Towards Dataset-scale and Feature-oriented Evaluation of Text Summarization in Large Language Model Prompts

Sam Yu-Te Lee (iD), Aryaman Bahukhandi (iD), Dongyu Liu (iD) and Kwan-Liu Ma (iD)

Abstract— Recent advancements in Large Language Models (LLMs) and Prompt Engineering have made chatbot customization more accessible, significantly reducing barriers to tasks that previously required programming skills. However, prompt evaluation, especially at the dataset scale, remains complex due to the need to assess prompts across thousands of test instances within a dataset. Our study, based on a comprehensive literature review and pilot study, summarized five critical challenges in prompt evaluation. In response, we introduce a feature-oriented workflow for systematic prompt evaluation. In the context of text summarization, our workflow advocates evaluation with summary characteristics (feature metrics) such as complexity, formality, or naturalness, instead of using traditional quality metrics like ROUGE. This design choice enables a more user-friendly evaluation of prompts, as it guides users in sorting through the ambiguity inherent in natural language. To support this workflow, we introduce Awesum, a visual analytics system that facilitates identifying optimal prompt refinements for text summarization through interactive visualizations, featuring a novel Prompt Comparator design that employs a BubbleSet-inspired design enhanced by dimensionality reduction techniques. We evaluate the effectiveness and general applicability of the system with practitioners from various domains and found that (1) our design helps overcome the learning curve for non-technical people to conduct a systematic evaluation of summarization prompts, and (2) our feature-oriented workflow has the potential to generalize to other NLG and image-generation tasks. For future works, we advocate moving towards feature-oriented evaluation of LLM prompts and discuss unsolved challenges in terms of human-agent interaction.

Index Terms—Visual analytics, prompt engineering, text summarization, human-computer interaction, dimensionality reduction

✦

1 INTRODUCTION

Prompting is a new way of biasing Large Language Models (LLMs) towards the desired output with natural language instructions [9]. Recently, OpenAI's GPT Store opened the door for non-technical people to customize chatbots with prompts. The low technical barriers and high customizability of prompting democratize a wide range of tasks that previously required programming skills [18], e.g., personalized reading and writing assistants [37] or visualization creation [16, 52, 59]. On the other hand, composing a desired prompt (i.e., prompt engineering) is non-trivial. Design studies in prompt engineering [18, 20, 61] have pointed out that *evaluation* in prompt engineering remains arduous because of five challenges in current evaluation practices, i.e., evaluation is *Opportunistic, Manual, Multi-criteria, Dynamic, and Unactionable*.

These challenges exist when the evaluation is done on a few test instances, and are amplified as the evaluation scales up. For example, a prompt designed for text summarization on a news article dataset with one thousand articles requires an evaluation on a larger test set than just a few instances to ensure its robustness. We refer to this kind of prompt evaluation as "dataset scale", which is under-explored yet challenging. In contrast to machine learning evaluation where quantitative metrics like F1 score are typically used, the quality of a summary is hard to capture with quantitative metrics. Traditional metrics such as ROUGE [25] have been criticized in many ways [10], especially their inability to differentiate between state-of-the-art models, thus not suitable for evaluating summaries generated by LLM prompts. As a consequence, people choose the test set *opportunisticly*, e.g., whatever they see first or the data points that seem easy to evaluate, instead of choosing rigorous representatives of the whole dataset. Moreover, the test set is evaluated by *manually* scanning through the outputs. This introduces *multiple and dynamic criteria* in prompt evaluation, where people change evaluation criteria as they see fit after the scanning. Most importantly, this evaluation approach does not necessarily lead

- Sam Yu-Te Lee, Aryaman Bahukhandi, Dongyu Liu and Kwan-Liu Ma are with University of California, Davis. Email: ytlee, abahukhandi, dyuliu, klma@ucdavis.edu

to *actionable insights*, i.e., what kind of instruction is needed, or how should the criteria be expressed to generate the desired output, leading to a highly unpredictable trial-and-error prompt refinement process.

In this work, we target text summarization prompt refinement and attempt to explore the tasks that are involved in a systematic evaluation for dataset-scale prompts and the visualization designs that foster actionable insights. We target individuals who seek to tailor chatbots for their specific needs or work requirements using prompts and aspire to efficiently perform systematic evaluations of their prompts. Through the lens of text summarization, we seek to generalize our findings to broader real-world tasks that require dataset-scale evaluation. While text summarization may not be representative of all possible prompting tasks that LLMs could perform, it presents challenges that are not straightforward to solve and studying them could provide insights towards more generalizable solutions. First, summarization is extensively studied yet NLP researchers can not agree on a robust **quality metric** (metrics that seek to quantify the quality of a summary, e.g., ROUGE) for evaluating state-of-the-art systems. This indicates that quality metrics might have reached their limits in differentiating nuanced quality differences. Second, a significant cognitive load is required to manually evaluate summaries. Prompt designers need to read the lengthy text and the summarized text to decide the prompt performance and refinement directions, making it infeasible at the dataset scale.

Through a pilot study, we found that **feature metrics**, i.e., computational metrics that **characterize** the summary from different facets, are more desirable than quality metrics for both technical and non-technical prompt designers. For example, formality and complexity might be two critical characteristics (features) of a summary if the summarization goal is to produce academic-level writing. The key distinction of feature metrics is that each metric evaluates one characteristic of the generated summary, and the desired summary does not necessarily have a higher score, e.g., a complexity score suitable for kids (lower scores) might be more desirable than for professionals (higher scores). Compared to quality metrics, feature metrics provide a multi-faceted understanding of the generated summaries and a more concrete refinement direction.

Based on this finding, we introduce a feature-oriented workflow that involves four tasks: *Feature Selection, Example Sourcing, Prompt Refinement, and Evaluation for Refinement*. We develop *Awesum*, a VA system that implements the workflow for text summarization, incorporating computational linguistics metrics as features and intelligent agents to support the tasks. It uses cluster visualization to provide an overview of the dataset and support the identification of ideal examples.

Levels of Explanations for feature metrics (complexity)			Levels of Prompt Editing Support	
L1: General Description Complexity is the effort a reader needs to understand a text based on its content (vocabulary and syntax). Lower complexity in a text eases reading effort and speed for the general population of readers. Every writer has a target audience that they have to adjust their complexity levels to.	**L2: Computational Linguistics** Complexity metrics aim to quantify the complexity of a piece of writing by considering various linguistic features, such as sentence length, syllable count, familiarity and commonality or semantic difficulty.	**L3: Computational Metric** The Flesch Reading Ease Index is a readability formula that considers Average Sentence Length and Average Word Syllables. `Formula:` $206.835 - 1.105(\frac{total\ words}{total\ sentence}) - 84.6(\frac{total\ syllables}{total\ words})$ Complexity is thus the inverse of readability. Higher scores means the text is very difficult and vice versa.	**L1: Prompt Replacement** <u>User prompt</u>: Summarize the article in a readable way <u>Replaced with</u>: The user instructs you to [User prompt]. Strictly follow the user's instruction. **L3: Keyword Replacement** <u>User prompt</u>: [Summarize] the article in a readable way. <u>Replaced with</u>: [Briefly describe] the article in a readable way.	**L2: Sentence Replacement** <u>User prompt</u>: [Summarize the article in a readable way]. <u>Replaced with</u>: Use a tone for kids to summarize the article **L4: Prompt Suggestions** <u>User prompt</u>: Summarize the article in a readable way. <u>Suggestion</u>: Be more specific about what 'readable' means.

Fig. 1: **Left**: Levels of explanations for feature metrics (using complexity for demonstration). L1: a textual description that vaguely defines the complexity of a text. L2: a textual description but in the context of computational linguistics, outlining common linguistic features that are considered. L3: A specific computation metric for complexity, including a textual description and the computation formula. **Right**: Levels of prompt editing support. L1: the system inserts user prompts into a template prompt under the hood. L2: the system suggests a better sentence. L3: the system suggests a better keyword or noun phrase. L4: the system gives suggestions on the prompt based on established prompting methodology.

