Supporting Sensemaking of Large Language Model Outputs at Scale

Katy Ilonka Gero
katy@g.harvard.edu
Harvard University
USA

Chelse Swoopes
cswoopes@g.harvard.edu
Harvard University
USA

Ziwei Gu
ziweigu@g.harvard.edu
Harvard University
USA

Jonathan K. Kummerfeld
jonathan.kummerfeld@sydney.edu.au
University of Sydney
Australia

Elena L. Glassman
glassman@seas.harvard.edu
Harvard University
USA

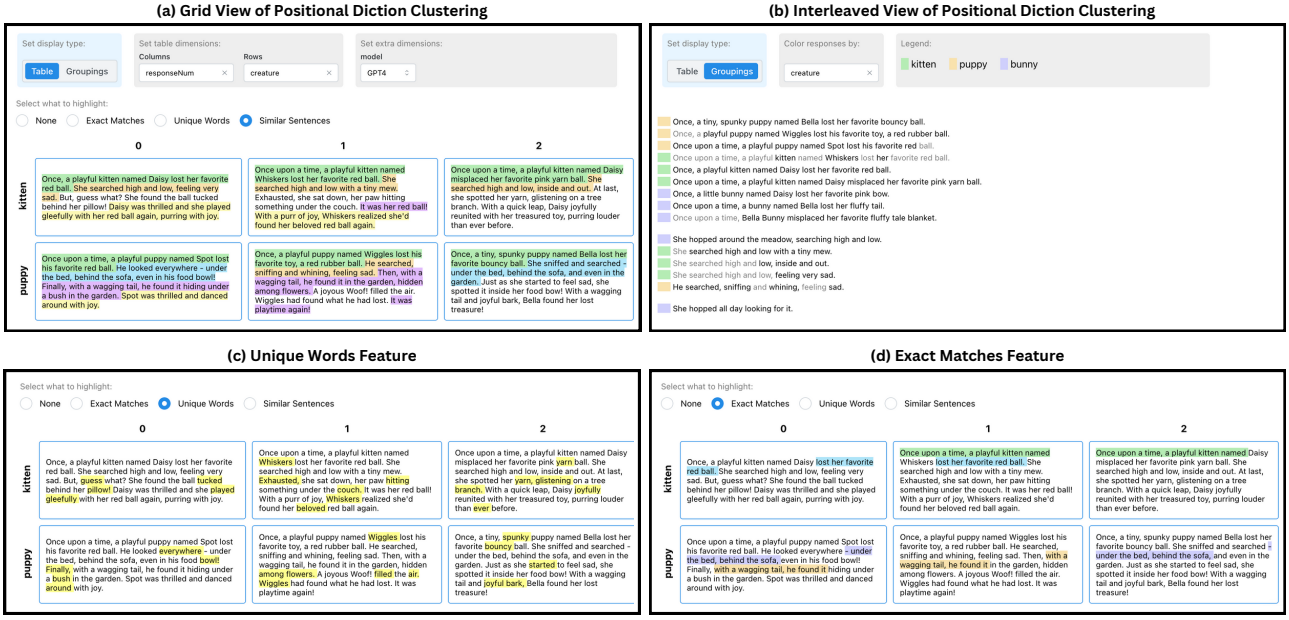


Figure 1: Our exploratory interface instantiates five combinations of text analysis (unique words, exact matches, and a novel algorithm we call Positional Diction Clustering) and renderings (within a grid with highlights or interleaved with grayed out redundancies), which can help users scale up the number of LLM responses they can reason about, e.g., for ideation, model comparison, or response selection. These figures represent how four of the five different combinations render the top of a page of a large collection of LLM responses generated from the test prompt "*Write a short story for a five year old child about a* `{creature}` *that loses something and then finds it again*" for three values of `{creature}`: `kitten`, `puppy`, and `bunny`. The fifth view tested—a grid layout without any visual additions based on text analysis—is not shown.

ABSTRACT

Large language models (LLMs) are capable of generating multiple responses to a single prompt, yet little effort has been expended to help end-users or system designers make use of this capability. In this paper, we explore how to present many LLM responses at once. We design five features, which include both pre-existing and novel methods for computing similarities and differences across textual documents, as well as how to render their outputs. We report on a controlled user study (n=24) and eight case studies evaluating these features and how they support users in different tasks. We find that the features support a wide variety of sensemaking tasks and even make tasks tractable that our participants previously considered to be too difficult to attempt. Finally, we present design guidelines to inform future explorations of new LLM interfaces.

CHI '24, May 11–16, 2024, Honolulu, HI, USA

ACM ISBN 979-8-4007-0330-0/24/05
https://doi.org/10.1145/3613904.3642139

CCS CONCEPTS

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

KEYWORDS

large language models, language models, foundation models, sensemaking, variation theory, analogical learning theory, reading, skimming

ACM Reference Format:
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 CHI Conference on Human Factors in Computing Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA.* ACM, New York, NY, USA, 21 pages. https://doi.org/10.1145/3613904.3642139

1 INTRODUCTION

Large language models are capable of generating multiple different responses to a single prompt. While several tools have recently been developed for structuring the generation of prompts and collecting responses [4, 42, 46], relatively little effort has been expended to help either end-users or designers of LLM-backed systems to reason about or make use of the *variation* seen in the multiple responses generated. We see utility in helping users make sense of multiple LLM responses. For instance, users may want to select the best option from among many, compose their own response through bricolage, consider many ideas during ideation, audit a model by looking at the variety of possible responses, or compare the functionality of different models or prompts.

However, representing LLM responses at a scale necessary to see the distribution of possibilities also creates a condition where relevant variation may be hidden in plain sight: within a wall of similar text. One could turn to automatic analysis measures, but we constrain ourselves to showing the entirety of the text itself, as this does not constrict, either by top-down design or bottom-up computation, which variations will be most useful to the user.

Based on our formative study interviews with a domain expert end-user and multiple system designers, model characterizers, and model auditors, we found that these users typically engage in iterative inspection of 10s to 100s of LLM responses via a chat interface (which is slow) or by pasting responses into a spreadsheet (which is arduous and clunky).

This scale of 10s to 100s of LLM responses, where formative study participants reported spending time sensemaking and making decisions about model and prompt selection, is greater than the inspection of one to two outputs at a time (as is common in a chat or 'playground' type environments) but less than the many thousands of outputs typically involved in annotation studies. We therefore call this *mesoscale* ("middle scale") of LLM response sensemaking.

In order to support users' sensemaking of LLM responses at the mesoscale, we designed and implemented several existing and novel text analysis algorithms and rendering techniques, each of which captures one or more aspects of LLM responses' possible variation and consistency. Established theories of human cognition describe how exposure to variation and consistency within prescribed structures can help people more robustly form mental models of a phenomenon, e.g., how an LLM behaves. Specifically, in line with Variation Theory [35], the features we instantiate identify patterns of consistency (Figure 1d, "Exact Matches"), variation (Figure 1c, "Unique Words"), or both (Figures 1a, 1b, "Positional Diction Clustering (PDC)"—a novel algorithm we introduce in this paper). In line with Analogical Learning Theory [13], PDC highlights analogous text across LLM responses, i.e., positionally consistent and similar in diction, such that users can see emergent relationships.

By evaluating these interface features, we shed light on the answer to our research question:

> *RQ: How can text-rendering interface design better support sensemaking of LLM responses at the mesoscale?*

To understand users' sensemaking support needs and how well these features did or did not serve them, we ran a controlled user study and a series of eight open-ended case studies. In the controlled user study, our system was compared to a baseline interface where responses are presented linearly in groups based on the model or prompt; we investigated tasks that varied both in the number of responses that participants saw (from 9 to 50) and the kinds of sensemaking involved (an email rewriting task and identifying differences between models). In the case studies, participants were asked to use our system to support their own real world LLM uses. The case studies covered a range of tasks, including poetry and fiction writing, identifying social bias, and investigating LLM-generated legal advice, to name a few.

We report on a number of themes that emerged through these studies. These include the kinds of subtasks users perform (e.g., detecting stylistic versus content variation), approaches to these tasks (e.g., hypothesis confirmation), information processing styles (e.g., preferring an overview versus pagination through subsets), as well as user hesitancy to inspect too many responses at once (which was often overcome when exposed to our features).

The contributions of this paper are:

- Formative studies that collect evidence of mesoscale text analysis of LLM responses for a variety of use cases.
- A controlled user study and open-ended case studies that demonstrate how our interface features can make sensemaking of LLM responses easier, and that many LLM-related tasks are intractable with current interfaces.
- A novel algorithm identifying similarities and variations across LLM responses, called Positional Diction Clustering (PDC), as well as a novel rendering for presenting the results of PDC.
- Design implications for future work on LLM response inspectors.

In the discussion, we outline future design directions inspired by our findings, consider limitations to our approach to manual inspection, and reflect on the similarities and differences between designing for LLM response inspection versus other kinds of textual or machine learning data.

2 RELATED WORK

The objective of this work is to explore both the tasks for which is it helpful to view many LLM responses at once in a single scrollable page, as well as the possible interface supports that could help users complete their tasks. Here we review precedents and influences on

our work along three axes: skimming support, cross-document text visualization, and sensemaking interfaces for generative AI.

2.1 Skimming and Skimming Support Tools

Reading and skimming are two distinct cognitive processes. Reading involves a sequential and comprehensive engagement with the text, whereas skimming is a strategic, selective, and non-sequential form of reading focusing on extracting the most salient information quickly [1]. Skimming requires focused attention and strategic choices from the reader, which present additional cognitive challenges [11]. Despite its challenges, eye-tracking studies have found that skimming is very common because of the time it saves [12, 40]. In this work, we use a combination of word-, phrase-, and sentence-level highlights to call attention to text that may be relevant to sensemaking [41] tasks.

So far, the focus of skimming research has predominantly been on individual documents. However, our investigation diverges from this norm by emphasizing comparison across documents. Traditional definitions might not classify such comparative reading as "skimming," yet we posit that skimming operates as a fundamental cognitive process when readers contrast texts. In the next subsection, we consider work on comparing documents. By consciously integrating skimming into the document comparison paradigm, we bridge these two realms of study, aiming to unveil innovative and more efficient interfaces for text comparison.

2.2 Cross-Document Text Visualization

While skimming and reading strategies for a single document are well-established, generalizing to a collection of documents introduces the challenge of visualizing similarities and differences across documents [8]. Many systems have been proposed towards this goal, but they either abstract away documents as a dot on a 2D plane [27, 31], as word pairs within a word cloud [10, 52], or as nodes within a graph [20, 21]. Direct access to the text itself, which is a precondition for the user to consider differences across texts, is often only possible as short document excerpts retrieved by a query containing a selected phrase or linguistic pattern (e.g., choosing a particular search term for text sliding [38]), hovering and clicking over abstract document representations to open a full-text view, or drilling down into deeper layers of the interface. In this work, we render all the documents (i.e., LLM responses) in their entirety as well as their relationships at a close textual level without requiring an initial query, as users may not yet know what an appropriate query would be, or even what they are looking for. This may be possible for LLM response inspection without overwhelming users in large part because of the generative processes that produce the responses.

One approach to visualizing text across documents involves leveraging the structure and layout of textual documents to aid skimming and comparison of text. For instance, VarifocalReader [28] supports skimming large complex text documents by using a multi-level layout where abstract summaries of varying detail are shown alongside the source document itself. The role of layout is important, especially in the early stages of interacting with a document, because readers often scan a document before delving into its details [29, 32]; there is a human tendency to treat words as "locations in space" [48].[1] Furthermore, the visual structure of documents can profoundly influence readers' comprehension by affecting readers' assumptions, reading strategies, willingness to read, cognitive costs, and effort they must make to read [26, 37, 39, 49]. In our work, we take advantage of the spatial aspects of document comprehension by decorating segments of text with highlights to show pre-computed cross-document relationships. We hypothesize that this will engage both visual and spatial memory as well as pattern recognition.

Text alignment is often used to facilitate direct comparison of the text across multiple documents, which can involve designing an algorithm for identifying shared patterns across texts and a method of visualising those shared patterns [53]. For example, Tempura [50] uses "structural templates"—structures of linguistic features—to find and summarize patterns in a corpus of queries collected by search engines and intelligent assistants. (Note that these queries tend to be shorter and more structured than ordinary text.) Other work on text alignment has treated it as a sequence alignment problem and focused on algorithms involving edit distances and document-to-document matrices [9, 36]. CollateX [19] introduced variant graphs to enable the comparison and alignment of more than two documents and integrated the process into the digital collation workflow. However, the proposed alignment algorithms are still limited to single sentences with highly similar sentence structures [19]. Gero et al.'s preliminary work presented a sensemaking interface that uses concordance tables to display LLM responses to support users in investigating problematic responses and distribution shifts across responses [15], but this preliminary work was not evaluated and was designed for single sentences rather than entire multi-sentence LLM responses.

