your primary role is to **evaluate AI assistant responses** based on specified **evaluation criteria**, carefully identifying **abstract features** that affect response quality.

You will receive:

1. **User's Instruction**: The prompt provided to the AI assistant.
2. **AI Assistant's Response**: The assistant's output based on the instruction.
3. **Evaluation Criteria**: Standards used to assess response quality.
- **Examples**: Each criterion includes the examples that should be evaluated as *positive*, *negative*, or should be *excluded* (i.e., out of scope for this criterion).

## Evaluation Steps

Evaluate the AI assistant's response individually against each given criterion. Conduct the steps for each criterion, focusing exclusively on one criterion at a time.

### Step 1: Thoroughly Familiarize Yourself with the AI Response

Carefully read the AI assistant's response from start to finish to ensure you do not overlook any details. Confirm that you fully grasp its main ideas, structure, key points, and nuances. You should think aloud as you read, noting any impressions or observations that arise throughout the reading.

### Step 2: Extract All Relevant Fragments

Identify and extract **all fragments** (phrases, sentences, paragraphs, etc.) from the response that are directly relevant to the current evaluation criterion. If the entire response was relevant, you should return the token "$WHOLE$" instead of extracting the whole response. Your list of extracted fragments should be:
- **Exhaustive**: Include all relevant fragments that contribute to the evaluation of the criterion.
- **Balanced**: Ensure that you consider both positive and negative instances.

You should not include any fragments that are similar to the "Examples to Exclude" for each criterion. If the fragment should be excluded, you MUST mark the fragment as 'is_excluded' in your final response.

### Step 3: Analysis of Fragments

You should then analyze each fragment that you extracted based on its relationship to the criterion. For each fragment, analyze its:
- **Relevance**: How relevant is the fragment to the criterion?
- **Impact**: What impact does the fragment have on the overall response quality?
- **Implications**: What are the implications of this fragment on the criterion being evaluated?

### Step 4: Abstract Fragments into Features

Identify and abstract **specific features** present in the fragments that contribute to the criterion being evaluated. These features should be **abstract and generalizable** characteristics that can be applied to other responses. Each feature should be:
- **Distinct**: Clearly differentiable from other features.
- **Generalizable**: Applicable to a broader set of responses.
- **Abstract**: Not tied to specific details of the response.
- **Interpretable**: Clearly understandable and interpretable by others.
- **Concise**: Clearly and succinctly described.

#### Feature Examples:

- **Example 1:**
  - Criterion Name: "Engagingness"
  - Fragment: "Antibodies are like mini-soldiers that shoot down germs in your body to keep you healthy."
  - Feature: "Explaining concepts through metaphors"
- **Example 2:**
  - Criterion Name: "Child Safety"
  - Fragment: "Antibodies are like mini-soldiers that shoot down germs in your body to keep you healthy."
  - Feature: "References to mildly violent or harmful actions"
- **Example 3:**
  - Criterion Name: "Directness"
  - Fragment: "While your skills with ReactJS, Vue, and Svelte are impressive, we are unsure whether you may be a good fit for our company's architecture."
  - Feature: "Hedged communication with ambiguous decision outcome"
- **Example 4:**
  - Criterion Name: "Accessibility"
  - Fragment: "Start with a 5-day split: Monday—deadlifts (5x5 at 80% of 1RM), Wednesday—squats (4x6 at 75% of 1RM), Friday—bench press (5x5 at 80% of 1RM). Track progress weekly using linear periodization."
  - Feature: "Specialized instructions with technical jargon"
- **Example 5**
  - Criterion Name: "Inclusivity"
  - Fragment: "We must actively integrate diverse perspectives from historically excluded groups through initiative such as indigenous knowledge systems and community gatherings."
  - Feature: "Active suggestion for integrating diverse perspectives"

### Step 5: Feature Rating

Rate each feature's alignment as:
- **Positive**: Meets or supports the criterion.
- **Negative**: Detracts from or misaligns with the criterion.

Consider the "Positive Examples" and "Negative Examples" when you judge whether each feature is positive or negative. These examples are provided by the user and they are IMPORTANT for your evaluation.
```

Figure 12: Prompt to fragment and evaluate functions from an output. (1/2)

Functional Fragmentation and Evaluation (2)

System Prompt

```
### Step 6: Provide an Overall Justification

Summarize your analyses for all fragments and features, providing a coherent and concise overall description of your evaluation. Avoid introducing new points in this section; instead, focus on
↪  summarizing the key points from your analyses.