It provides prompt suggestions based on established prompting methodologies to support prompt refinement. Finally, a scatter plot inspired by BubbleSets and enhanced with dimensional reduction techniques supports users generate actionable insights for prompt refinement. We recruit experts from various domains for practitioner review, from which we confirm the effectiveness of the system. We report the generalizability of the system to a broader range of tasks and discuss insights into human-agent interaction. Our contributions are as follows:

- We introduce a feature-oriented workflow to address challenges in supporting dataset-scale prompt evaluation, which we summarized from a literature review and a pilot study.
- We develop a VA system, *Awesum*, that supports the feature-oriented workflow for text summarization with computational linguistics, intelligent agents, and interactive visualizations.
- We evaluate the effectiveness and generalizability of the system with a case study and interviews with practitioners from various domains, and report implications for future directions in prompt evaluation and human-agent interaction.

2 RELATED WORKS

In this section, we first explore the technical challenges and solutions of text summarization evaluation. We then examine current interactive visual interfaces and design studies for iterative prompt engineering, highlighting the research gap in evaluating dataset-scale prompts.

2.1 Text Summarization Evaluation

Automatic summarization is a classic natural language generation (NLG) task that converts a lengthy source text into a condensed text containing the most important information. However, the evaluation of a summarization system has remained a persistent challenge. In the past, n-gram overlap metrics like BLEU [36] or ROGUE [25] were used to assess the quality of a generated summary. Later, they were enhanced with embeddings-based similarity measurements like BERTScore [62] and MoverScore [64]. Still, studies have shown that they cannot reliably quantify improvements if the difference is too small, because they are not sensitive enough to capture the subtle differences in high-quality summaries that humans can perceive [10]. More recently, these metrics have become ineffective as LLMs demonstrated the capability of generating human-level summaries. Another limitation with these metrics is that they all require a labeled dataset (i.e., references) to work with, essentially transforming the summary quality evaluation into a similarity evaluation, where a human-written summary is assumed to be optimal. This is not practical in prompting as the practitioners can be non-technical and the outcome of prompting can easily exceed in quality that of any human-labeled dataset. Researchers have been calling for new metrics to assess the NLG quality of most recent models [7,34].

While some reference-free metrics that do not require labels are proposed, such as QA-QG metrics [13,41] or LLM-as-evaluators [55,65], they do not yet show a consistent correlation with human judgments [43], might be capturing spurious correlations [12], or might exhibit various biases [50]. We argue that metrics that attempt to capture the overall "quality" of a prompt output (i.e., quality metrics) are susceptible to misaligning with human judgment, as previous studies

have shown that human judgment is multi-criteria and dynamic [20]. As an alternative, we introduce *feature metrics* that characterize the outputs to support sensemaking on the prompt performances. For summarization, we use feature metrics such as formality and naturalness to support dataset-scale prompt evaluation.

2.2 Interactive Visualizations for Model Refinement

Facilitating model refinement with interactive visualizations is an extensive field [15]. Many works help model developers debug their models [27,38,47,57] by visualizing the data flows or training patterns, but they are designed specifically for the underlying model architecture and are not applicable in prompt engineering. Works that focus on model tuning [3,4,14,54] are more similar to the settings in prompt engineering, where visualizations such as line charts or parallel coordinates [14] are used to visualize the performance metrics and their relations with hyperparameter settings.

Still, new visualization techniques are needed in prompt engineering for three reasons. First, previous systems are designed for model developers. In prompt engineering, the target users are extended to non-technical people whose knowledge on machine learning can not be assumed. Second, previous systems evaluate performances with quantitative metrics such as the F1 score, but these quantitative metrics are not applicable in most prompt evaluation scenarios. Third, previous systems guide users by visualizing the relations between certain hyperparameter settings and the performance metrics. In prompt engineering, the search space is all possible text expressions that can not be enumerated, and the non-deterministic nature of prompting introduces a high uncertainty in the outputs, making it hard to identify relations between prompts and performances. In our work, we explore the possibility of evaluating prompt performances with feature metrics, which are more approachable for non-technical users and can guide them in discovering the relations between prompts and performances.

2.3 Interactive Prompt Engineering and Design Studies

Given its uniqueness, several design studies have been conducted to explore the challenges in prompt engineering [18,61] and evaluation [20]. Zamfirescu et al. [61] found that prompt evaluation practices of non-technical users are *opportunistic* rather than systematic, due to the lack of experience in controlling automatic systems. Jiang et al. [18] found the mental load of *manually* skimming large volumes of text introduces a significant evaluation challenge. Moreover, it is hard to transform the evaluation into prompt refinement, i.e., *little actionable insight is generated*. Kim et al. [20] focused on prompt evaluation and reported two additional challenges. First, evaluation is *multi-criteria*, i.e., the quality of the outputs could not be evaluated with a single criterion. Second, evaluation is *dynamic*, i.e., designers expand or change their criteria as they observe unexpected flaws in the outputs.

Many works have attempted to support prompt engineering in an interactive, code-free environment. PromptIDE [48] and PromptIterator [49] support the iterative experimentation process. As researchers and practitioners introduce more prompting techniques and best practices [1,2], Kim et al. [19] and Arawjo et al. [6] propose to design chains of reusable blocks to separate prompt design, model selection, and eval-

uation. Another line of work focuses on providing prompt suggestions, ranging from word-level [32], sentence-level [8], to prompt-level [29], where users prompts are replaced with an expert prompt template.

Previous design studies and applications either focus on evaluation at the instance scale, where the prompt is tested on at most a few instances, or support only evaluation at the dataset scale in supervised settings, where performances can be measured with loss or accuracy with labeled datasets. In our work, we explore the tasks and designs to support text summarization evaluation at the dataset scale, which is under-explored and calls for new visualization techniques.

3 PILOT STUDY

From the literature review, we have learned that quality metrics are ineffective in text summarization evaluation [7, 10, 34] and hypothesized that feature metrics, such as complexity and naturalness, could be a new way of dataset-scale prompt evaluation. As prompting is a relatively new research area and the typical workflows and challenges are not well-studied, we conducted semi-structured interviews with 6 participants to verify findings from existing works [18, 20, 61], explore new challenges that emerge at dataset scale, and confirm the feasibility of using feature metrics for dataset-scale prompt evaluation. We recruited users of LLMs from both technical and non-technical backgrounds (e.g., ChatGPT). Below, we report our study design and findings.

3.1 Study design

In our interview, seek to answer the following research questions (**RQs**):

- **RQ1**: *What are the challenges in dataset-scale summarization prompt evaluation?*
- **RQ2**: *Can feature metrics address the challenges in **RQ1**?*
- **RQ3**: *What is the preferred level of explanation of feature metrics?*
- **RQ4**: *What is the preferred way of supporting prompt optimization?*

To provide context for the participants, we used summarization on a news article dataset as a simulated scenario in the interview. For **RQ1**, participants were first asked to write an initial prompt that can summarize a news article, and then brainstorm ways to evaluate it. Then, we introduced a set of potential features to be used for evaluation, and then asked participants to brainstorm ways of evaluation again (**RQ2**). To answer **RQ3** and **RQ4**, we prepared different levels of feature metric explanations and prompt editing support, as shown in Figure 1, and asked the participants to choose levels that they prefer and explain why.

Participants   We recruited participants with varying backgrounds: two data/visualization scientists (**P1-2**), two NLP researchers (**P3-4**), and two non-technical researchers with backgrounds in environmental science (**P5-6**). They also have varying levels of experience with prompts. The most experienced participant is an LLM researcher who has worked on multiple related research projects and an internship. Middle-level experienced participants (N=2) had designed prompts programmatically. The least experienced participants (N=3) had used ChatGPT for preparation of presentations, or summarizing long texts. Participants reported frequent usage of prompts for at least half a year.