Related work can also be found in the space of rendering code corpora, in particular OverCode [16] and Examplore [17], which pre-compute and render sub- and cross-document relationships using visual text attributes and spatial layout. Both OverCode and Examplore render entire corpora of text (code) with the same or similar purposes: a corpus of Python solutions to the same programming problem and a corpus of Java code examples that all call the same API, respectively. However, code and natural language text present distinct challenges in terms of cross-document comparison. And while our features also precompute similarities and differences, there is no abstraction away from the source documents (as in both OverCode and Examplore) nor a pre-defined template for identifying analogous components (as in Examplore).

LLM responses are a timely, important, and distinct type of corpus, and our features' designs have been inspired by—and, as in the case of PDC, are necessarily unique from—prior systems' features, as these features did not transfer from other domains without significant insight.

2.3 Sensemaking Interfaces for Generative AI

Generative AI models span multiple modalities, and interfaces for helping users understand and leverage their stochastic capabilities are in their infancy. Many systems in adjacent fields that aim to facilitate comparison among types of data other than text have

[1] This is why people often remember where on the page a given piece of information was located, even if they cannot remember the information itself.

adopted a grid view where information is organized along columns and rows. For instance, Mesh [7] helps consumers evaluate evidence about a product gathered across many different sources on a grid view, where columns are options to choose from and rows are criteria. Similarly, MLCube Explorer [25], an interactive visualization tool for comparing machine learning results, uses a grid view where each row represents a subset of the data and the columns are summary statistics for each subset. In our work, the grid layout is specifically inspired from text-to-image generation "style guidelines" where AI-generated images are laid out in a grid that reflects controlled variations in the prompt [34]. For our design, the variation rendered in the grid is the result of multiple draws from the stochastic LLM, and, where indicated, different LLMs or variations on a prompt.

The public availability, broad applicability, and performance of LLMs specifically has increased their adoption for a diverse range of applications that vary in domain and complexity. Prior work has shown that LLMs often generate long responses that negatively impact the user's ability to understand and interact with the given output [23]. To mitigate this, there are interfaces to assist users with sensemaking and evaluating responses generated from LLMs. For instance, Graphalogue [23] and Sensescape [44] transform long LLMs responses into diagrams that connect the concepts in the response using a graph. These systems enhance the rendering of individual LLM responses, rather than rendering the distribution over possible LLM responses to the same query, and are therefore do not explicitly address the stochastic nature of LLMs. Meanwhile, Promptfoo [42] was developed for prompt engineering and evaluating prompts against predefined test cases; users can view prompts and inputs in a side-by-side display but the focus is on supporting automatic evaluation. In this work, we assume that users either do not have automatic quality measures for their task or that they do not yet know how to precisely define their goal. We seek to encourage inspection of outputs and do not abstract away from the text itself, as other prior very preliminary work has done [15].

3 FORMATIVE INTERVIEWS

We interviewed eight people working with or developing systems with LLMs. In semi-structured interviews, we asked interviewees about their process, how they selected pre-trained and/or fine-tuned models, and how any prompt engineering was structured. The interview guide can be found in Appendix A.

3.1 Participants

Participants were recruited through the authors' professional networks. Our interviewees included two doctors investigating LLMs in medical contexts, one researcher investigating the creative abilities of LLMs, three start-up founders or CTOs (from three different companies) who oversee the development of LLM-based public-facing products that support writing, and two artist-researchers who build and interrogate LLM attitudes towards queer identities.

3.2 Findings

Overall, we found that everyone we interviewed engaged in manual inspection of outputs. This happened at different scales: from comparing two or three outputs to reading a list of 1000 outputs. (As described in the introduction, we refer to the middle of this range as the *mesoscale* for text analysis.) Sometimes it involved clearly-defined annotation, but often it involved discussions among the system designers or with users. All interviewees had developed ad-hoc processes to support this inspection. Many noted that they put outputs in a spreadsheet, as this facilitated both sharing of outputs as well as increasing their readability. In addition, three main themes emerged:

3.2.1 Failure of Automatic Evaluation. All interviewees said that automatic evaluations were not useful when it came to developing LLM applications. Several interviewees had explicitly investigated if benchmark evaluations of LLMs correlated with which model worked best for their use case and found no correlation. Because automatic evaluations could not predict success at their task, all interviewees engaged in some kind of manual inspection for evaluation.

3.2.2 Sensemaking of LLM Responses. Interviewees discussed a sensemaking process with LLM responses. The artist-researchers discussed reading and comparing outputs as a key part of their development process, to understand if a model they were training was "getting better" or to compare how models treated heterosexual vs. homosexual couples. The researcher working on creative writing applications discussed both the prompt engineering he did to develop the system as well as the prompt engineering he saw his users engage in, e.g., comparing model capabilities for prompts in first v. third person. One of the start-up co-founders mentioned a variety of kinds of manual inspection that involved sensemaking, from panel discussions with users to internal evaluations where "we rely on our own literary skills to evaluate the outputs." Another start-up founder described their process as putting a few people in a conference room with a lot of outputs and having them read them all. In particular, he noted the importance of detecting problematic outlier outputs, because preventing these was key to developing and maintaining user trust. These observations show the importance of sensemaking to the use of LLMs and the range of forms sensemaking takes.

3.2.3 Complex or Nuanced Annotation. The third start-up founder described their process as printing out a thousand outputs and reading them all manually, noting which ones "worked" for their intended use case and which did not, and using this ad-hoc, manual annotation to select an appropriate fine-tuned model. Both doctors we interviewed were engaged in research projects evaluating LLMs in a clinical setting. One doctor described the detailed and highly-skilled human annotation involved in evaluating LLMs, and noted that the same grading metrics used for evaluating doctors were being used to evaluate LLM responses.

4 FEATURE DESIGN: EXISTING AND NOVEL SENSEMAKING SUPPORT FEATURES

We designed and prototyped instantiations of several existing and novel algorithms and renderings for scaling up LLM response sensemaking. Each highlights or juxtaposes words, phrases, or entire sentences based on their relationship to the entire collection of LLM responses. We view these features as *technology probes* [22]: a

non-exhaustive set of points in a design space for LLM response-rendering interfaces that scale up human inspection.

4.1 Design Goals

4.1.1 Scale up the number of LLM outputs a human can consume. We aim to make 10s to 100s of LLM responses cognitively comfortable to peruse, as this was the scale we found to be most relavent in our formative study. This range is relevant for a variety of tasks. When writing an email, a writer may have an easier time recognizing tone and diction variation across 10 different LLM responses. When looking for inspiration, a designer may look at 50 responses to surface diverse possibilities. When checking for outlier responses, a system developer may want to look at 100s of responses.

4.1.2 Show the entire collection of LLM outputs as text—not abstractions. In our formative study, we found that automated analysis rarely captured what the participants were looking for when inspecting LLM responses. The choice to have minimal textual abstractions in prior publications like OverCode [16] served users well: users could recognize what *they* thought were good and bad aspects of programming composition in student solutions which were worthy of comment. Similarly, we think users in our context should be able to see all of the responses and perform their own sensemaking.

4.1.3 Do not require users to select text "lenses" with which to see the data. Some text analysis tools require users to select (or accept recommended) search terms in order to access the most powerful text analysis algorithms and renderings [38]. However, in our formative study, participants seemed to prefer engaging with the text directly without having to articulate a lens with which to look at the corpus, since their analysis goal may be initially under-defined. For this reason, we want to instantiate flexible features that allow users to immediately inspect the entirety of LLM responses without requiring the user to choose or accept a particular lens (e.g., search term) with which to render them.

4.1.4 Show pre-computed relationships within the rendering of the text itself. Like prior work on rendering sub- and cross-document relationships within a textual corpus [16, 17], we want to decorate text to show pre-computed relationships, such as string matches or analogous sentences, across responses. In this way, we help users shift cognitive bandwidth away from identifying overlapping or "unique" language to answering more complicated questions. Additionally, we create skimmable visual patterns across all of the responses.

4.1.5 Support a variety of sensemaking sub-tasks for both system designers and system end-users. We want to support a wide range of tasks that involve sensemaking. For example, we want to support the detection of similarities and differences between individual responses as well as groups of responses, and support the detection of "outlier" responses (or parts of responses). In alignment with sensemaking literature [41], the goals of user tasks could be very open-ended or be hypothesis driven depending on the task itself and how "far along" a user is in the sensemaking process.

4.2 Relevant Theory

By interacting with a system, users may develop a mental model that guides their future interactions with it [43]. Variation Theory [35] and Analogical Learning Theory [13, 14] each propose mechanisms for how people may conceive and update their mental models based on concrete examples, or use their mental model in new situations. Extensive evidence congruent with each theory has been collected in many domains [2], though not yet for mentally modeling LLMs or mentally modeling the space of possibilities, e.g., in an ideation task, generated by an LLM. One prior piece of HCI work, ParaLib [51], does explicitly exploit these theories for system feature design, but does this in the domain of code.

Variation Theory describes how helping people perceive the different dimensions of consistency and variation across examples (here, LLM responses) of the object of learning helps them more quickly and robustly leap to more accurate mental models. Analogical Learning Theory describes how people can form mental models or schema from perceiving structural analogical relationships across superficially varying examples (again, here LLM responses). In this work, in line with Variation Theory, the existing and novel features instantiated and described in the next subsection collectively identify patterns of consistency, variation, or both; they are explicitly designed to make emergent dimensions of consistency and variation easier for the user to perceive.

4.3 Feature Descriptions

We implemented our LLM response sensemaking features within a fork of the open source project ChainForge [5], which is a visual authoring interface for generating, inspecting, and analyzing LLM responses.[2] The features are a combination of text analysis algorithms and rendering. The initial two algorithms and first layout are conceptually straight-forward extensions of existing features, while the final novel algorithm and its corresponding custom layout are designed specifically for analyzing and rendering collections of LLM responses that are, by construction (e.g., multiple draws from the same model), variations on a theme.

4.3.1 Exact Matches. This conceptually straight-forward feature enables users to see how similar responses are by finding and identifying "exact matches" across responses. There are many ways to implement this functionality. We detect and highlight the longest common substrings, as that appeared to be most robust to a wide variety of response types during informal evaluations. The hypothesized benefit of this feature is that users can shift cognitive bandwidth from recognizing overlapping language to answering more complicated questions. Additionally, it gives a skimmable sense of literal repeating patterns across all of the responses.

Algorithm and Rendering. We identify substrings to highlight in five steps: (1) find common substrings in pairs of responses, (2) split substrings that span sentence boundaries, (3) filter short substrings, (4) rank them based on a combination of substring length and how many responses they appear in, and (5) retain the top $k = \min(12, |\text{responses}|/2)$. For details see Appendix subsection B.1.

[2] ChainForge, and the text rendering functionality we built atop it, is written in Typescript. Our code imports the ChainForge-retreived LLM responses as a list of JSON objects and renders them.

Each set of exact matches is highlighted in its own color; examples are shown in Figure 1(d) and Figure 2.

4.3.2 Unique Words. This feature allows users to see what is distinctive about responses by highlighting "unique" words in each response. There are many ways to measure uniqueness; we use a simple measure that has been widely studied in Natural Language Processing and Information Retrieval: term frequency-inverse document frequency (TF-IDF) [24]. TF-IDF uses the frequency of a word in a given document as well as its frequency across all documents to calculate a score for how "representative" a word is of a document relative to the rest of the collection.

Algorithm and Rendering. We calculate TF-IDF values using the Wink-NLP Javascript library [47]. The set of documents is defined as the set of responses generated by the selected LLM(s). For instance, if the system was run with two models, three prompt variations, and three responses for each combination, there are $2 * 3 * 3 = 18$ responses, which comprise the set. Term frequency is calculated for each response. We remove stop words. We highlight the five top-scoring unique words in each response. See Figure 1(c) and Figure 3 for examples.

4.3.3 Positional Diction Clustering (PDC). This novel algorithm is designed to identify, when present, any emergent structure (and variation within that structure) in a set of LLM responses by finding sets of sentences across different responses that are similar in textual content, i.e., diction, *and their location within their respective responses*. One could refer to these sentences as *analogous sentences* across responses. Sentences in any single response which do not have analogous sentences in any other responses are preserved as singletons.

Theories of human concept learning suggest that a key step in forming accurate, robust mental models of a phenomenon is to be able to discern the underlying dimensions of variation (Variation Theory) and any latent structures beneath superficial details (Analogical Learning Theory). By detecting and communicating which sentences are both structurally analogous (by virtue of their position within the response) and semantically related (by virtue of highly overlapping content), users should be able to more easily identify emergent structures, as well as compare and contrast particular compositions of structural elements across responses and syntactic elements that may vary in meaningful ways across analogous sentences within those responses. These theories assert that these subtasks are key ingredients in forming those robust accurate mental models, i.e., learning from the LLM responses in order to better perform their overarching task.