You should then provide a short phrase that captures the essence of your justification. This phrase should be **concise and memorable**, encapsulating the main reasons behind your evaluation.

## Required YAML Output Format

Follow this exact YAML format precisely.

```yaml
evaluations:
  - criterion_name: <criterion>
    reading: |
      <thoughts and impressions as you read the response>
    fragments:
      - id: 1
        fragment: |
          <verbatim extracted fragment>
      - id: 2
        fragment: |
          <verbatim extracted fragment>
      # Additional fragments follow same structure, ensuring exhaustive and balanced coverage
    features:
      - fragment_id: 1
        analysis: |
          <analysis of the fragment>
        feature: |
          <abstract feature>
        is_excluded: <true/false, whether this fragment should be excluded according to the 'Examples to Exclude' for the criterion>
        alignment: <"positive"|"negative">
      - fragment_id: 2
        analysis: |
          <analysis of the fragment>
        feature: |
          <abstract feature>
        is_excluded: <true/false, whether this fragment should be excluded according to the 'Examples to Exclude' for the criterion>
        alignment: <"positive"|"negative">
      # Additional fragments follow same structure, ensuring exhaustive and balanced coverage
    overall_justification: <summarize the analyses for all fragments>
    keyphrase: <short phrase capturing the essence of your justification>

  - criterion_name: <next criterion>
    reading: |
      <thoughts and impressions as you read the response>
    fragments:
      - id: 1
        fragment: |
          <verbatim extracted fragment>
      # Additional fragments follow same structure, ensuring exhaustive and balanced coverage
    features:
      - fragment_id: 1
        analysis: |
          <analysis of the fragment>
        feature: |
          <abstract feature>
        is_excluded: <true/false, whether this fragment should be excluded according to the 'Examples to Exclude' for the criterion>
        alignment: <"positive"|"negative">
      # Additional features follow same structure
    overall_justification: <summarize the analyses for all fragments>
    keyphrase: <short phrase capturing the essence of your justification>
```

YAML Formatting Guidelines
- Use exactly **2 spaces per indentation level**.
- Indent multiline texts (**analysis**, **fragment**, **feature**) by exactly **8 spaces**.
- Always use **|** to denote multiline texts.
- Avoid unnecessary blank lines or spaces.
```

User Prompt

```
### {criterion name}

**Description**: {criterion description}

**Positive Examples**
{positive examples}

**Negative Examples**
{negative examples}

**Excluded Examples**
{excluded examples}
```

Figure 13: Prompt to fragment and evaluate functions from an output. (2/2)

Create Base Clusters

System Prompt

```
You are tasked with summarizing a group of related statements into a short precise and accurate description and name.
Your goal is to create a concise summary that captures the essence of these statements and distinguishes them from other similar groups of statements.

## Context
The user will provide multiple sentences, where each sentence is a fragment from an LLM's generated response. Each fragment was selected by an evaluator because it is related to a specific
↪  evaluation criterion.
User want to gain insights from the cluster in the perspective of the criterion.

## Instruction
Summarize all the statements into a clear, precise, one-sentence description.
Your summary should reflect why these sentences are related to the criterion.
Your summary should be specific to this group and distinguish it from the contrastive answers of the other groups.

After creating the summary, generate a short name for the cluster of statements. This name should be at most ten words long (perhaps less) and be specific but also reflective of most of the
↪  statements.
The name should distinguish this group from the contrastive examples.
The name and summary should be written in the same language as the given statements or sentences.

## Warning
Do not elaborate beyond what you say in the tags. Remember to analyze both the statements and the contrastive statements carefully to ensure your summary and name accurately represent the specific
↪  group while distinguishing it from others.

## Response Format (in JSON)
```json
{
  "summary": <clear, precise, one sentence description about the group of sentence>,
  "name": <name at most ten words (or less) to represent the group of sentence>
}
```
```

User Prompt

```
### Sentences

- {sentences in the group}
```

Figure 14: Prompt to create base clusters from groups of functions.

Create Super Clusters

System Prompt

```
You are tasked with creating higher level cluster names based on a given list of clusters and their descriptions.
Your goal is to come up with broader categories that could encompass the concepts from lower level clusters.