Procedure   The 30-minute interview was conducted in a semi-structured way. We started by asking about the participants' background, experience with prompting, and their typical workflow. Then, we introduced a simulated scenario in which they designed a summarization prompt for news articles, that takes **one** news article and outputs **one** summary and emphasized that the prompt should be generalizable to 100 articles. Participants were not required to write actual prompts since we were only interested in their thought process. Finally, we asked about their preferences on different levels of feature metrics and prompting support. All participants received a 5 USD compensation.

3.2 Findings

Regardless of their backgrounds, all participants reported a similar workflow for prompting: first write an initial prompt to get a baseline response, and iteratively refine the prompt by skimming through the responses. The reported challenges align with the five challenges we summarized from the literature review. In addition, we identified several new findings when prompts are evaluated at the dataset scale:

F1 **Dataset-scale evaluation is infeasible without support.** The simulated scenario revealed that some external support is necessary for dataset-scale evaluation (**RQ1**). In their current practice, participants would randomly pick a few examples to see if the prompt works well. All participants (**P1-6**) agree that such a way of evaluation becomes infeasible as the evaluation scales up to a dataset as they would need to manually skim through hundreds of summaries, far beyond the physical cognitive limit of human beings. Some (**P3, P4**) suggested automated metrics as a possible solution, but *"it is hard to find a metric that can generalize across the diverse tasks enabled by prompting"*. The non-deterministic nature of prompting makes it hard to guarantee that the prompts can generalize well without actually executing the prompts, which would be costly in time and slow down the iterative refinement.

F2 **Features are critical to dataset-scale evaluation.** Decomposing goals into features presents opportunities to address the dynamic and unactionable challenge. When asked to evaluate prompts that generate an "academic" summary, participants were clueless about the suitable evaluation criteria. However, after introducing the feature metrics, all participants realized and agreed that these metrics could be used as evaluation criteria to cover various summarization requirements (**RQ2**). For example, "academic" could be expressed as complex, very formal, and having a neutral sentiment. Previous works have pointed out the *dynamic* challenge, Our interview revealed that not knowing which features constitute the intended goal is the major cause of the *dynamic* challenge, where designers frequently change the evaluation criteria and redefine "success" after finding unexpected flaws in the output [20]. By assisting designers to systematically make sense of potential features and select the most appropriate ones, the evaluation criteria are less likely to change, and designers can identify weak aspects in the iterative refinement process, addressing the *unactionable* challenge.

F3 **Metrics are guidance, not target.** When asked about the preferred explanation of feature metrics (**RQ3**), participants predominantly (**P2-6**) chose L2: *textual definitions in computational linguistics* (Figure 1). Most participants excluded L1 for being too generic and L3 for being too specific for the model to follow as an instruction. Moreover, some (**P2, 4, 6**) had doubts about the reliability of the L3 computational formula, questioning that *"it might not be a good representation of complexity"*. **P2** emphasizes that *"(designers) should not optimize for the formula, because LLMs can understand complexity more deeply"*. This observation echoes with GoodHart's Law [46]: *when a measure becomes a target, it ceases to be a good measure*, and is reflective of the current situation in text summarization, where researchers are discouraged from using metrics like ROGUE to evaluate LLM-generated summaries. As a result, we assign each feature with semantically meaningful categorizations and refrain from showing metric values.

F4 **Prompting methodology is more important.** All participants preferred prompt suggestions over prompt replacements (**RQ4**), despite the different levels of prompt editing support introduced in previous works. **P6** commented that *"(receiving suggestions) is a more learnable experience"*. Other participants also expressed the desire to *learn* to design good prompts, i.e., the methodology of prompting [1, 2]. We also found that even without knowledge of the established methodologies, participants developed similar methodologies through their own prompting experience, such as chain of thoughts (CoT) [56] or providing examples. This suggests that the design of prompting support should communicate the underlying prompting methodologies.

4 REQUIREMENT ANALYSIS

Combining the literature review and the pilot study findings, we identified four tasks that users must perform in a systematic evaluation workflow: Feature Selection, Example sourcing, Prompt Refinement, and Evaluation for Refinement, as shown in Figure 2. Since the workflow is guided by features, we refer to it as a *feature-oriented* workflow. Next, we introduce each task and discuss its necessity in detail.

4.1 Systematic Workflow

T1 **Feature Selection:** Designers need to make sense of the potential features and their correlations, and then pick a subset that best represents the intended goal. Designers should avoid fixating on minor feature values F3, and instead set target ranges
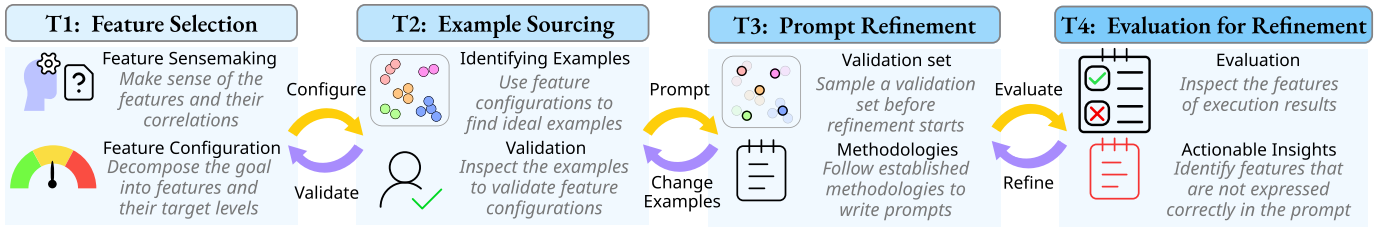
Fig. 2: The feature-oriented workflow for a systematic dataset-scale prompt evaluation. **T1**: Feature Selection. Designers should make sense of potential features to use for evaluation and their correlations, and then select a feature configuration that consists of a subset of features and corresponding target ranges. **T2**: Example Sourcing. Designers should use the feature configuration to find ideal examples, which are used to validate (and refine) the configuration and in the prompt. **T3**: Prompt Refinement. First, select a validation set before the refinement starts, and then follow established prompting methodologies to write the prompt that expresses the feature configurations. Execute the prompt for validation when ready. **T4**: Evaluation for Refinement. Inspect the evaluation result on selected features, identify cases that do not fit the feature configuration, and refine the prompt accordingly. We emphasize that features should be used as guidance, not targets.

(configurations) of the features. As indicated by **F2**, making Feature Selection explicit and systematic enables the subsequent evaluation to become less dynamic and more predictable.

T2 Example Sourcing: Once the features are selected, designers should find ideal examples that fit the features, which can be used to validate the feature configurations. Examples are also used in the prompts or compare the outputs in subsequent tasks.

T3 Prompt Refinement: To guarantee a systematic evaluation, designers need to also refine prompts systematically. First, a representative *validation set* must be chosen statistically to guarantee its validity. Then, prompting should follow the methodologies proposed by prompt engineers and researchers to write better and more controllable prompts **F4**. Finally, prompt drafts are executed on the validation set to generate insights for refinement.

T4 Evaluation for Refinement: With previous tasks, we establish a base to ensure that a systematic evaluation is possible. After getting feedback from the selected features, we emphasize that designers should use features as guidance, not targets **F3**. Once the refinement on the validation set is completed, designers can test the prompt on the whole dataset to confirm the performance.

Ideally, designers would complete the tasks sequentially (Figure 2, yellow arrows). Nevertheless, the workflow supports an iterative process: designers can use the feedback from a later task to refine a previous task (purple arrows). For example, if **T3** shows that the examples are not representative, designers can go back to **T2** for better examples. Such a systematic workflow presents a structured way of refining and evaluating prompts, allowing easy identification of failure points.

4.2 Design Requirements

A systematic evaluation with the above tasks is non-trivial for prompt designers to do properly **F1**, especially for non-technical people. We summarized four design requirements to support **T1–4**:

R1 Support sensemaking and recommendation of feature metrics. As our target audience does not necessarily come from a technical background, selecting features introduces a steep learning curve. Our system should support an automatic recommendation of feature metrics based on the user's goal and provide explanations.

R2 Support overview and identification of ideal examples. Identifying and validating ideal examples at the dataset scale is cognitively demanding. To facilitate this process, our system should use statistical analysis to guarantee the soundness of the examples and interactive visualizations to reduce the cognitive load.