Algorithm and Rendering. We create groups of sentences using single-link agglomerative clustering. For every pair of sentences, we calculate their content similarity as exact diction overlap normalized by their combined length.[3] Initially, every sentence is placed in its own group. Then, we iterate through all sentence pairs, in decreasing order of content similarity. For each pair, we merge the groups that contain those sentences if (1) the two sentences are sufficiently similar in content and normalized location in the LLM responses[4] and (2) merging creates a group where at least 70% of sentences are all from different responses, since we are interested in analogous *cross*-document sentences. Finally, for each group, we calculate the mean normalized location in the LLM response of sentences in the group. Groups of one are permitted; these capture sentences that have no analogous sentence (in terms of both content and approximate normalized location) in any other response. The algorithm returns (a) all the groups, including groups of one, and (b) the mean position for each group. This algorithms' results can be rendered in one of two ways: using the same color to highlight all the sentences that share a group, as is done in the Grid Layout (shown both in Figure 1(a) and Figure 4), and listing sentences, by group, in order of their groups' normalized mean location within the responses, as is done in the Interleaved Layout (subsubsection 4.3.5), which is specifically designed as an alternative layout for the results of PDC (shown in Figure 5).

4.3.4 Grid Layout. LLM responses are laid out in a grid, with user-defined variables for the columns and rows. For instance, users can select that the different models queried determine the columns, and repeated generations from the same prompt determine the rows. This view allows users to see many responses with controlled variations (model, prompt, temperature, etc.) side by side. For example, in Figure 1, the template prompt asks the model to generate a short story. Here, prompt variations—different possible story characters—define the rows, and the $n = 3$ different responses per prompt define the columns. There may be more than two variables a user is interested in, for instance also comparing models. In the top right of the interface, the user must select which value of the remaining variables to surface. Currently, the grid does not support viewing more than two variables (i.e., the column and row variables) at a time, though extensions could allow this.

There are two hypothesized benefits of this view. One is based on an understanding of human perception: the grid layout should help users compare more LLM responses because the spatial arrangement assists their memory. The other benefit is based on Variation Theory, which posits that discerning the impact of a critical aspect, for example model temperature, is only possible when experiencing variation along that dimension, isolated from variation along other dimensions. The user-configurable grid layout makes it possible for users to isolate dimensions of variation, enabling them to discern their qualities and critical values, for instance how model version affects the language of its responses to the same prompt.

4.3.5 Interleaved Layout (enabled by PDC). Sentences from different responses are strategically interleaved. The rendering (Figure 1(b) and Figure 5) is generated by printing out the groups produced by the algorithm. Groups are ordered based on the average position calculated by the algorithm. This means they roughly follow the flow of most responses. Each group is rendered with one sentence per line, in an order that maximizes exact word overlap between *adjacent* sentences in the group. If a sentence has any exact word overlap with the sentence above, i.e., if the i^{th} word in both sentences is the same, the word in the sentence below is grayed out. A small amount of whitespace separates each group. A colored

[3]This is a slight variant of Bray–Curtis Similarity. For details see the Appendix subsection B.2.

[4]See the Appendix subsection B.2 for the calculation, threshold values, etc., used in user study.

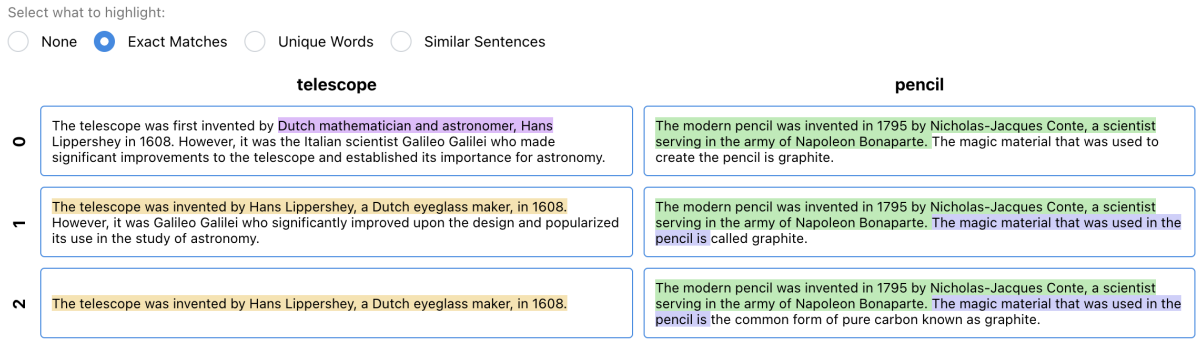


Figure 2: Example of the 'exact matches' feature for the prompt "*Who invented the* `{object}`*?*" where the objects are `pencil` and `telescope` and each prompt had $n = 3$ generations. Exact matches makes it easy to identify portions of responses that are matching across multiple responses.

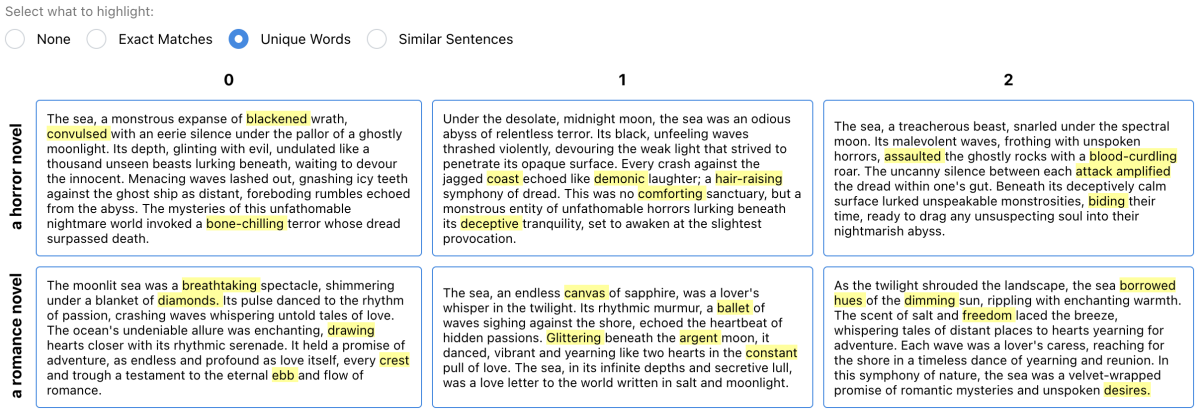


Figure 3: Example of the 'unique words' feature for the prompt "*Write a short paragraph about the sea in the style of* `{style}`." where the styles are `a horror novel` and `a romance novel` and each prompt had $n = 3$ generations. Unique words makes it easy to see how word choice is influenced by the style.

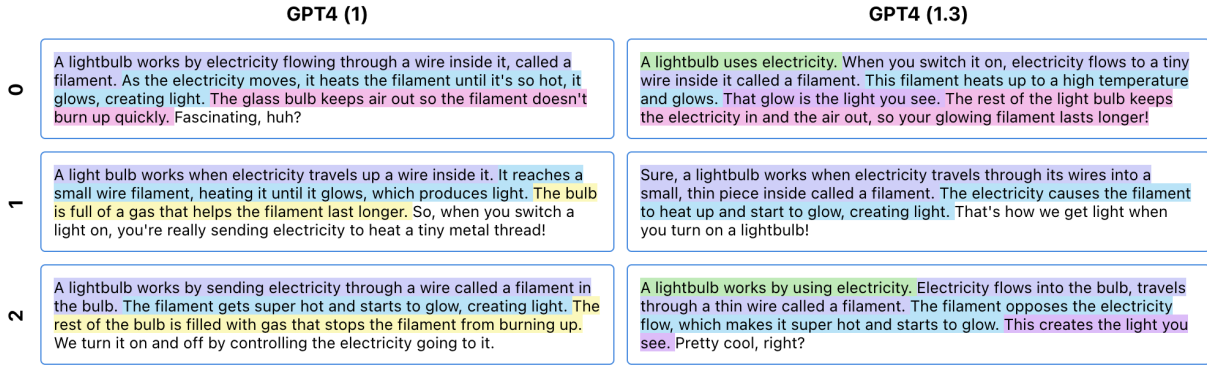


Figure 4: Example of the PDC feature in the grid layout for the prompt "*Explain how a lightbulb works to a 12 year old.*" for GPT4 temperature=1 and GPT4 temperature=1.3. In the grid view, sentences with similar relative position and diction are highlighted in the same color; notice that the sentences highlighted in yellow are both about how gas supports filament longevity.

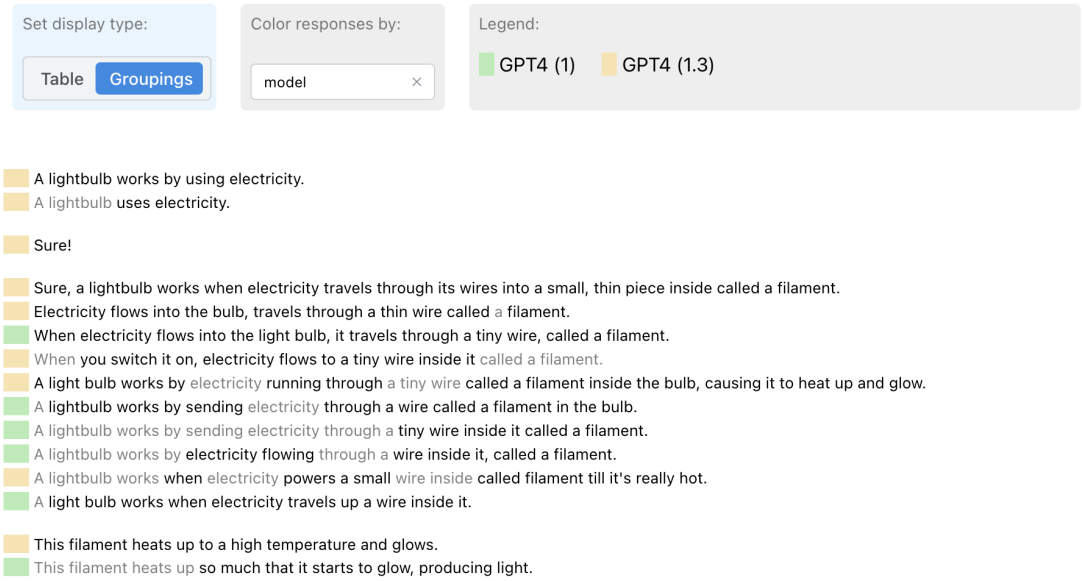


Figure 5: Example of the PDC feature in the interleaved layout for the same prompt, model, and temperature settings as in Figure 4, i.e., "*Explain how a lightbulb works to a 12 year old.*" for GPT4 temperature=1 and GPT4 temperature=1.3. In the interleaved view, sentences with similar relative position and diction are grouped, with the color patch to the left indicating which model version produced them; notice that all the opening 'topic' sentences are shown together with redundant text grayed out.

square to the left of each sentence indicates which prompting condition generated it. Note that, due to the prototype implementation not ignoring punctuation (e.g., differentiating between `Once` and `Once,`) fewer words were grayed out than a user might expect, e.g., the adjacent `Once`'s with and without commas in Figure 1(b). We do not believe this significantly affected the study results.

We hypothesise that this rendering will (1) be particularly useful for users who wish to remix parts of several different LLM responses, (2) support the identification of rare response components, as they will be easily identified as singleton groups, and (3) support characterizing the complete distribution of LLM responses, since the volume of screen real estate taken up by any given group is proportional to the number of responses that include a sentence in that group.

5 USER STUDY

We instantiate these features into a single interface, which we call the "exploratory interface". We also implement a baseline interface that represents a status quo in LLM response rendering: a linear list of responses, as one might get from an API call or pasting responses into a spreadsheet, as some formative interviewees did. The baseline interface has two additional capabilities: listing responses grouped by model or prompt, and putting each group in a collapsible container, such that users can have as many groups open at a time as they like. Within each group, responses are presented linearly top-to-bottom. See Appendix subsection C.3 for a screenshot.

We ran a controlled user study that investigated when and how the different features were helpful relative to the baseline interface. This allowed us to investigate:

> *Which features best support sensemaking tasks over many LLM responses?*

The user study put participants in two different scenarios (writing an email, comparing two models) at two different scales (10 and 50 LLM responses, respectively). This study evaluates these features' utility for end-users who attempt these tasks.[5]

5.1 Participants

Participants were recruited from a local university via mailing lists and flyers. Participants had to be over 18 years old, fluent in English, and a student of some kind (undergraduate, masters, or PhD).[6] We recruited 24 participants (11 women, 12 men, and 1 non-binary). Eight participants were 18-24 years old, and the remaining 16 were 25-34. All were graduate students of some kind (15 Ph.D. students, 9 Master's students). In the pre-study survey, we asked participants about their confidence in writing important email and their experience with LLMs. Response details can be found in Appendix subsection C.4. Most participants felt very confident writing important email and had some experience with LLMs. Few participants had explored the differences between LLMs before.