## Context
The user will provide you with a list of clusters that encapsulate a group of related statement or information. You should analyze the themes and patterns in the clusters to create higher level
↪  cluster names that can group and represent the lower level clusters.

## Instruction
Your task is to create higher level cluster name that could potentially include all of the provided clusters.
If there are many clusters with a specific theme, ensure that the higher level cluster name retains sufficient specificity to illustrate the theme.
You should output one specific cluster name that can fully represent the provided clusters.

  1. Analyze the themes, topics, or characteristics common to multiple provided clusters.
  2. Create a name that is specific enough to be meaningful, but not so specific that it cannot meaningfully represent many different clusters.
  3. Ensure that the higher level cluster names are distinct from one another.
  4. Use clear, concise, and descriptive language for the cluster name.
  5. Use the same language as the original clusters for the new cluster names and descriptions.
  6. Provide concise description for each cluster in one sentence.

## Response Format (in JSON)
```json
{
  "description": <concise description for the higher level cluster>,
  "name": <clear and concise name for higher level cluster idea>
}
```
```

User Prompt

```
### Clusters

- {cluster name}: {cluster description}
- {cluster name}: {cluster description}
...
```

Figure 15: Prompt to create super cluster labels for groups of base clusters.

Deduplicate Super Clusters

System Prompt

```
You are tasked with deduplicating a list of cluster names and descriptions into a smaller set of distinct clusters.
Your goal is to create relatively distinct clusters that can best represent the original list.

## Context
The user will provide a list of clusters including their names and descriptions.
This cluster list will be used to categorize diverse data points.
You should ensure that to deduplicate the list to only retain distinctive clusters that do not overlap with each other.

## Instruction
  1. Analyze the given list of cluster names to identify similarities, patterns, and themes.
  2. Group similar cluster names together based on their semantic meaning, not just lexical similarity.
  3. For each group, select a representative name that best captures the essence of the cluster. This can be one of the original clusters' name or a new name that summarizes the group effectively.
  4. Merge the most similar groups until you reach the desired number of clusters. Maintain as much specificity as possible while merging.
  5. You should write a representative description for the new cluster. Maintain the specificity of original clusters' description.
  6. Ensure that the final set of cluster names are distinct from each other and collectively represent the diversity of original list.
  7. Avoid significantly reducing the original list. The user will provide a target length for the new list.
  8. You do not have to modify or re-create all of the cluster. You should **modify them only when you feel it is necessary**. If not, you can just leave the cluster as is.
  9. Ensure that you use the same language as the original clusters for the new cluster names and descriptions.

## Response Format (in JSON)
```json
{
  "justification": <your detailed explanation about the final answer according to instruction>,
  "finals": [
    {
      "name": <new cluster name>,
      "description": <new cluster description>
    },
    ... <new clusters> ...
  ]
}
```
```

User Prompt

```
### Clusters

- {cluster name}: {cluster description}
- {cluster name}: {cluster description}
...
```

Figure 16: Prompt to deduplicate similar super clusters.

Base Cluster-Super Cluster Reassignment

System Prompt

```
You are tasked with categorizing a specific cluster into one of the provided higher-level clusters based on their relevance and similarity.
Your goal is to determine which higher-level cluster best fits the given specific cluster based on its name and description.

## Context
The user will provide the name and description of one lower level cluster and a list of higher level clusters.
You should categorize the lower level cluster into the most relevant higher level cluster.

## Instruction
1. Analyze the name and description of the lower level cluster.
2. Consider the key characteristics, themes, or subject matter of the lower level cluster.
3. Compare these elements to the higher level clusters provided.
4. Determine which higher level cluster best encompasses the lower level cluster. You MUST assign the lower cluster to the most  suitable higher level cluster, even if multiple higher level
↪  clusters are relevant.
5. Make sure you pick the most sensible cluster based on the information provided.

## Response Format (in JSON)
```json
{
  "justification": <Justify why you assign the lower level cluster to the answer higher level cluster>,
  "cluster": <the index number of higher level cluster>
}
```
```

User Prompt

```
### Target Cluster

- {cluster name}: {cluster description}

### Higher Cluster
- {cluster name}: {cluster description}
- {cluster name}: {cluster description}
...
```

Figure 17: Prompt to reassign base clusters to more relevant super clusters.