R3 Provide suggestions based on established methodologies. Prompting is a relatively new area and methodologies are constantly improving. As our users might not be aware of such advancements, our system should be designed to enforce state-of-the-art methodologies and give suggestions when necessary.

R4 Support visual tracking of prompt refinement effects. It is critical yet demanding for designers to track the evaluation results with pure statistics. Our system should support such tracking through visualization and convey the performance of each prompt.

5 *Awesum*: System Design

Based on the design requirements, we developed *Awesum*[1], a visual analytics system that supports the feature-oriented workflow on dataset-scale summarization prompt refinement and evaluation (Figure 3). Next, we introduce the details of each component in detail.

5.1 Feature Computation

Awesum characterizes summaries with six features and guides users toward their goals. We use well-established metrics for complexity, formality, and sentiment, but no known metrics suit our needs for faithfulness and naturalness, so we introduce new computations for them. Following **R1**, we categorize each metric into easy-to-understand levels, encouraging users to not fixate on minor differences in metric values. Below, we briefly summarize their definitions. Computational details and categorizations are presented in the appendix.

Complexity The complexity score characterizes the ease with which a reader can understand a written text. We use the Flesch Reading Ease Index [21] to measure the complexity of a text based on sentence count, word count, and syllable count. We simplify its original eight-level categorization into five levels: *Elementary, Middle School, High School, College*, and *Professional*, which indicates the knowledge level required for a reader to easily understand the text.

Formality The formality score characterizes how formal a piece of text is in terms of linguistic structures, conventions, and vocabulary. We use the measure of textual lexical diversity (MTLD) [30], which is calculated from the ratio of unique word stems to the number of words. We categorize it into *Informal, Standard, Formal* and *Very Formal*.

Sentiment The sentiment score characterizes the emotional tone expressed in a piece of text. We use VADER [17], a lexical-based sentiment analysis model to measure the sentiment of a text, which generates a score between $[-1, 1]$. Then we categorize the sentiment into *Negative, Neutral,* and *Positive* using a threshold of 0.3.

Faithfulness Faithfulness characterizes the degree to which a generated text is consistent with the input information in terms of semantic similarity, completeness, and accuracy [26]. We incorporate this feature in response to the "hallucination" issue that exists in most LLMs and is of the most concern to prompt designers. Although more advanced approaches exist, we calculate the faithfulness score based on Named Entity Recognition overlap (NER-overlap) [23] for its transparency, robustness, fast computation speed, and reasonable alignment with human judgment. We categorize it into *Bad, Low, Avg,* and *Good*.

Naturalness Naturalness characterizes how well a text reads like human-written. To the best of our knowledge, no metrics have been proposed to evaluate the naturalness of texts. Based on the insights from Pu et al. [33] that LLM-generated text exhibits statistical differences in certain linguistic features, such as part-of-speech (POS) tags, we conducted experiments on a dataset [63] that contains both human-written and LLM-generated summaries to select differentiable linguistic features and use them to compute the naturalness score with a weighted sum. The score is similarly categorized into *Bad, Low, Avg* and *Good*.
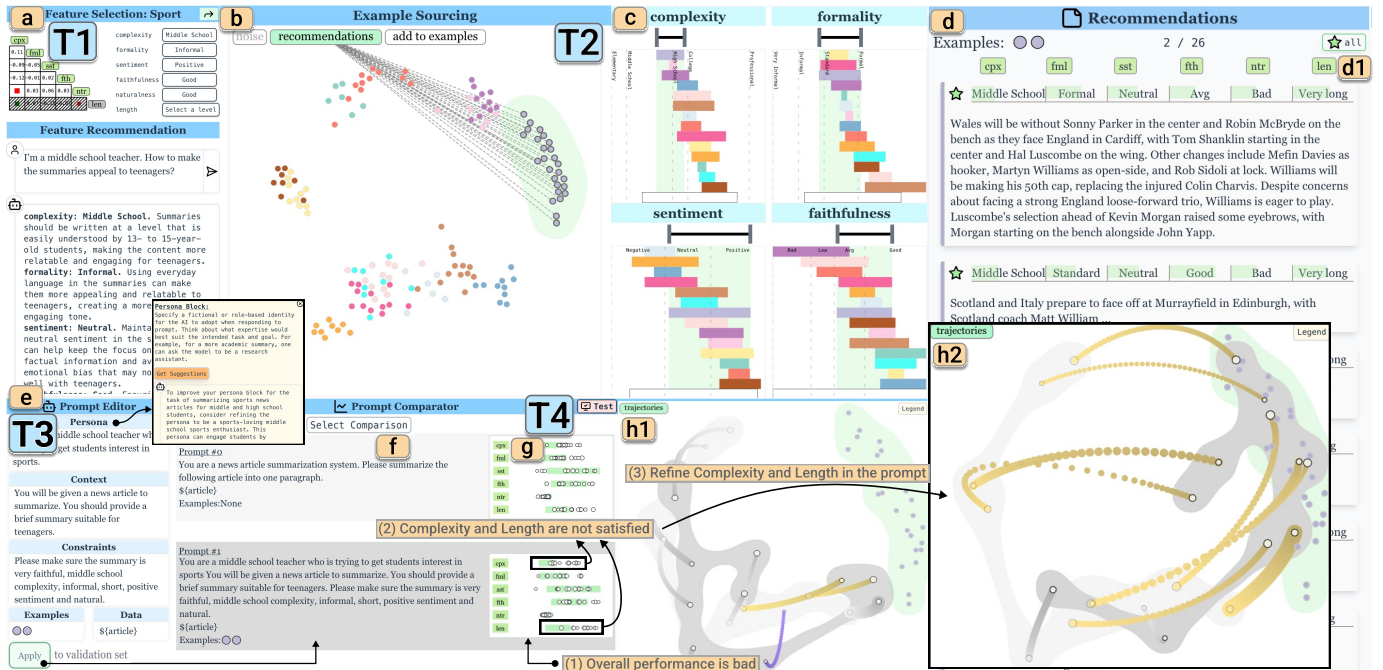
---
[1] https://github.com/SamLee-dedeboy/Awesum

Fig. 3: Actions done by Alice in the case study. Alice uses Feature Selection View (a) to decompose the summarization goal "appeal to teenagers" into feature configurations T1. After setting the configuration, the system finds the closest cluster in (b) and highlights it in a green bubble. She uses Recommendation View (d) to validate the examples and adjust the feature configuration in (c) T2. Then, she moves on to Prompt Editor View (e) and writes the first prompt according to the feature configuration T3, with the help of the prompt suggestions given by a chatbot. She executes the first prompt and to compare it with the baseline prompt in Prompt Comparator View (f), which provides visual tracking of all the versions of prompts. From Bubble Plot (h1), she can see that the prompt is performing badly, as the dark and light gray bubbles are overlapping and not close to the green bubble. She goes to the dot plot (g) and finds that complexity and length are not satisfied, as many validation cases are falling out of the green bars. She refines the prompt by adjusting how she expresses complexity and length T4 by taking prompting suggestions from the system, and the new prompt yields much better performance (h2), as the dark gray bubble overlaps significantly with the green bubble, and most of the curves are yellow.

Length We use word count as the length of a text. Although a simple feature, prompt designers and our pilot study participants have reported difficulties in controlling the length of the generated text. We thus include it and categorize it into *Short, Mid, Long,* and *Very Long*.

Finally, we construct a feature vector $F = (f_1, f_2, f_3, \dots)$ for each summary, where f_i is the numerical score of a feature calculated on the generated summary. Then for each feature, we compute its z score as the value of each dimension in the feature vector. The feature vector of each summary is the basic computation unit in other parts of the system, namely correlation analysis, clustering, and dimensionality reduction.