5.2 Study Procedure

All studies took place in person. Participants completed the tasks using the facilitator's laptop. Facilitators were authors of the paper.

[5] In the case studies we investigate how more targeted user groups (like researchers and system designers) use the features in self-selected and self-directed tasks.
[6] Although narrowing our participants to students is not necessary to answer our research questions, by narrowing the pool to students we were able to pick tasks that were more likely to be personally compelling and realistic.

The general flow of the study procedure is rendered visually in Figure 6, and is described in more detail in Appendix subsection C.1. The exact prompts can be found in the Appendix subsection C.5.

5.3 Task 1: Email (Re)writing

We chose an email (re)writing task as prior literature has used this task with LLMs [6, 18] and we found this is a common use case of LLMs [30, 33]. Additionally, we hypothesized that users may benefit from seeing multiple variations at once, rather than one at a time (as would be the case in a chat interface) as users may better be able to compare and contrast responses to select the best one, and may be more able to recombine elements of different versions into their final draft.

Participants were shown nine different LLM responses that each rewrote the same initial email draft. The prompts used in the study were chosen to be realistic for the recruited participants (university students): asking a professor for a recommendation (Task 1A) and asking an internship manager for a later start date (Task 1B). Participants were asked to first select the LLM response that was closest to the one that they would send in real life. We gave participants up to 3 minutes to complete this part of the task, given that pilot participants typically did not take more than 2.5 minutes. Then, participants were asked to edit their chosen LLM response to make it closer to what they would want to send. We mentioned that participants could also remix parts from different LLM responses. We gave participants 2 minutes for this part of the task, as most participants could complete it within this time and the outlier participants would be prevented from editing for too long.

5.4 Task 2: Model Comparison

In this task, participants were asked to compare models; they looked at 25 responses from GPT-3.5 and 25 from GPT-4. We selected this task to be one that users may want to engage in (compare model behaviour) and hypothesized that it may be quite difficult with status quo tools. There were two prompts, one for Task 2A and one for Task 2B: asking for advice about how to skim a book and asking for advice on how to prepare in the week before an important final exam. The participants' task was to list as many differences as they could between the two models' responses. This task is quite open-ended. In pilot studies, participants needed at least 5 minutes to come up with even a few model differences. However, some participants found the task exhausting and would ask to give up around 8 minutes. For this reason, we gave participants 10 minutes for this task, and allowed them to stop early if they felt they had completed the task to the best of their ability.

5.5 Analysis

5.5.1 Quantitative Analysis. We ran statistical tests to compare responses to all the Likert scale survey questions as well as time on task. In the model comparison task, to determine how many differences were found by each participant, we first put all participants' listed differences into one list and had one author, who was blind to participant and condition, clean the data manually.[7] Then we use counts of how many differences each participant wrote down in each condition.

5.5.2 Qualitative Analysis. Two authors followed a general inductive approach for analyzing qualitative data [45]. Both listened to all interviews to gain context, and, via transcriptions, pulled quotes relevant to the research questions. With a shared set of quotes, the researchers independently came up with codes for the quotes, then came together to discuss and create a codebook. This codebook was shared with other members of the research team, discussed and revised. With the revised codebook, the two authors re-coded half of the data, and disagreements were discussed and codes were revised based on discussion. Finally, the remaining data was re-coded by a single author.

Table 1 shows the results of our analysis of the interviews with participants. These themes reflect a wide range of participant responses, from subtasks performed (e.g., detecting diversity of responses) to methods for performing these subtasks (e.g., confirming hypotheses) to the cognitive elements involved in performing them (e.g., focus and working memory). We use these themes when analyzing how participants did or did not make use of the features in each of the tasks in the user study, as seen in the subsequent two subsections, but also present them as a result of the study that can inform future system designers and user studies, as they reflect a range of methods, issues and concerns that come up when users inspect LLM responses.

5.6 Email (Re)writing Results

5.6.1 Few Quantitative Differences Between the Exploratory and Baseline Interface. Participants completed Task 1B faster than Task 1A, and felt less rushed. For this reason, we split out analysis comparing interface conditions between Tasks 1A and 1B; further details can be found in Appendix subsection C.10. Participants in both interface conditions and scenarios were quite successful at the task, with all participants rating their final edited email as above a 5 on a 7-point scale where 7 was "I would definitely send this email." There were no significant differences between the interface conditions for either Task 1A or 1B for how successful they were at the task (how likely they were to send the email) nor how long they took. This suggests that the exploratory features did not obviously impact participants' ability to do the task, perhaps because the task was easy enough to achieve near perfection in both interfaces. When asked which of the two interfaces was easier and which was more overwhelming, there was no clear preference; participants varied in which interface they preferred.

5.6.2 Qualitative Differences Between the Exploratory and Baseline Interface. Despite in aggregate there being few differences between the interfaces, there did exist preferences that varied across participants.

In the exploratory interface, both the layout and multiple different highlighting features were explicitly called out as helpful, relative to the baseline interface. Some participants appreciated being able to see all responses at once: *"I think that's what's nice about the grid; [the emails] are all right there. And so your eyes can sort of flutter and dart around as you're reading each one"* (P9). Some participants appreciated how the highlighting features allowed them

[7] This was because some participants would write paragraphs instead of bullet points, or include two differences in a single bullet point.

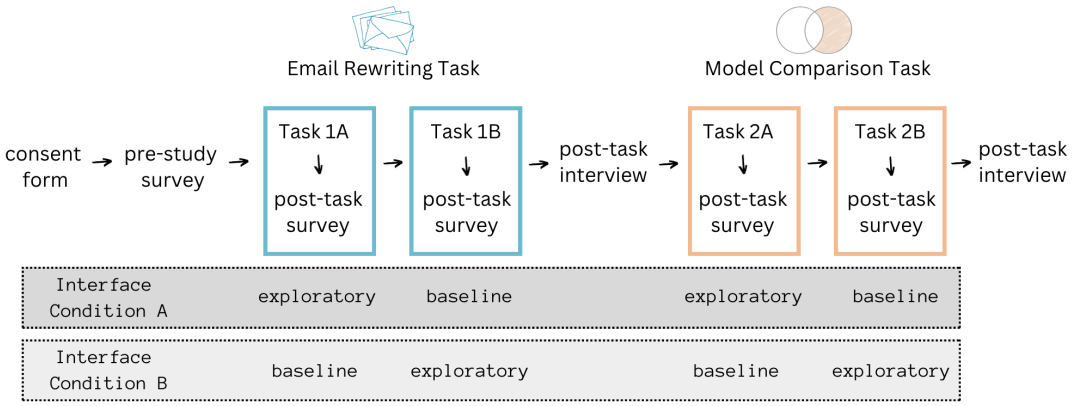


Figure 6: Study Process: each participant performs two email rewriting tasks with different UIs and two model comparison tasks with different UIs. Interface conditions are counterbalanced.

Table 1: Codebook.

Code	Definition
Kinds of Similarities and Differences Noticed	
style	Stylistic or tone similarities and differences.
content	Content similarities and differences, including counting phrases and the length of responses.
granularity	Level of abstraction of similarities and differences, e.g., high level v. low level or "granularity".
structure	Structural similarities and differences between responses, including ideas of responses being 'segmented'.
outliers	Identifying outliers or unique features.
diversity	Determining consistency or diversity across sets of responses.
Preferences about Interface Elements	
display	Preferences about viewing all or many responses at once.
scroll	Preferences about needing to scroll to return to responses.
options	Preferences about the ability to disable features.
delineation	Preferences about delineation between prompt groups (e.g., responses to different prompt variations).
visual learner	Preferences about learning or processing information visually.
Elements of Reading Process	
speed	What made reading responses faster or slower.
read all	Reading the entirety of groups of responses (as opposed to skimming).
skim	Skimming groups of responses.
Cognitive Elements	
memory	Issues of working memory, forgetting response content or location, or needing to reread to recall responses.
ease	Cognitive demand of different interface elements.
overwhelm	Feelings of overwhelm or stress, especially in response to inspecting too much information at once.
exhaustion	Feelings of exhaustion or "zoning out".
focus	Ability (or inability) to focus on specific responses or groups of responses.
difficulty	Feelings of task difficulty, including not knowing how to start a task.
Methods of Detecting Similarities and Differences	
confirmation	Performing "hypothesis testing" or otherwise confirming or verifying an idea.
comparison	Directly comparing responses by having them nearby or side by side.
absent color	Making use of unhighlighted text segments.
Feature Accuracy and Understanding	
accuracy	Determining how accurate or trustworthy different features are.
not understanding	Not understanding how a feature worked or why it performed a certain way.

to easily compare segments of responses (using PDC in the Grid Layout) and identify stylistic choices that matched their own style (using the Unique Words feature). These participants also noted that, in the baseline interface, needing to scroll to revisit responses

was challenging because *"if you want to reference the first email that you read versus the ninth, you have to scroll up and then you lose the view of the ninth one"* (P9).

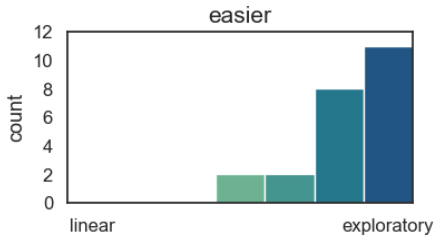
In contrast, other participants said that the exploratory interface felt overwhelming because seeing all nine at a time *"was too much information. . . . The [baseline linear view] was more organized"* (P1). In particular, the baseline interface allowed users to collapse groups of responses, allowing participants to focus on one group of responses at a time. Participant responses to the two interface conditions seemed to be influenced by their information processing style, something we saw come up in the case studies as well, and we discuss in further detail in the Discussion section.

5.6.3 Utility of Different Features. Most participants spent little time with any of the features, although some participants spent significant time in Grid PDC and Unique Words; details can be found in Appendix subsection C.10. The four participants who spent the majority of their time in the Grid PDC feature noted that because the email messages had a common structure, the highlighted sentences allowed for easy visual segmentation of the responses—this allowed participants to easily compare across the individual segments. One participant said that the Interleaved PDC feature matched the way he currently used LLMs for writing tasks, except he normally had to do it by manually copy and pasting sentences from multiple responses grouped by their semantic function, such that he could pick the best sentence per group. Two participants spent the majority of their time in Unique Words feature. One of these, P7, said that this feature made it *"easier for me to just look and see what makes each response unique."* Overall, participants found the Grid PDC most useful, but it also appears that this task may have been too easy to require much detailed exploration of the responses, as participants could read the entirely of all nine in just a few minutes. The next task proved more difficult, and therefore highlighted the utility of the features when users are more likely to struggle with existing interfaces.

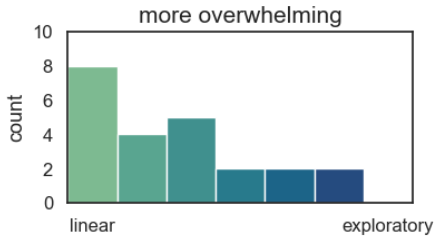
5.7 Results for Model Comparison Task

5.7.1 Quantitative Differences Between Interface Conditions. When comparing the interface conditions, we found no difference between the cognitive load questions. However, we did find that participants reported being more successful in finding differences with the exploratory interface than with the baseline interface ($p < .05$, Mann Whitney U task for nonparametric data). When participants were asked to rate which interface made the task easier and which was more overwhelming, participants reported a strong preference for the exploratory interface, as can be seen in Figure 7.

Participants reported feeling more successful with the exploratory interface, and reported that the exploratory interface made the task easier, but did they actually come up with more differences? Since we may reasonably expect that participants may get better at this task the second time around, we ran an ANOVA analysis of how task order and interface condition impact the number of differences found. Our dependent variable is the number of differences found, our fixed effects are task order and interface condition, and our random variable is participant ID. We find that the task order is not significant but the interface condition is ($p < 0.01$), demonstrating that the exploratory features did help participants find more differences.



(a) "Which interface made the task easier?"



(b) "Which interface was more overwhelming?"

Figure 7: Participant interface preferences for the model comparison task.

We found that participants spent significantly less time with the baseline interface than the exploratory one (baseline: 8.8 minutes; exploratory: 9.7 minutes; $p < .05$; two-tailed t-test for data with similar variance). Since participants also found more differences with the exploratory interface, it's reasonable to assume that there may be a correlation between time on task and number of differences found. We calculate the Pearson correlation coefficient and p-value for testing non-correlation, one-tailed where we test for positive correlation, and find that time on task and number of differences is significantly positively correlated ($p < 0.05$). This indicates that the exploratory interface may have allowed participants to perform the task better through increasing engagement, allowing participants to stay on task longer.

5.7.2 Qualitative Differences Between Interface Conditions. In the baseline interface, because participants had to scroll up and down to get to responses from the two models, many participants reported forgetting what they had determined about a single model. P2 suggested this was *"because we have a limited memory window,"* such that skimming through the linear view resulted in forgetting what they were even trying to discern. P13 described the exploratory interface as: *"We can literally see the difference, what this group says versus what that group says. So it gives a side by side comparison."*

All participants reported that the exploratory interface made the task easier because it allowed for easier recognition of similarities and differences between the two sets of responses. Participants used the features in the exploratory interface in a variety of ways, some of which we had imagined and some of which we did not; details can be found in the next section. While this may be an

expected result, we did not see this result in the email rewriting task, suggesting that the utility of LLM inspector interfaces may be dependent on the number of responses users need to inspect (in these studies, 50 LLM responses versus 9, the former of which is more firmly at the mesoscale text analysis) or something about the task itself. In the next section we dig into the variable utility of different features to support this task.

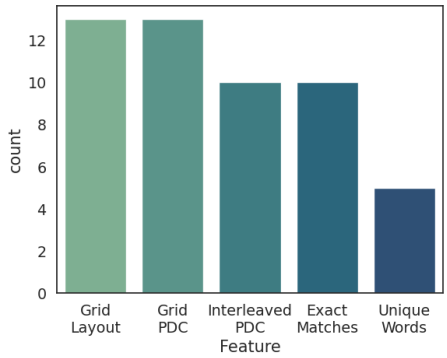
5.7.3 Utility of Different Features. Figure 8 shows the time participants spent with each feature, as well as how many participants indicated a given feature was useful. Grid PDC was the most used and most highly preferred feature. Participants found that Grid PDC help them read responses faster: *"If I saw a similar sentence highlighted then I didn't read the sentence completely, I knew that I already heard them before"* (P1). Additionally, this feature helped people identify response segments; P5 said that *"in the [baseline], I had to actually segment everything in my head after reading it. And in the [exploratory], I didn't have to spend much time reading ... because the segmentation was already being performed."* While we expected participants would use Grid PDC to notice similarities based on what was highlighted, we didn't expect participants to also consider what wasn't highlighted, as P7 did, who said *"if it's not color coded at all then it's probably a unique thing."*

Interleaved PDC was the second most preferred feature, along with Exact Matches. Interleaved PDC was mainly helpful in detecting content differences between the two models. P1 described that *"there were certain clusters which were only present in one of the models."* Similarly, participants noted that singleton groups indicated that only one model came up with that suggestion. P15 also used Interleaved PDC to detect structural differences between the two models: *"It would help me to find out that there's some major difference in the distribution, like GPT3.5 had occupied a bunch of them at the top. ... I found out that it actually shows more introductory sentences."*

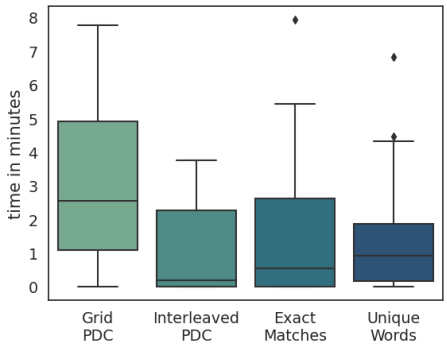
All participants used Exact Matches in very similar ways: to determine how consistent or diverse a model was. Still, this was very useful, and participants spent a decent amount of time in this feature. Unique Words was the least preferred feature, though participants did spend time with it. No participants mentioned in the interview a preference for Unique Words or how they used it.

6 CASE STUDIES

The controlled user study investigated how participants used the features in two different tasks. However, participants had no control over the prompts, models, or what they were attempting to do. This section reports on eight case studies, where we recruited participants via our professional networks who were interested in understanding and examining LLM responses in the context of their own work. Participants were asked to bring their own tasks, and attempt those tasks using the exploratory interface. Table 2 shows for each case study the domain of the task, a short description of the task(s) performed, as well as the maximum number of responses viewed at any one time and the approximate number of words per response. We attempted to recruit a wide range of case studies, in terms of the kinds of tasks but also the anticipated number and length of responses inspected.



(a) How many participants marked a given feature as useful in the post-task survey. Here 'Grid Layout' refers to the general laying out of responses in a user-defined grid.



(b) Time spent in each feature across all participants, ordered by how many participants marked a given feature as useful in the post-task survey, i.e., counts from Figure 8a.

Figure 8: Figure (a) shows the 'popularity' of features based on how many participants marked that feature as useful in the post-task survey. Figure (b) shows how much time participants spent in each feature. Although participants reported equal preference for Interleaved PDC and Exact Matches, on average they spent more time in Exact Matches.

The case studies were open-ended, with a facilitator (one of the authors) giving participants a tutorial of the system for 10-15 minutes and then the participants interacting with the system in whatever way they were most interested in. Participants were asked to share their screen and to think aloud as they worked. The facilitator asked questions during the study to illuminate the participants' goals, findings, and struggles; the interview guideline is in Appendix D. Case studies lasted between one and two hours.

We report two kinds of results. The first is a set of themes that emerged about the utility of the different features. These themes were determined by two authors watching the case studies, writing descriptive summaries of each, and then collecting and collating themes iteratively. The second is an analysis of how user preferences, task, and the number and length of responses informed differences across the case studies. This analysis was done via repeated conversations with all paper authors, all of whom read the descriptive summaries of the case studies and two of whom had

Table 2: Details of the participants in the case studies.

ID	Domain	Task/s	Max Num. Responses	Approx. Words per Response
P1	creative writing	generating insightful connections; generating character voices	20	500
P2a, P2b	model auditing; intersectional AI	how models treat identity markers; how models understand historical events	39	100
P3	creative writing	story continuation; poetry writing	40	50
P4	journalism	prompt engineering for journalistic feedback	50	250
P5	academic writing	prompt engineering for writing task; improve section outline	20	150
P6	business	business idea generation	50	350
P7	law	identifying trademark confusion	20	200
P8	history	how agency is represented in responses to questions about historical events	100	100

watched the video recordings. During our analysis, we recognized that similar themes identified for the controlled user study (see Table 1) were emerging. This demonstrated the generalizability of these themes.

6.1 Utility of Different Features

In this subsection we go over the features and how they were useful (or not) in different contexts. Features could perform quite differently in different contexts, where the combination of a participant's goal and the way the feature performed on the responses they were inspecting accounted for variation in utility.

6.1.1 Exact Matches. As in the controlled user study, participants used Exact Matches mostly to identify how consistent or diverse a set of responses was (i.e., more exact matches indicated more consistent). However, participants were sometimes unsure if there should be many exact matches or not. For, instance P1 asked the facilitator if there should be so few exact matches. Across most of the tasks there were few exact matches, and the matches tended to be syntactic phrases (such as "The relationship between" or "However, it is important to note") rather than semantic ones.

One exception to the trend of few exact matches came in P8's use case, where they looked at the Falcon7B [3] model's response to a question about the industrial revolution. In this case, there were many sentence-length exact matches. This meant they were able to use the un-highlighted text to identifier outliers in responses. Most participants worked with an OpenAI GPT model, so it may be that some models have less variation—and more exact matches—than others. With P8, for the same prompt there were far less exact matches when prompting GPT4 than Falcon7B.

6.1.2 Unique Words. Several participants used Unique Words as a kind of extractive summary of a response. P1's first task was asking a model to identify how two unrelated concepts might be related, with the intention of getting the model to generate "insightful connections." As soon as he turned on Unique Words he commented that the highlighted words might be considered "intersecting" words between the two concepts, and subsequently used this feature to skim through responses to quickly identify what connection(s) the model had generated. P6, who had a model generate business ideas in response to a prompt about how to support the circular economy, noted that Unique Words allowed her to quickly identify what the generated business idea was about or its general domain (e.g., fashion, or food packaging).

Other participants thought that Unique Words could summarize a response, but found that, as implemented, it did not quite do what they wanted. P7 prompted a model to identify potential ways two trademarks might get confused. She thought that Unique Words could be useful to skim the "argument" of the response, but determined that it didn't really do this. To consider a very different context, P3, after having a model generate a poem in the style of a particular poet, wanted Unique Words to highlight "strong choices" or "interesting phrases" rather than just words that were unique, as unique words were sometimes circumstantially unique rather than unique in a way that related to the task at hand.

6.1.3 Grid PDC. As in the controlled user studies, Grid PDC allowed participants to determine consistency or uniqueness across groups of responses in a way that went beyond exact matches. P1, when having a model take on different character "voices" (such as answering a question as if it were the philosopher Hegel or a secret service MI6 Officer), used Grid PDC to immediately and visually tell whether or not the model had a strong sense of what that character was like. If, for example, the set of responses in the voice of Hegel was highlighted in similar colors, indicating the text had been grouped by the PDC algorithm, P1 knew the model had developed a distinct character voice; in contrast if the colors were more varied, it meant the model had no distinct voice but responded to the prompt quite differently each time. If the model was generating a distinct character voice, P1 would then use the highlights to determine what characteristics were present by looking at what the grouped

sentences were about. In a very different context, P5 was trying to determine if his chosen model was capable of doing the rewriting task he had prompted, where the goal was to have the model rewrite a paper section outline to be more clear and concise. He noted that, outside the case study, when he uses an LLM with a chat interface, he is often trying to determine if the model is incapable of doing the task or if the problem is one of prompt engineering. Grid PDC helped him view at a glance, by looking at around five responses to each of several prompt variations, whether or not the model was capable of the task at hand. P7, looking at ways two trademarks might get confused, used Grid PDC not to skim over semantic variation but syntactic variation. She would focus on a particular cluster and look at the variation across responses, some of which would use different adjectives which could make a big difference when adjudicating trademark confusion.

However, Grid PDC did fail if there was too much variation. P3, looking at story continuation and poetry generation, had nothing highlighted in this view because there was so much variation across the generated responses. Another failure mode was the opposite: so much was highlighted that it was difficult to identify clusters.

6.1.4 Interleaved PDC. Similar to the controlled user study, Interleaved PDC was useful to identify smaller variations within a cluster of similar sentences. P5, who had prompted the model to rewrite an outline to be more clear and concise, talked about this view as "shopping cart" mode, where he could pick the best version of a rewritten sentence. P2b, looking at how a model understood certain historical events and historical writers, particularly enjoyed how this view allowed her to see repeated phrases and syntactic patterns that occurred across responses which, had the model known more about the events or people, should have been more distinct. P6, looking at business idea generation, thought this view was useful for noticing themes, as each cluster showed responses that had a similar business idea.

The main issue with Interleaved PDC was a lack of context. P7, looking at trademark confusion, felt she needed to see sentences in context in order to understand what they really meant. In contrast, P5, asking the model to rewrite an outline, felt context was not quite as necessary, perhaps because the task was list-like and therefore each sentence was not as dependent on the surrounding sentences. P2b, looking at historical events and writers, similarly was unworried by lack of context, perhaps because she was less interested in the utility of a particular response and more interested in reading "across" responses in order to understand generally how a model would respond to certain prompts.

6.2 Differences Across User Attitudes, Tasks, and Number and Length of Responses

6.2.1 User Hesitancy with Many Responses. Several participants were hesitant to look at too many responses at once, generally because they assumed it would be too overwhelming. This was the case with P4, who had been doing prompt engineering via an API; P5, who was familiar with a chat interface and regenerating only a few times to test prompts; and P8, who looked at four responses per prompt and would use automatic analysis for larger numbers of responses.

All of these participants, when the facilitator suggested looking at more responses, immediately saw the value and began to attempt new kinds of tasks. For instance, P5 noted that he could put together a better single response by inspecting and selecting portions of many responses. He said, *"the more the better, because I have more ways to choose from...the more I can explore."* P4 noticed that her prompt failed 1 out of 50 times, which was important as she wanted to incorporate the prompt into a web application where incorrect outputs would break downstream functionality. P6 realized that she could inspect up to 50 business ideas at a time, and then drill down into subsets of ideas around a common theme to identify the most innovative ideas, which would be impossible to do with fewer responses.

Not all participants had this initial hesitancy. Some participants had an existing practice inspecting many responses, such as P3 who often looked at 100s of responses when using an LLM to generate creative writing. Others, such as P1, P2a, and P2b, had wanted to look at many responses at once but had never had an appropriate interface to do so. These participants were interested in model capabilities generally, rather than using a model to do a particular task. It may be that users interested in *understanding* models, rather than using them, may more easily understand the utility of our system, whereas users with specific tasks may need demonstration to see the potential for many responses to be useful.