We chose the above six feature metrics because there is a clear semantic meaning to each feature. Considering that our target users could include non-technical people, the interpretability of the metrics is essential to a systematic and rigorous evaluation. Moreover, we refrain from using LLM-evaluators to support user-defined features [20], even though it might broaden the applicability of the system, for three reasons. First, LLM evaluators have received criticism for their inconsistent alignment with human judgment [43] and potential biases [50]. Second, we have conducted experiments to show that the non-deterministic nature of LLMs makes them unreliable in dataset-scale evaluation. Experimentation details are presented in supplemental materials. Third, the goal of developing *Awesum* in this paper is to verify the effectiveness of the feature-oriented workflow on text summarization task. Considering the above limitations of LLM evaluators, it might introduce unnecessary confounding factors in the evaluation.

5.2 Feature Selection

Feature Selection View supports T1 with two sub-tasks: feature correlation analysis and feature recommendation R1, which supports users to set feature configuration for their goals of the summarization prompt.

Feature Correlation Matrix Feature correlation matrix shows the Pearson correlation coefficients between all pairs of features (Figure 4-a), which is designed to prevent users from selecting conflicting features that are hard to accomplish simultaneously. For example, if the correla-

tion matrix suggests that complexity and naturalness have a negative correlation, users should not pick a configuration that has both high (or low) complexity and naturalness. Visualizing the correlations as a matrix allows users to avoid such conflicts at a glance R1. The L2 explanations of each feature can be inspected by hovering over the feature tags. Based on the result of the baseline prompt, the system calculates all the feature vectors F on the generated summaries and the Pearson correlation coefficient between all pairs of features. If the coefficient exceeds a threshold, we use a square to indicate its significance and encode the correlation strength with the length. Negative and positive correlations are encoded by red and green, respectively. Features omitted by the user in the configuration are shaded in stripes.

Feature Recommendation In case the feature correlation matrix does not provide a concrete idea for the user, *Feature Recommendation Panel* (Figure 4-c and -d) integrates a chatbot to recommend feature configurations. Users can express their goals in natural language (e.g., generate summaries suitable for academic writing), and the chatbot will respond with a recommended configuration with explanations R1. Under the hood, we provide the L2 feature definitions and their categorizations to the chatbot to ensure reasonable responses. If significant correlations exist in the features, the chatbot would highlight the correlations and give suggestions accordingly. The system automatically fills in the recommended configuration (Figure 4-b). Dropdown menus are provided to change feature levels manually.

5.3 Example Sourcing

Example Sourcing View supports T2 with Cluster Plot (Figure 3-b), which presents an overview of the dataset, and a side panel that can be switched between Cluster Profiles (Figure 5), which visualizes cluster feature characteristics; and Feature Distributions (Figure 3-c), which provides finer control over feature ranges selection R2. Users can inspect Cluster Profiles or Feature Distributions and narrow down the target feature ranges (green bubble in Figure 3-b and green ranges in Figure 3-c) to identify ideal examples.
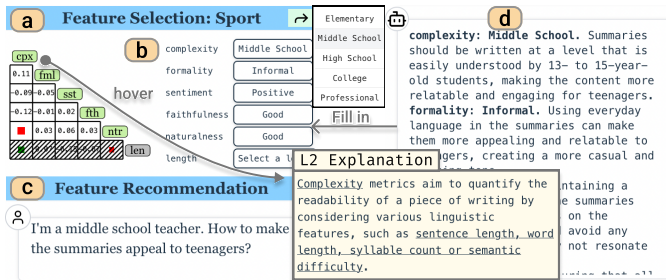
Fig. 4: Feature Selection View is designed to support T1. The feature correlation matrix (a) facilitates sensemaking of the feature definitions and their correlations. If the correlation exceeds a certain threshold, it is encoded by a red (negative) or green (positive) square. The rows and columns are sketched out in stripes if the user deems a feature irrelevant to the goal. Then, users can manually set the feature configuration in (b) using the dropdown menus, or use the feature recommendation chatbot (c) to get recommendations. The chatbot responds in (d) with a recommended configuration by considering the L2 definition of each feature and the user's goal and automatically fills them in (b).

Cluster Analysis The system applies the OPTICS clustering algorithm [5] on the initial summaries using their feature vectors. The OPTICS algorithm has several benefits over other clustering algorithms such as KMeans [28]. First, OPTICS is more flexible as it automatically detects the cluster densities to decide cluster numbers and thus does not require prior knowledge of the dataset's cluster shapes. Second, it ensures that all generated clusters have low variance in feature vectors, making each cluster distinctive. Third, it removes "noises" that are not close to any clusters, which is ideal for identifying examples as our users might not have the cognitive power to identify examples from a large amount of data. These benefits make the OPTICS algorithm ideal for our system. In addition, The clustering results are used to generate the validation set by under-sampling [60], i.e., cluster centroids are assigned to the validation set, ensuring the diversity of the validation set for robust evaluation.

Cluster Plot In Cluster Plot, each initial summary is encoded as a circle and colored by their corresponding cluster. *Awesum* applies Kernel PCA [40] with cosine distance on the feature vectors of the initial summaries to generate 2D coordinates, which plots clusters with similar feature characteristics closer to each other, effectively creating regions in 2D space that represent certain characteristics. One design consideration for Cluster Plot is to maintain the visual continuity between example identification and tracking prompt refinement effects (subsection 5.6). Thus, we look for *parametric* dimensionality reduction methods, where an explicit mapping function (i.e., projection to low-dimensional space) can be reused, excluding popular non-parametric methods like t-SNE [51], UMAP [31] or MDS [11]. Kernel PCA allows us to reuse the same projection in subsequent steps where a visual tracking of the performances of different versions of prompts is provided, while capturing the non-linear feature relationships in the data. In addition, we employ two techniques to improve visual clarity. First, we hide noise points deemed by the OPTICS algorithm as they are not representative. We provide a toggle to show the noise in case users are interested. Second, we apply collision detection and a force-directed layout that attracts each point to its cluster's centroid. Although this would affect point positions given by Kernel PCA projections, this is acceptable as the exact position of each point would not affect users identifying ideal examples T2. Since our target audience is non-technical prompt designers, we emphasize clarity over accuracy.

Cluster Profiles Cluster Profiles show the feature ranges of each cluster in a scaled vertical bar chart (Figure 5). Considering R2, we want to highlight distinguishing feature characteristics among clusters through Cluster Profiles. Inspired by the Difference Overlay design [44], we scale the bars by aligning the global mean of each metric at the center of a profile. Then, we take the maximum of $(global\_max - global\_mean, global\_mean - global\_min)$ as the range of half of the width and scale each bar accordingly. This way, users can easily identify the distinctive features of each cluster. Clicking on a
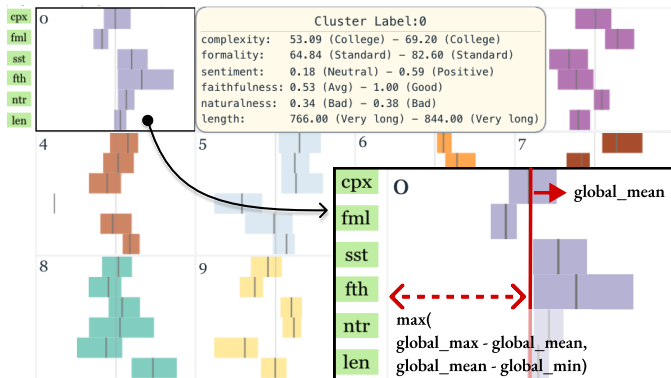

Fig. 5: Cluster Profiles visualize the feature ranges of each cluster with a scaled vertical bar chart. Each cluster is encoded by a unique color. The bars are aligned by the center vertical line, which encodes the global mean. The half of the profile width is encoded by $max(global\_max - global\_mean, global\_mean - global\_min)$. This way, the bars are scaled to show the distinctiveness of each cluster. Hovering over a bar chart shows the ranges in values and categorization.

cluster profile highlights its position in Cluster Plot (green bubble) and automatically sets the cluster points as ideal examples.