One participant retained their hesitancy even after experiencing our exploratory interface. P7, looking at trademark confusion, could not imagine looking at more than 20 responses at a time, and thought 10 was an appropriate number. P7 also had the least experience with LLMs of all our participants.

6.2.2 Task Stage and User Preferences Impacted How Many Responses to View at Once. Even participants who wanted to inspect 100s of outputs would sometimes want to only see a few at a time. A quintessential example was P3, working on creative writing, who wanted to go over 100s of outputs but felt seeing them all at once was overwhelming. This is a reflection of the information processing style preferences we saw in the email rewriting task, where some participants wanted to see all the emails at once and others wanted to be able to view just three at a time. While information processing style may be driving user preferences here, it was also the case that some participants wanted to view fewer responses after first having inspected many more. For instance, both P1 and P6 noted that viewing many responses allowed them to notice which subsets of responses were most relevant to their task, and then dig into this subset.

6.2.3 Response Attributes Required Algorithms be More Adaptive. Some participants looked at very short responses, just one or two sentences; P3 even suggested wanting to look at sentence fragments. Others looked at responses with five or more paragraphs, getting up to 500 words per response. P4 wanted to look at outputs that were returned as a JSON array, a very distinct syntactic format we had not considered. As these attributes changed, the algorithms needed to respond to them appropriately. For instance, the PDC algorithm segmented responses based on a sentence parser, which was not appropriate for responses only one or two sentences in length, nor JSON outputs. As responses got longer and the number

of responses got higher, the number of clusters detected by PDC increased, often resulting in too many clusters to highlight in distinct colors. With shorter and smaller numbers of responses, our Unique Words algorithm would often highlight less meaningful words, as the TF-IDF metric had insuffieicnt data for patterns to emerge.

7 DISCUSSION

Overall, we saw that the existing and novel features we designed can support LLM response sensemaking. Our novel algorithm, PDC, and both the Grid and Interleaved renderings, were particularly helpful for a variety of tasks, and often the most popular feature with participants, indicating the value of algorithms and renderings that are designed specifically for LLM responses. As LLMs are increasingly adopted, supporting end-users, system developers, and system examiners in making sense of the stochastic capabilities of LLMs becomes an increasingly important area of study. Given that the features implemented in this work are in line with design implications of Variation Theory and Analogical Learning Theory, the results suggest that there may be further utility of these theories for guiding the design of future systems that help users make sense of data and form mental models from examples.

7.1 Design Implications and Future Work

We report on design implications that arose from our studies. The suggested features were either explicitly mentioned by participants or developed through observation of how participants used and responded to our exploratory interface.

7.1.1 New Algorithms and Renderings. When participants discussed the utility of the features, they often suggested new algorithmic goals or additional renderings that would better support their tasks. For instance, while participants consistently used Exact Matches to determine how consistent or diverse responses were, participants also commented that the highlighted phrases were mostly stylistic or syntactic phrases, rather than semantic or content-heavy ones. In this way, they wanted a different feature, one that did not focus solely on exact matches but rather one that took into account other features in the text. Below we list a variety of new algorithm goals for highlighting words or phrases:

- Select words or phrases unique to a general linguistic corpus (rather than to the set of LLM responses).
- Select phrase-length "fuzzy" matches, rather than exact matches.
- Select only exact matches which contain "content" words, as opposed to phrases that are more stylistic.
- Allow users to select a single PDC group and remove the highlighting of other groups to improve visual focus.
- Show descriptive characteristics of the PDC algorithm output, such as number of sentences per group.
- In the Interleaved PDC view, allow users to see the response from which a given sentence came from.

We also saw a need for applying algorithms over subsets of responses. For instance, instead of highlighting words that are unique to a single response, highlight words that are unique to the set of responses from a single model or prompt. All our algorithms (and future algorithms) could be applied to, for instance, rows or columns, which would allow the algorithms to reflect meaningful variation across user-selected subsets of responses.

7.1.2 Support User-Defined Queries. Although we did not want users to have to pre-determine a specific "lens" (i.e., search term) through which to view the data initially, after interacting with our features many participants wanted to customize the algorithms in some way. For instance, participants wanted to define a phrase on which to search for "fuzzy" matches, or define the part-of-speech an algorithm focused on. Another route would be to let users write their own algorithm, which could then be applied to the responses. In a way, our features represented good "defaults" that let users determine what kind of queries they would like to create or customize.

7.1.3 Support Response "Subsetting". Many users wanted to "drill down" into a subset of the responses, or select or save responses such that they could view just these "good" ones without having to sort through the rest. Interface features to support this kind of dynamic inspection of responses would allow users to move along the sensemaking process, narrowing the responses they are investigating. A very simple version of this is allowing users to collapse or hide columns or rows of responses; more sophisticated versions include letting users mark some responses to "store" and letting them hide all unmarked responses at a later point, or letting them dynamically hide or show individual cells.

7.1.4 Support Explicit Annotation. Future work could use our exploratory interface as the basis for an annotation tool, where annotations could then be exported for analysis or used to subset the data. This would allow users to partake in more structured tasks while retaining the utility of the highlighting features. This could also make response inspectors useful in the 1000+ response scale, where users typically focus on annotation rather than sensemaking, but still could use the support of algorithms and renderings.

7.1.5 Integrate Automated Analysis into Response Inspectors. Some participants noted that they do use automated analysis as part of their sensemaking process. For instance, they may apply sentiment analysis to LLM responses and then inspect responses according to which sentiment bucket they fall in. We see potential for integrating such analysis into response inspectors where, for instance, responses could be colored according to how they are classified by an algorithm, or reordered in the grid according to their classification. We still firmly believe that, in sensemaking processes, users must be able to look at the raw text itself as automated analysis may hide or fail to capture variation that users would be interested in if they knew it was there. But visually incorporating automated analysis could be the best of both worlds: users can make use of automated analyses to direct their attention, but retain closeness to the text.

7.2 Limitations

7.2.1 Limitations based on algorithm and rendering implementations. As we consider our work to be a tech probe to better understand how to support the inspection of many LLM responses at once, the algorithms and renderings could be improved. There were small problems with our implementation. Occasionally Exact

Matches would highlight, e.g., a three-word and four-word phrase in different colors, although the three-word phrase was a subset of the four-word phrase. Sometimes the PDC groups got too large, resulting in it not being clear why two sentences were clustered together. On the rendering front, users noted that some colors in the highlighting were too similar to distinguish. Although these problems decreased the utility of the features, we did not observe a huge impact on our results.

7.2.2 Limitations to our studies, and future directions for study. The controlled user study explored only two tasks. While our case studies attempted to test a wider range of tasks, they could not dive deep into any one task. Future work could more precisely investigate the utility of LLM response inspectors for specific tasks, such as auditing models for harmful content, end-user selection of a preferred model, or prompt engineering for system designers. Another angle would be to investigate particular domains, such as the use of LLMs in legal or medical contexts.

A final angle for future studies would be testing at larger scales, e.g., 1000 responses at once, which is 10 times the upper limit of what case study participants examined for tasks they brought with them to the session. We saw that with less than 10 responses total, users typically can read all responses with little support. Similarly, in the case studies we rarely saw participants express interest in much more than 100 responses, at least all at once. However, we did not test many tasks that focused on outlier detection, like in a business context where preventing harmful or strange outputs from occurring is more important. It may be that there are some outlier detection tasks where many 100s of responses, even 1000s, are necessary. However, it does seem like, once we get to 1000s of responses, users typically engage in formal annotation studies. As mentioned in subsubsection 7.1.4, incorporating formal annotation features into a response inspector could be beneficial.

ACKNOWLEDGMENTS

We appreciate the engagement of all our interviewees, lab study participants, and case study participants, who graciously gave us a view into their needs and provided informative feedback on our designs. This material is based upon work supported by the National Science Foundation under Grants No. IIS-2107391 and IIS-2040880. This material is also based on work that is partially funded by an unrestricted gift from Google.

REFERENCES

[1] Denise E Agosto. 2002. Bounded rationality and satisficing in young people's Web-based decision making. *Journal of the American society for Information Science and Technology* 53, 1 (2002), 16–27.

[2] Gerlese Åkerlind. 2015. From phenomenography to variation theory: A review of the development of the variation theory of learning and implications for pedagogical design in higher education. *HERDSA Review of Higher Education* 2 (2015), 5–26.

[3] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The Falcon Series of Open Language Models. (2023). arXiv:2311.16867 [cs.CL]

[4] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena Glassman. 2023. ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing. arXiv:2309.09128 [cs.HC]

[5] Ian Arawjo, Priyan Vaithilingam, Chelse Swoopes, Martin Wattenberg, and Elena Glassman. 2023. ChainForge. https://www.chainforge.ai/. Accessed: 2023-07-21.

[6] Daniel Buschek, Martin Zürn, and Malin Eiband. 2021. The Impact of Multiple Parallel Phrase Suggestions on Email Input and Composition Behaviour of Native and Non-Native English Writers. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 732, 13 pages. https://doi.org/10.1145/3411764.3445372

[7] Joseph Chee Chang, Nathan Hahn, and Aniket Kittur. 2020. Mesh: Scaffolding Comparison Tables for Online Decision Making. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 391–405. https://doi.org/10.1145/3379337.3415865

[8] M. Correll, M. Witmore, and M. Gleicher. 2011. Exploring Collections of Tagged Text for Literary Scholarship. *Computer Graphics Forum* 30, 3, 731–740. https://doi.org/10.1111/j.1467-8659.2011.01922.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2011.01922.x

[9] R Haentjens Dekker and Gregor Middell. 2011. Computer-supported collation with CollateX: managing textual variance in an environment with varying requirements. In *Supporting Digital Humanities 2011: Answering the unaskable*.

[10] Nicholas Diakopoulos, Dag Elgesem, Andrew Salway, Amy Zhang, and Knut Hofland. 2015. Compare clouds: Visualizing text corpora to compare media frames. In *Proceedings of IUI Workshop on Visual Text Analytics*. Citeseer, 193–202.

[11] Geoffrey B Duggan and Stephen J Payne. 2009. Text skimming: The process and effectiveness of foraging through text under time pressure. *Journal of experimental psychology: Applied* 15, 3 (2009), 228.

[12] Geoffrey B Duggan and Stephen J Payne. 2011. Skim reading by satisficing: evidence from eye tracking. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1141–1150.

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

[14] Dedre Gentner and Linsey A Smith. 2013. Analogical learning and reasoning. *The Oxford handbook of cognitive psychology* (2013), 668–681.

[15] Katy Ilonka Gero, Jonathan K. Kummerfeld, and Elena L. Glassman. 2022. Sensemaking Interfaces for Human Evaluation of Language Model Outputs. In *NeurIPS: In Workshop on Human Evaluation of Generative Models*.

[16] 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 Trans. Comput.-Hum. Interact.* 22, 2, Article 7 (mar 2015), 35 pages. https://doi.org/10.1145/2699751

[17] Elena L. Glassman, Tianyi Zhang, Björn Hartmann, and Miryung Kim. 2018. Visualizing API Usage Examples at Scale. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (<conf-loc>, <city>Montreal QC</city>, <country>Canada</country>, </conf-loc>) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174154

[18] Steven M. Goodman, Erin Buehler, Patrick Clary, Andy Coenen, Aaron Donsbach, Tiffanie N. Horne, Michal Lahav, Robert MacDonald, Rain Breaw Michaels, Ajit Narayanan, Mahima Pushkarna, Joel Riley, Alex Santana, Lei Shi, Rachel Sweeney, Phil Weaver, Ann Yuan, and Meredith Ringel Morris. 2022. LaMPost: Design and Evaluation of an AI-Assisted Email Writing Prototype for Adults with Dyslexia. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility* (Athens, Greece) *(ASSETS '22)*. Association for Computing Machinery, New York, NY, USA, Article 24, 18 pages. https://doi.org/10.1145/3517428.3544819

[19] Ronald Haentjens Dekker, Dirk Van Hulle, Gregor Middell, Vincent Neyt, and Joris Van Zundert. 2015. Computer-supported collation of modern manuscripts: CollateX and the Beckett Digital Manuscript Project. *Digital Scholarship in the Humanities* 30, 3 (2015), 452–470.

[20] Uta Hinrichs, Stefania Forlini, and Bridget Moynihan. 2015. Speculative practices: Utilizing infovis to explore untapped literary collections. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 429–438.

[21] Mengdie Hu, Krist Wongsuphasawat, and John Stasko. 2016. Visualizing social media content with sententree. *IEEE transactions on visualization and computer graphics* 23, 1 (2016), 621–630.

[22] Hilary Hutchinson, Wendy Mackay, Bo Westerlund, Benjamin B Bederson, Allison Druin, Catherine Plaisant, Michel Beaudouin-Lafon, Stéphane Conversy, Helen Evans, Heiko Hansen, et al. 2003. Technology probes: inspiring design for and with families. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 17–24.

[23] Peiling Jiang, Jude Rayan, Steven P Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. *arXiv preprint arXiv:2305.11473* (2023).

[24] Daniel Jurafsky and James H. Martin. 2023. Question Answering and Information Retrieval. In *Speech and Language Processing*.

[25] Minsuk Kahng, Dezhi Fang, and Duen Horng Chau. 2016. Visual exploration of machine learning results using data cube analysis. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. 1–6.

[26] Panayiota Kendeou and Paul Van Den Broek. 2007. The effects of prior knowledge and text structure on comprehension processes during reading of scientific texts. *Memory & cognition* 35, 7 (2007), 1567–1577.

[27] 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.
[28] Steffen Koch, Markus John, Michael Wörner, Andreas Müller, and Thomas Ertl. 2014. VarifocalReader—in-depth visual analysis of large text documents. *IEEE transactions on visualization and computer graphics* 20, 12 (2014), 1723–1732.
[29] Gunther Kress and Theo Van Leeuwen. 2020. *Reading images: The grammar of visual design.* Routledge.
[30] Krtxoe. 2023. *What do you all actually use chatGPT for?* https://www.reddit.com/r/ChatGPT/comments/133mc2v/what_do_you_all_actually_use_chatgpt_for/ Accessed: 2023-09-14.
[31] Tuan Le and Leman Akoglu. 2019. ContraVis: contrastive and visual topic modeling for comparing document collections. In *The World Wide Web Conference*. 928–938.
[32] Sara Leckner. 2012. Presentation factors affecting reading behaviour in readers of newspaper media: an eye-tracking perspective. *Visual Communication* 11, 2 (2012), 163–184.
[33] Lego_Hippo. 2023. *How are you using ChatGPT to write emails?* https://www.reddit.com/r/sales/comments/10s24wy/how_are_you_using_chatgpt_to_write_emails/ Accessed: 2023-09-14.
[34] Vivian Liu and Lydia B Chilton. 2022. Design Guidelines for Prompt Engineering Text-to-Image Generative Models. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–23. https://doi.org/10.1145/3491102.3501825
[35] Ference Marton. 2014. *Necessary conditions of learning.* Routledge.
[36] Frank Meng, Craig A Morioka, and Suzie El-Saden. 2011. Determining word sequence variation patterns in clinical documents using multiple sequence alignment. In *AMIA Annual Symposium Proceedings*, Vol. 2011. American Medical Informatics Association, 934.
[37] Bonnie JF Meyer. 1999. Importance of text structure in everyday reading. In *Understanding language understanding: Computational models of reading*. 227–252.
[38] Aditi Muralidharan, Marti A. Hearst, and Christopher Fan. 2013. WordSeer: A Knowledge Synthesis Environment for Textual Data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management* (San Francisco, California, USA) *(CIKM '13)*. Association for Computing Machinery, New York, NY, USA, 2533–2536. https://doi.org/10.1145/2505515.2508212
[39] Thierry Olive and Marie-Laure Barbier. 2017. Processing time and cognitive effort of longhand note taking when reading and summarizing a structured or linear text. *Written Communication* 34, 2 (2017), 224–246.
[40] K Pernice, K Whitenton, J Nielsen, et al. 2014. How People Read Online: The Eyetracking Evidence. *Fremont, USA: Nielsen Norman Group* (2014).
[41] 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.
[42] promptfoo. Accessed 2023. Test your prompts. https://www.promptfoo.dev/.
[43] Nancy Staggers and Anthony F. Norcio. 1993. Mental models: concepts for human-computer interaction research. *International Journal of Man-machine studies* 38, 4 (1993), 587–605.
[44] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. *arXiv preprint arXiv:2305.11483* (2023).
[45] David R. Thomas. 2006. A General Inductive Approach for Analyzing Qualitative Evaluation Data. *American Journal of Evaluation* 27, 2 (June 2006), 237–246. https://doi.org/10.1177/1098214005283748
[46] Weights and Biases. Accessed 2023. Weights and Biases Docs: Prompts for LLMs. https://docs.wandb.ai/guides/prompts.
[47] wink js. Accessed 2023. Wink NLP. https://winkjs.org/wink-nlp/.
[48] Maryanne Wolf and Catherine J Stoodley. 2008. *Proust and the squid: The story and science of the reading brain.* Harper Perennial New York.
[49] Patricia Wright. 1999. The psychology of layout: Consequences of the visual structure of documents. *American Association for Artificial Intelligence Technical Report FS-99-04* (1999), 1–9.
[50] Tongshuang Wu, Kanit Wongsuphasawat, Donghao Ren, Kayur Patel, and Chris DuBois. 2020. Tempura: Query analysis with structural templates. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.
[51] Litao Yan, Miryung Kim, Bjoern Hartmann, Tianyi Zhang, and Elena L. Glassman. 2022. Concept-Annotated Examples for Library Comparison. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (Bend, OR, USA) *(UIST '22)*. Association for Computing Machinery, New York, NY, USA, Article 65, 16 pages. https://doi.org/10.1145/3526113.3545647
[52] 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.
[53] Tariq Yousef and Stefan Janicke. 2020. A survey of text alignment visualization. *IEEE transactions on visualization and computer graphics* 27, 2 (2020), 1149–1159.

A FORMATIVE INTERVIEW GUIDELINE

The interview guideline was meant to be extremely broad, given the range of kinds of LLM-based tasks participants were engaged in. Here we list all questions, however many were not relevant for certain participants, and many custom follow-up questions were asked depending on the context in which the participant was working.

- How do you select models or design prompts?
- Do benchmarks work for you? Do you use metrics? Trial and error? To what extent does quantitative evaluation or benchmarks work for your use case?
- Do you ever regenerate or compare outputs? What about between prompts or models?
- Have you noticed models changing over time? Have you tried different models?
- How do you know your system is working? What's success for you?
- How often do you update models / prompts?
- What might help qualitative evaluation?
- What does qualitative evaluation look like? Are you comparing models, prompts, other things? Do you rely on annotations, having people read outputs and discuss, or other methods?
- Does any evaluation involve comparing multiple outputs in a way other than via annotation? Do you ever look for qualitative differences between outputs? e.g., "This prompt produces more verbose and flowery outputs."
- When you have people manually inspect outputs, how many outputs does any one person typically look at at a time?

B FEATURES IMPLEMENTED

B.1 Exact Matches

Here we describe in more detail the exact algorithm used to detect exact matches:

We first look for the longest common substring between all possible pairings of responses. We split any substrings that cross sentence boundaries, as this often resulted in exact matches that didn't represent meaningful text segments. We remove substrings with fewer than three words, as we want to prioritize text segments rather than single words, though this is a variable that could be tuned. All substring matches are then matched to each other across all responses, not just the initial pairing they were originally derived from, such that for each substring we know how many responses in the whole collection it occurs in. Substrings are sorted based on a weighted function of a) how many responses that substring occurs in and b) how long the substring is. For the study, we give length of substring a weight of .75 and number of responses it occurs in a weight of 1.[8] Finally, the top k substrings are returned, where k is the minimum of 12 and half the number of all responses in the collection. The value 12 is selected as a maximum value based on the fact that we intended to visualize each match in a different color, and we had a palette of 12 colors which we thought to be visually distinct.

B.2 Positional Diction Clustering (PDC)

B.2.1 Definition of content similarity. Our score for content similarity is $(|XinY| + |YinX|)/(|X| + |Y|)$, where X and Y are the two sentences,

[8] Future work could look at how to balance or side-step the tension this algorithm creates between identifying shorter matching text segments that are prevalent across many LLM outputs and longer matching text segments that only occur in a smaller subset of LLM outputs.

and $|XinY|$ is the count of words in X that appear in Y. This is the same as Bray–Curtis Similarity except that our numerator is the sum of counts rather than two times the min of counts.

B.2.2 Details of clustering. We use a form of single-linkage agglomerative clustering to form groups of sentences across all responses. First, we split all responses into individual sentences and calculate similarity as described in Section 4.3.3. We start with each sentence in its own group. Then, we iterate through all pairs of sentences, sorted by their text similarity, and if a metric that combines text similarity and position similarity is higher than a threshold, we consider combining their groups. The metric is a linear combination of the two similarities, with a weight of 1.5 on text similarity and a weight of 1 on position similarity, and the threshold is 1.2. When considering combining groups, we only merge them if at least 70% of the sentences are from distinct responses. The groups are ordered by their median normalized location in responses, with ties broken to put sentences from longer documents first.

B.2.3 Notes on choices.

- *Measuring content similarity with exact diction overlap.* This metric is chosen because it maximises the graying out which is used in the second rendering, Interleaved Layout. That, in turn, helps users notice what is similar and different across sentences within the same group.
- *Using both content and location similarity in the grouping algorithm.* This ensures that the groups correspond to parts of the emergent templates in responses. Location similarity alone would lead to groups with no coherent meaning. Content similarity alone could lead to groups with sentences from the start of one response and the end of another.[9]
- *Only merging groups if the new group has sentences from a range of responses.* This helps form groups that represent emergent patterns across responses, rather than patterns within individual responses.

C USER STUDY

C.1 Procedure Details

Participants were walked through informed consent, and then audio and screen recording began. Participants then filled out a short demographic survey, including questions about their exposure to LLMs. Participants then went through two different tasks—the email rewriting task and the model comparison task—doing each task twice, once in each condition. For the first task, before each interface condition, participants were shown a short tutorial introducing the interface's features.

Within a given task, e.g., email rewriting, the order of the scenarios remained the same and *the interface conditions were counterbalanced.* After completing the task in each scenario, participants filled out a short survey, which included close-ended questions about cognitive load, how realistic the task was, and how well they believed they had performed on the task. After each task, the facilitator conducted a short semi-structured interview with open-ended questions about the utility of the features in both interfaces. At the end of the study, the facilitator stopped the recording, allowed participants to ask questions about the study, and conferred study payment. See Appendix subsection C.6 for survey and interview questions.

C.2 Determining Time on Task and Time in Feature

We used the study video recordings to manually determine the amount of time each participant required to finish each task, as well as to identify if participants manually edited their email or used copy/paste in task one, and note if participants used the keyboard search function (command+F) when evaluating model differences in task two. We also used the video recordings to manually log the total time each participant spent in each of the five system features while completing task one and task two. Members of the research team evaluated the timing data for accuracy.

[9]For such a group, it would be unclear where to show it in the Interleaved Layout rendering. For instance, the mean location, the middle, would not be faithful to the source location for any of the sentences. The median location would either be the start or end, which would not be faithful to half of the sentences.

C.3 Baseline Interface

Figure 9 shows the baseline interface using the same example prompt (writing a short story for a child about a creature) found in subsection 4.3.

C.4 Participant Details

Figure 10 shows participant responses to the pre-study survey questions.

C.5 Task Prompts

C.5.1 Email Re-writing.

- **Task 1A**:
  `I'm writing an email to my professor asking for a letter of recommendation. Could you rewrite it to be more style? Make sure it's less than 100 words.`

  `Hi Prof. Sandy,`
  `I took your intro to algorithms class last fall and got an A. I really liked the class and thought you were a great teacher. The class was super hard; can't believe I got an A, haha! I am applying to a summer internship as a software engineer and was wondering if I could put you down as a reference. If you can't do it, no worries! Thanks,`
  `[your name]`

- **Task 1B**:
  `I'm writing an email to the person who will be my manager for my summer internship. I already have the job offer, but I want to ask to start two weeks later than suggested. Can you rewrite it to be more style? Make sure it's less than 100 words.`

  `Hi Sandra,`
  `I'm really excited about my summer internship with your team. I am already thinking of project ideas! I know that you said the internship start date for all interns is May 1st. But could I start two weeks later than that? Obviously then I would end two weeks later as well. Totally get it if this isn't possible, but it would really help me out to make this change. Thanks,`
  `[your name]`

C.5.2 Model Comparison.