Feature Distributions Feature Distribution Panel (Figure 3-c) supports users in identifying ideal examples with finer control over the feature configurations R2. It shows the unscaled ranges of each cluster for each feature, which complement the scaled ranges in Cluster Profile while maintaining the same visual encoding for colors (cluster label), emphasizing visual continuity and simplicity. The cluster bars are ordered from top to bottom by their mean value on the corresponding feature. For each feature, users can click on a cluster bar to set its range as a target or drag the double-direction slider at the top to adjust the target range, indicated by a light green background. Supporting T2 at a finer scale ensures the system adapts to diverse goals.

5.4 Recommendation

Recommendation View (Figure 3-d) supports validation of the target ranges of the features T2. It shows the content of the examples with their feature categorizations. After inspecting the examples, users can click the star icon and add them to the prompt.

The design of Recommendation View considers two weaknesses in Example Sourcing View. First, it is hard for users to estimate the number of recommended examples in Cluster Plot, as users would need to estimate the region's area to estimate the number. To complement, we use a fixed height for each example in Recommendation View by collapsing excessive content, ensuring as many examples are shown in the viewport as possible. This allows users to estimate the number of examples at a glance with the total height of the examples. Second, Cluster Profiles are scaled so they do not strictly encode the range of each feature. This could mislead the users into choosing the wrong feature configuration. To complement, we encode the feature values in horizontal green bars (Figure 3-d1). Users can skim through the examples with awareness of the true ranges of each feature at a glance.

5.5 Prompt Editor

Prompt Editor View supports users with Prompt Refinement T3. Informed by state-of-the-art prompting methodologies [1,2], we divide a prompt into five blocks: *Persona, Context, Constraints, Examples* and *Data*. In each block, users have a clear goal for the content of the section R3. For example, Persona block should *specify a role-based identity for the AI to adopt*. This explicit division of a prompt enforces the users to follow established prompting methodologies with a minimum requirement for knowledge of the methodologies. Once a prompt is written, users can click the "Apply" button to test it on the validation set. In addition, users can hover over the block titles to learn the purpose of the block and get suggestions specific to their feature configuration from a chatbot R3, as shown in Figure 3-e.
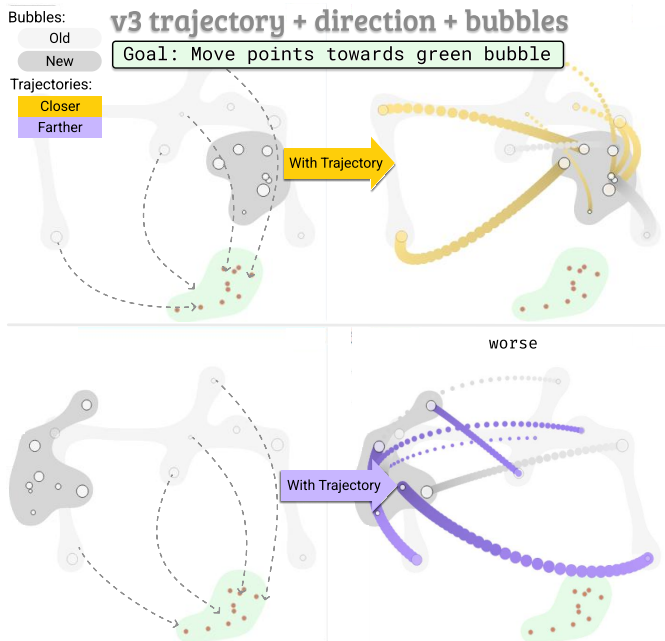
Fig. 6: Final design of Bubble Plot. The light green bubble represents the goal. Light and dark gray bubbles (representing old and new prompts) allow users to estimate prompt performances by the distances from the green bubble. A visual pattern that indicates a "better" prompt (top) would have the dark bubble moving closer to the light green bubble, and vice versa (bottom). Colored curves connect validation cases in old and new prompts, with color encoding changes in feature values (yellow for better, purple for worse, gray for insignificant). The importance of a validation case is encoded by the circle size (and trajectory width).

5.6 Prompt Comparator

Prompt Comparator View supports T4 with two components: Prompt Tracking Panel (Figure 3-f) and Bubble Plot (Figure 6). Both components incorporate visualizations to reduce the cognitive load of keeping track of multiple iterations of prompts and their performances R4. Since the refinement is conducted on the validation set, a "Test" button is provided to execute the prompt on the whole dataset once users are satisfied with the prompt. Next, we introduce each component in detail.

Prompt Tracking   Prompt Tracking Panel (Figure 3-f) stores all the prompts that the user has written, including the prompt content, the examples, and their evaluation result using a horizontal dot plot (Figure 3-g). Tracking the history of prompts is essential as prompt refinement and evaluation is a highly iterative process. To provide visual tracking R4, each prompt snippet includes a dot plot that visualizes the performance of the prompt in detail, where each row is a feature and each dot in a row is a validation case. Since the validation set is generated from the cluster centroids, we encode the dot size with its corresponding cluster size to indicate its importance, i.e., larger dots are more important because they represent more points. We stroke the feature configuration in light green on the dot plot to clearly present the goal of the prompt: make all the dots fall into the light green bars.

Bubble Plot   Bubble Plot (Figure 6) supports visual comparison between any two prompts and visualizes their "distances" from the ideal examples to support T4. As illustrated in subsection 5.3, we reuse the Kernel PCA projection in Cluster Plot for all points in the plot to ensure visual continuity. Ideal examples are highlighted in a light green bubble as an indication of the goal. Two iterations (old and new) of the validation cases are plotted as circles, where the circle radius encodes its corresponding cluster size to suggest importance, and are surrounded in light and dark gray bubbles, respectively. This allows the user to estimate the performance of a prompt through the distance between its bubble and the green bubble R4. The expected visual pattern for a "better" prompt is that the dark bubble is "moving towards" the green bubble, and vice versa. To make the comparison more explicit, we connect the same validation case from two versions of prompts with
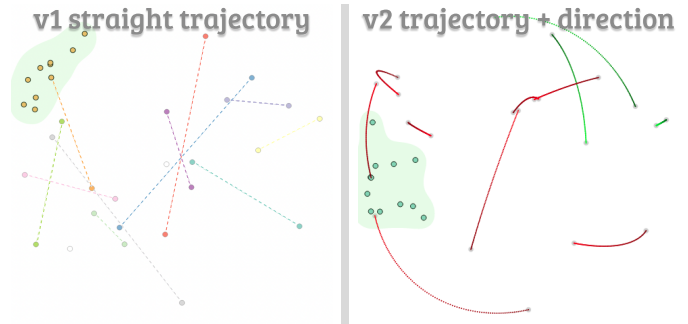

Fig. 7: Previous designs of Bubble Plot. **v1**: Validation cases are encoded by cluster colors. Animated dotted straight lines connect validation cases from old and new prompts. It is cluttered and hard to gain any valuable insight. **v2**: Cluttering is reduced by using curves to connect points, which are generated by sampling in high-dimensional space. Colors are based on the directions (green for better, red for worse). The brightness (light to dark) encodes the connection direction (old to new prompt). The visual pattern highlights comparison, but it is hard to make sense of the performance of individual prompts.

curves generated with linear sampling in high-dimensional space. This effectively visualizes the "trajectory" of the validation case incurred by the new prompt. Specifically, for each pair of feature vectors (F_1, F_2), we linearly sample $\varepsilon = 100$ points between F_1 and F_2 and project each point to 2D space, thus forming the curve and encoding the importance with curve width.   Since the projection is non-linear, the sampled trajectory appears to be curved. Also, using a fixed number of sampling points (ε) makes long trajectories appear dotted.   We use a color-blind-friendly color palette to indicate the direction of the trajectory: yellow if the validation case is moving closer to the ideal examples, and vice versa, purple. Validation cases with non-significant changes (below a threshold) are colored in gray. Users can click the "Select Comparison" button to select any two iterations of prompts to compare.