- **Task 2A**:
  `There's a general audience nonfiction book I'd like to read about neuroscience. I won't have time to read it all. What are some ways I can read just part of it, or skim the book, to get the most out of the book and my time? Keep your response to less than 100 words.`

- **Task 2B:**
  `I have a week until an important final exam. I need to study a lot. What do you recommend I do to make sure I perform at my best? Keep your response to less than 100 words.`

C.6 Pre-Study Survey

(1) What is your participant ID?
(Open-ended response, ID given to participant by researcher)

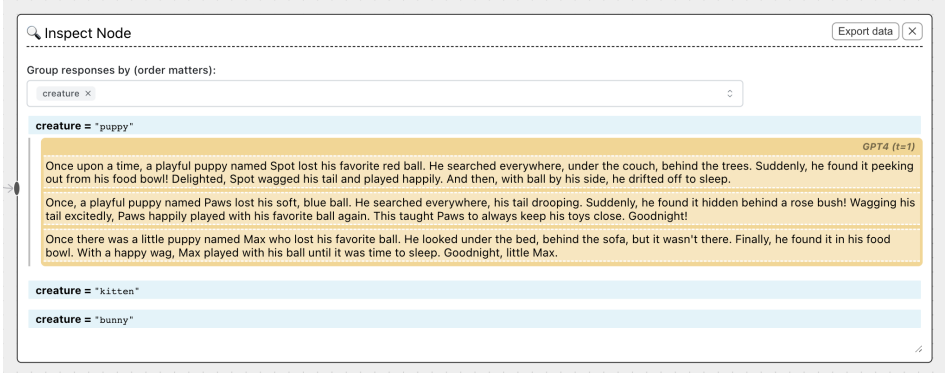


Figure 9: The baseline interface allows users to collapse or 'hide' groups of responses, such that all or just some groups of responses can be seen at once. Responses can be grouped by model or prompt variation. The example here uses the same example from Figure 1: "Write a short story for a five year old child about a {creature} that loses something and then finds it again."

(2) What is your age?
○ 18-24 ○ 25-34 ○ 35-44 ○ 45-54 ○ 54+
(3) What is your gender?
○ Woman ○ Man ○ Non-binary ○ Prefer not to disclose ○ Other:
(4) What kind of student are you?
○ Undergraduate ○ Masters ○ PhD ○ Other:
(5) What is your field of study?
(Open-ended response)
(6) How confident do you feel when writing important emails? For example, asking for an extension on a project.
(Answered on a seven point Likert scale from "Not very confident" to "Very confident")
(7) How much do you know about large language models or chatbots like ChatGPT?
(Answered on a seven point Likert scale from "I have not heard of these things" to "I feel like I am an expert on these models")
(8) How much have you investigated the differences between different language models, for instance the differences between GPT-3.5 and GPT-4?
(Answered on a seven point Likert scale from "I have never investigated this" to "I have investigated the differences extensively")
(9) How often do you use large language models or chatbots like ChatGPT?
○ Never ○ I've used them a few times, but not regularly ○ A few times a month ○ A few times a week ○ Once a day or more
(10) How often do you use large language models or chatbots like ChatGPT for writing? (That is, not for coding or searching for information.)
○ Never ○ I've used them a few times, but not regularly ○ A few times a month ○ A few times a week ○ Once a day or more

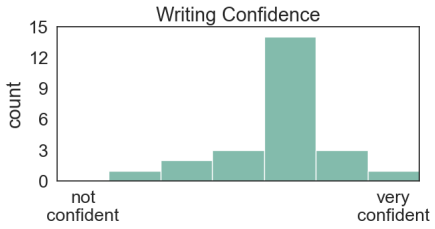
C.7 Post-Task Interview

After completing the second version of each task, participants received the following questions in a post-task interview:

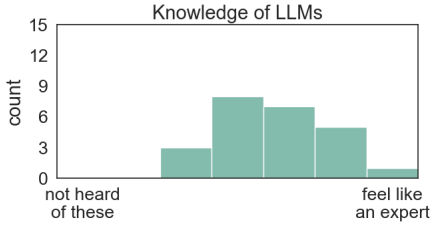
(1) What was your approach toward doing this task?
○ Was it different in different interfaces (linear v. grid)?
(2) Did the linear or grid view make it easier to find the best output? (Why?)
(3) Was the linear or grid view more overwhelming? (Why?)
(4) Were the grid highlighting or grouping features useful? (Why or why not?)
(5) Is there anything that I didn't ask about that you want to share?

C.8 Email Rewriting Survey

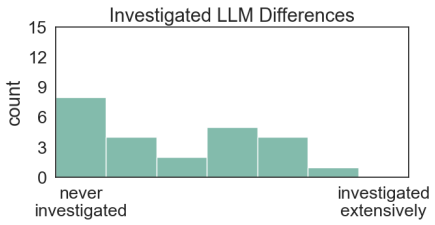
(1) What is your participant ID? (Open-ended response, ID given to participant by researcher)
(2) Which email rewriting task did you just do?
○ Asking for a reference letter ○ Requesting a later start date
(3) Which interface did you have?
○ Linear inspect node ○ Grid inspect node
(4) How mentally demanding was the task?
(Answered on a seven point Likert scale from "Very low" to "Very high")
(5) How hurried or rushed was the pace of the task?
(Answered on a seven point Likert scale from "Very low" to "Very high")
(6) How mentally demanding was the task?
(Answered on a seven point Likert scale from "Very low" to "Very high")
(7) How successful were you in accomplishing what you were asked to do?
(Answered on a seven point Likert scale from "Very low" to "Very high")
(8) How hard did you have to work to accomplish your level of performance?
(Answered on a seven point Likert scale from "Very low" to "Very high")
(9) How insecure, discouraged, irritated, stressed, and annoyed were you?
(Answered on a seven point Likert scale from "Very low" to "Very high")
(10) How realistic was the email writing task?
(Answered on a seven point Likert scale from "I have never had to write an email for that purpose" to "I have written emails for that purpose in the past")
(11) How close was your **selected response** to something you would send?
(Answered on a seven point Likert scale from "I would never send that response" to "I would definitely send that response")
(12) How close was your **edited response** to something you would send?



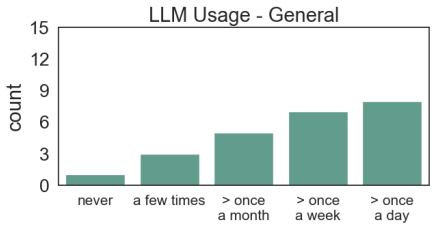
(a) "How confident do you feel when writing important emails? For example, asking for an extension on a project."



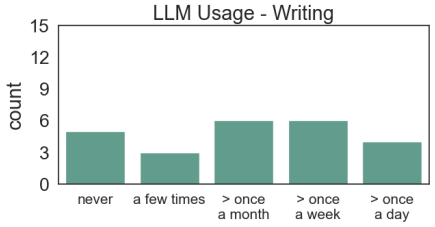
(b) "How much do you know about large language models or chatbots like ChatGPT?"



(c) "How much have you investigated the differences between different language models, for instance the differences between GPT-3.5 and GPT-4?"



(d) "How often do you use large language models or chatbots like ChatGPT?"



(e) "How often do you use large language models or chatbots like ChatGPT for writing? (That is, not for coding or searching for information.)"

Figure 10: Participant responses to pre-study questions about experience with writing and LLMs.

(Answered on a seven point Likert scale from "I would never send that response" to "I would definitely send that response")

After completing the second task, participants completed this Email Re-writing Survey again and received two additional questions:

(13) Which interface made the task **easier**?
(Answered on a seven point Likert scale from "linear inspect node" to "grid inspect node")

(14) Which interface felt more overwhelming?
(Answered on a seven point Likert scale from "linear inspect node" to "grid inspect node")

C.9 Model Comparison Survey

(1) What is your participant ID? (Open-ended response, ID given to participant by researcher)

(2) Which model comparison task did you just do?
- ○ Advice on how to skim read a book
- ○ Advice on how to prepare for finals

(3) Which interface did you have?
- ○ Linear inspect node
- ○ Grid inspect node

(4) How realistic was the advice topic?
(Answered on a seven point Likert scale from "I have never asked for similar advice before" to "I have asked for similar advice before")

(5) How mentally demanding was the task?
(Answered on a seven point Likert scale from "Very low" to "Very high")

(6) How hurried or rushed was the pace of the task?
(Answered on a seven point Likert scale from "Very low" to "Very high")

(7) How successful were you in accomplishing what you were asked to do?
(Answered on a seven point Likert scale from "Very low" to "Very high")

(8) How hard did you have to work to accomplish your level of performance?
(Answered on a seven point Likert scale from "Very low" to "Very high")

(9) How insecure, discouraged, irritated, stressed, and annoyed were you?
(Answered on a seven point Likert scale from "Very low" to "Very high")

(10) Thinking about the **differences between models**, how well were you able to determine the differences?
(Answered on a seven point Likert scale from "I wasn't able to determine many differences" to "I was able to determine most or all of the differences")

(11) Which features were most helpful to detecting **model differences**?
□ Grid layout □ Exact Matches □ Unique Words □ Similar Sentences □ Groupings View

After completing the second task, participants completed this Model Comparison Survey again and received two additional questions:

(12) Which interface made the task easier?
(Answered on a seven point Likert scale from "linear inspect node" to "grid inspect node")

(13) Which interface felt more overwhelming?
(Answered on a seven point Likert scale from "linear inspect node" to "grid inspect node")

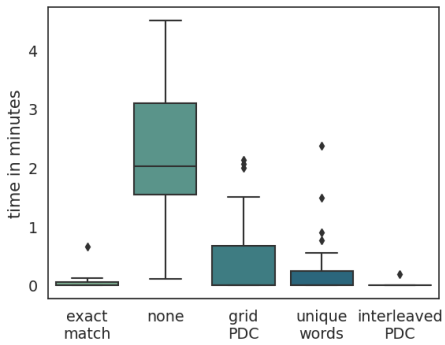
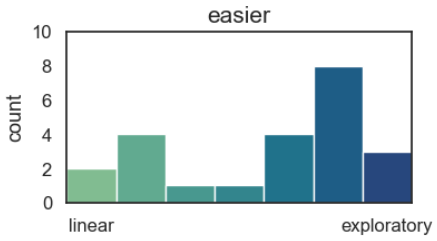
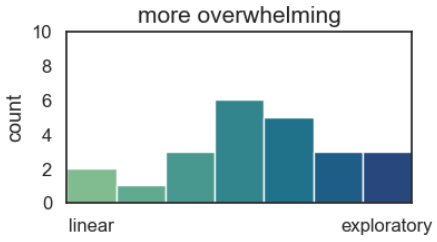


Figure 11: Boxplot of how much time participants spent in each feature when using the exploratory interface. Here 'none' refers to being in the grid interface with no highlighting features activated. Note that time spent in a feature does not necessarily indicate that a participant found the feature useful.



(a) "Which interface made the task easier?"



(b) "Which interface was more overwhelming?"

Figure 12: Participant interface preferences for the email rewriting task.

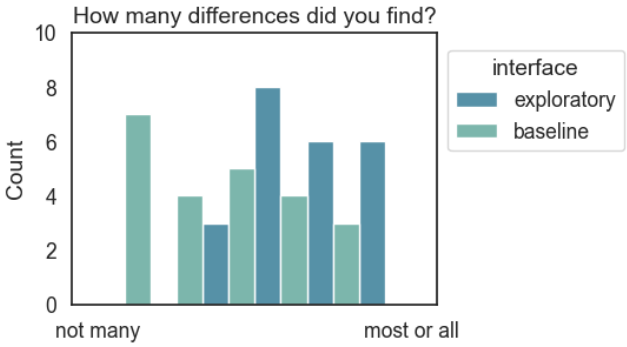


Figure 13: Participant responses to the question "Thinking about the differences between models, how well were you able to determine the differences?"

C.10 Email Rewriting Task Results

C.10.1 Learning Effects: Participants Were Faster the Second Time Around. Participants found both email rewriting tasks to be quite realistic. Analyzing their responses to 7-point Likert scale survey questions with a two-tailed Mann-Whitney U test, participants felt significantly more "hurried or rushed" in task A than task B ($p < .01$). Analyzing their time on task with a two-tailed t-test, split into the 'select' portion of the task and the 'edit' position of the task, we found that participants also took significantly more time on task A (select portion: $p < .05$, edit portion: $p < .01$). This aligns with their survey responses; participants took more time during task A and felt more rushed. For this reason, we split our analysis comparing interface conditions between task A and task B.

C.11 Model Differences Task Results

Figure 13 shows participant responses to the question "How many differences did you find?": a self-reported measure of success in the task. We see that participants reported finding more differences with the exploratory interface than the baseline interface.

D INTERVIEW GUIDELINE FOR CASE STUDY

- Explain consent details (recording, how data will be aggregated, anonymized, and used for research purposes, their ability to opt out, and their right to request a copy of the paper) and verbally request for their consent to record. Begin recording via Zoom.
- Demo interface
- Request for participant to open the interface and share their screen
- Send API keys; get them running on the interface
- Ask them to think aloud through their process; guide them to more interesting prompts if necessary; ask them what they are thinking about or seeing. Encourage them to try out all the features. Explain how features work when requested.
- Sample interview questions:
  - What kinds of tasks or information do you want to look at? Compare prompts, models, prompt variables? Look at long tail distribution, or set of standard responses?
  - Observations: What feature do they try first? Second? Ask why they tried particular prompts.
  - Does this tool support your exploration and inspection of outputs?
  - What works? What is missing?
  - How would you complete the selected task without this tool?
  - How does this experience differ from your prior interaction with LLMs?
  - The goal is to aid in skimming and comparison of responses: does it do that?