Bubble Plot Design Iteration   The design of Bubble Plot went through three iterations, as shown in Figure 7. In **v1**, we encoded the cluster of each validation case with color and connected validation cases from two different prompts with an animated straight dotted line to indicate their moving directions. However, the lines were cluttered, and interactions were needed for further analysis. Barely any insights can be generated at a glance. Inspired by Scheepens et al. [39], in **v2**, we mitigated the clutter issue by connecting validation cases with curves generated by linearly sampling in high-dimensional space and used green (closer) and red (further) to encode the moving direction from the validation cases. Users can now compare two prompts at a glance, but it was hard to estimate the performance of individual prompts, which is critical in R4. This motivated us to surround the validation cases with BubbleSets to indicate the relative distance from the target bubble, providing a clear visual tracking of prompt performance.

6 CASE STUDY

We demonstrate how a non-technical prompt designer with no prior prompting experience can use *Awesum* to refine and evaluate prompts.

Alice is a middle school teacher and she wants to customize a chatbot that can summarize sports news in a way that appeals to teenagers. She loads the sports-related materials into *Awesum* and uses a baseline prompt to generate the initial summaries. The system executes the prompt and processes the summaries. Alice starts by selecting the features that make a summary appeal to teenagers T1. She asks in Feature Recommendation Panel: "I'm a middle school teacher. How to make the summaries appeal to teenagers?". The chatbot answers with a recommended feature configuration with explanations. She checks the definitions of the features and adjusts "sentiment" to "positive" as she wants the summaries to be more energetic, as shown in Figure 3-a.

After setting the feature configuration, she then proceeds to find ideal examples that fit this configuration T2. She clicks the arrow in the green background at the top right, which triggers the system to automatically find a cluster that best fits the feature configuration. The system highlights the purple cluster in a green background and

dotted lines that connect to "recommendations" at the top (Figure 3-b). She skims through the content and the feature ranges of the purple cluster in Recommendation View (Figure 3-d) to ensure that they fit her expectations. She is satisfied with the cluster and selects the two best examples to use in the prompt (starred in green).

After Feature Selection and Example Sourcing, Alice proceeds to write the prompt T3 . Even without prior prompting experience, she can easily understand what each block means and what should be written as the blocks resemble interaction patterns with chatbots. She clicks the "Get Suggestions" button to brainstorm some ideas for Persona block (Figure 3-e), then decides to put "You are a middle school teacher who is trying to get students interested in sports news" in Persona block. She repeats similarly for Context block and Constraints block and then executes the prompt on the validation set. The system executes the prompt and recalculates the features. The new prompt's content and its evaluation result are presented in Prompt Tracking Panel (Figure 3-f), with the new prompt colored in dark gray and the baseline prompt colored in light gray. From Bubble Plot (Figure 3-h1), she can quickly tell that the new prompt has a bad overall performance as the gray bubble is not moving closer to the green bubble. She inspects the dot plot (Figure 3-g) and finds that *Complexity* and *Length* are not being satisfied. She thus goes back to the prompt and refines the parts where she describes complexity and length T4 , seeking the prompt suggestion chatbot for help. After the refinement, the new prompt is much better as the dark gray bubble has significant overlaps with the green bubble with only a few exceptions. The trajectories show that most validation cases are moving towards the green bubble, and only a few do not have significant changes. Alice is satisfied with this prompt and she clicks the "Test" button to test it on the whole dataset, which shows that the prompt works as expected.

This case study shows that the system is effective in supporting dataset-scale prompt refinement and evaluation. Using the system, even non-technical people with no prior prompting experience could write a reasonably good prompt, evaluate the prompt, and figure out which parts need to be refined. By following the systematic feature-oriented workflow behind the system, users overcome the first four challenges (opportunistic, manual, multi-criteria, and dynamic) in prompt evaluation. Cluster Plot and Bubble plot allow users to overcome the unactionable challenge by visualizing the prompt performance.

7 PRACTITIONER REVIEW AND DISCUSSION

Awesum is evaluated through a practitioner review. Even though the system is designed for text summarization, we aim to explore a broader range of tasks in people's everyday work that can also be supported. We recruit practitioners from various backgrounds who are interested in customizing LLMs to automate their professional workflow. We evaluate the effectiveness of the system with them and discuss its generalizability. In addition, we report challenges in prompt evaluation and human-agent interaction that arise in professional usage.

7.1 Participants and Procedure

Four practitioners from different backgrounds were recruited: C1 is a correspondent for the New York Times specializing in scientific topics; C2, C3, and C4 are researchers in ecology, NLP, and sociology, respectively. C1, C2, and C4 came from non-technical backgrounds and had only prompted in ChatGPT. All participants expressed strong interest in customizing LLMs with prompts to automate their workflow.

The review procedure consists of three sessions, 20 minutes each. In the first session, we introduce the background and give a quick walkthrough of the interface. In the second session, participants choose a topic (from Politics, Sport, Technology, and Business), propose a summarization goal, and then use *Awesum* to write prompts that fulfill that goal. Finally, participants engage in a semi-structured interview. We provide at least 25 US dollars in compensation for all participants.

7.2 User Feedback

Overall, the system received positive feedback on usability and user-friendliness. Still, a minimum amount of training is needed to use the system. Below, we report the practitioner review on system design, prompting methodology, learning curve, and visualization literacy.

System Design The system was highly praised for being easy to follow despite its complexity. Participants commented that *"I have no trouble understanding what I should do (C2)"* and that *"The organization makes a lot of sense (C1)"*. All participants successfully wrote a well-performed prompt for their goals based on the feedback from the system. As **C3** said, *"You can tell (from Bubble Plot) that initially my prompt wasn't good, then I changed the prompt according to the features (L2 descriptions) and it's giving exactly what I wanted."* Bubble Plot and dot plots facilitated participants to examine and locate unfulfilled features and the L2 descriptions helped them refine prompts.

Prompting Methodology Providing suggestions supported by the established prompting methodologies inspired less experienced participants to explore more possibilities, going beyond our initial expectation that simply facilitates prompt writing: *"(C4): If I had just a blank text box, I would not know how to write the first sentence . . . and I would not know that this is something that you can do. "* Most inexperienced prompt designers find prompting challenging because they do not know what the model is capable of. Before using the system, C4 did not know that designers could instruct the model to adopt a persona, and that inspired her to explore what could be used as a persona. As opposed to L1–3 prompt editing support (Figure 1), providing prompting suggestions (L4) taught participants what to expect from the model.

Learning Curve There is still a non-trivial learning curve to overcome, despite being highly praised for its clarity. For C1, *"without you to talk me through (the interface), it would have been hard to figure out, (because) there is a ton of information onto one page."* This shows that for people from non-technical backgrounds, engaging in a systematic workflow is not something they are familiar with and could be challenging at the beginning. Still, the clarity of the workflow and visualization design helped smooth this process, as C1 and C4 both commented that *"it was like playing a game"*. By visualizing the prompt's performance with Bubble Plot, the system provided a clear goal and abundant visual hints for the participants to grasp their current status and how far away they were from the goal, leading to a less unpredictable experience.

Visualization Literacy For people unfamiliar with high-dimensional visualization, Cluster Plot could be confusing. As C2 mentioned, *"I can understand that each region has its own characteristics and that the green bubble is my goal, but I would still want to know how it works under the hood to really make sure I'm not misunderstanding."* C4 also raised a similar concern: *"Figuring out what these functionalities do is one thing; figuring out what these functionalities mean, that's another thing."* This is a valid concern since Kernal PCA introduces distortion in the two-dimensional scatter plot, which could mislead users who are not aware of it. Even though we designed Bubble Plot with loose encodings to mitigate this issue and encouraged users to use metrics as guidance, users could still fixate on absolute metric values without a minimum amount of training.

7.3 Design Implications

In this section, we first discuss how the system can be generalized to a broader range of tasks. Then, we report three implications on human-agent interaction: the issue of trust, experimenting with LLM capabilities, and using features as boundary objects.

Supporting Information-Condensing Tasks Beyond summarization, using the system to evaluate prompts for condensing information is a highly praised usage. When asked about how well the system could be generalized to other tasks in their professional work, all participants agreed its strong potential in condensing information, e.g., generating logical relations for multiple articles (C1), leading to the importance of proper evaluation, as C4 mentioned:

"Say I take 2000 tweets, something I might do with AI is to clean these tweets because not all tweets have meaningful things according to my use. . . but I can't use it unless I have a grasp of what the AI's assumption is about 'meaningful'." Challenges in evaluating information condensing tasks resemble those in text summarization in many ways: there is a great amount of information (data); it is cognitively demanding to evaluate even a small amount of outputs; and it is probably unreliable to evaluate all of them with quality metrics. Participants agreed that the current feature metrics in *Awesum* most scenarios for

their professional work, which is reasonable because the standards for "professional writing" are quite similar. In general, the system is shown to readily apply to information-condensing tasks.

Shift in Task Formulation We observe a shift in the formulation of researchable tasks from practical tasks in the practitioner review. For example, to understand a large amount of customer feedback quickly (practical), researchers in traditional machine learning formulate the task as sentiment analysis or topic modeling task (researchable). In the pilot study and practitioner review, non-technical participants can design prompts that directly solve the practical task at hand. In prompt engineering, it is less important to rigorously define a researchable task given the high customizability of prompts. Therefore, while the system seems not applicable to many existing NLP tasks, such as data extraction [24], making it less advantageous for NLP researchers, it still brings much value to non-technical practitioners.

Applying to Image Generation Tasks C3 was very positive about applying the system to image generation tasks: *"you would need some other metrics (for image generation) like the styles of the images, but the other things are more or less the same."* The system design makes it easy to extend feature metrics for a broader range of tasks. Given the right set of features, the rest of the system is agnostic of the underlying features because OPTICS clustering and Kernel PCA both operate on feature vectors. For image generation prompt evaluation, a potential direction for computing feature metrics is to evaluate the style of an image through image similarity analysis [35]. For example, one could collect a dataset of cartoon images and calculate the similarity between a generated image and the dataset to quantify how "cartoony" the image is. Such stylistic metrics could be used as feature metrics to evaluate image generation prompts. In general, the system has the potential to be extended for image generation tasks, but developing corresponding feature metrics is out of scope and we leave it for future work.

Beyond generalizability, participants also brought up topics related to human-agent interaction in their professional work.

Experimenting with LLM Capabilities By observing the changes in features, designers can experiment with prompts to better understand the capability of LLMs, as shown by two unexpected usages by C3 and C4 during the practitioner review. C4 tried paraphrasing, e.g., changing the persona from "reporter" to "editor", and observing changes in relevant features, such as complexity and formality. For inexperienced prompt designers, coping with the ambiguity inherent in natural language has been one of the reasons why prompting is challenging. Experimenting with prompts mitigates this issue as it guides designers in sorting through the ambiguity: *"It's actually not putting stuff in unambiguous form, but still less ambiguous enough that you can track where things are going. (C4)"* C3 used the system for a more technical experiment, where he tested the positional bias [53] of LLMs. C3 put conflicting instructions in different parts of the prompt, e.g., asking the model to adopt the persona of "an elementary school teacher writing for kids" but generate summaries "as academic as you can", and observing which part of the instruction is followed based on the features. Both usages showed that evaluation through features has the potential for understanding and even diagnosing LLMs.

Features as Boundary Objects The practitioner review revealed the potential to use features as boundary objects [45], i.e., a common ground that could be reached between humans and agents in prompting. C2 and C4 both felt that using natural language to communicate with AI agents is different from communication with humans. When asked why, C4 replied that *"I know what I want, but how do I say it has to do with context"*, suggesting that context needs to be presented before proper communication can continue with each new conversation. C2 pointed out that the ability to sort through vague communication is critical: *"If the professor tells me to make something better, I'll figure out myself what is 'better' and how to make it better. But with GPT, I have to say exactly what is better."* We believe that this issue could be mitigated by using features as boundary objects and reverting the initiative in feature selection. Instead of a human selecting the feature configuration, the agent could use the features to make confirmations with the human. For example, to generate an "academic level" summary,

the agent could confirm with the human what "academic level" means by asking for confirmations on the complexity level, formality level, and so on. Incorporating features as boundary objects presents a promising direction for supporting better human-agent interaction.

Issue of Trust As a professional journalist, C1 had serious trust issues with applying intelligent agents in journalism work: *"Because I know about GPT hallucinations, and if I am saying false things in this material that I produce, that's really bad for me."* For C1, the level of trust needed for the agent seems unreachable: *" (To trust it) I have to check every single thing that it says, and it defeats the purpose of having a tool that increases my efficiency."* In the feature-oriented evaluation workflow, we do not consider the evaluation of the trustworthiness of the prompts as it can not be easily measured through any feature that we know of. Supporting the evaluation of trustworthiness remains challenging yet critical for many professional tasks.

8 LIMITATIONS AND FUTURE WORK

Limitations of the system are centered around features. First, the feature metrics have their limitations. For example, the Faithfulness score computation is confined to the token level, which could hinder its accuracy when the entities have different meanings across sentences. The capability of *Awesum* to capture user intent accurately is thus limited by the feature metrics. We provide a more thorough discussion of the limitations of feature metrics in the supplemental material.

Developing or selecting suitable features that match the task at hand remains a challenge in the feature-oriented workflow, especially for non-technical prompt designers. This includes the challenge to decompose a complicated task into subtasks that are easier to evaluate, which remains hard even for technical designers with relevant training [58]. One possible solution is to maintain a repository of features and provide recommendation support for each task, following the approach for facilitating the selection of layouts for large graphs [22]. The practitioner review revealed that the evaluation criteria of most prompting tasks could be decomposed into a combination of features, making it possible to cover endlessly long-tail tasks with a finite amount of features. This observation aligns with the findings of Kim et al. [20], that task-specific considerations are often secondary to broader criteria. Technical contributors can conduct meta evaluation on the features to guarantee their efficacy and maintain a repository of the results, thereby benefiting non-technical prompt designers.

Reliance on the dataset hinders generalizability, as one is not always available in every application scenario. A potential solution is LLM-powered data augmentation [42]. Given a small dataset, LLMs have the capability to generate high-quality data points by retrieving and extrapolating from large datasets with sufficient contextual information. The discussion with C3 revealed that data augmentation is a promising direction to deal with the absence of a dataset.

Finally, certain social factors might not be applicable to the feature-oriented workflow, such as trustworthiness, privacy, ethics, and social biases, which have a significant impact on the adoption of LLM-based intelligent agents deeper into our everyday lives. As mentioned in the practitioner review, the level of trust needed for an intelligent agent to conduct certain professional tasks might take a long time to be gained, defeating the purpose of adopting them for efficiency in the first place. This is the same for other social factors, in that intelligent agents can not be recognized without a transition process that lasts for a significant period. We believe evaluating these social factors is critical before deploying prompts for high-stake tasks, but it is out of the scope of our work and we call for future work to continue on this issue.

9 CONCLUSION

Evaluating human-level summarizations of LLMs requires capturing nuanced quality differences, surpassing the capabilities of quality metrics. Via *Awesum* , we have shown that feature metrics have the potential to provide actionable insights in the summarization prompt evaluation setting. While still limited, the practitioner review reveals positive directions to extend the feature-oriented workflow for broader prompting tasks. We call for more research to follow this direction to advance prompt evaluation via a human-in-the-loop approach.

REFERENCES

[1] Prompt engineering from openai. https://platform.openai.com/docs/guides/prompt-engineering. Accessed: 2024-02-12. 2, 3, 6

[2] Prompt engineering guide. https://www.promptingguide.ai/. Accessed: 2024-02-12. 2, 3, 6

[3] Mljar. https://mljar.com/, 2018. Accessed: 2024-06-12. 2

[4] Sigopt. https://sigopt.com/, 2018. Accessed: 2024-06-12. 2

[5] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: ordering points to identify the clustering structure. In *SIGMOD '99*, 12 pages, p. 49–60. ACM, New York, 1999. doi: 10.1145/304182.304187 6

[6] I. Arawjo, C. Swoopes, P. Vaithilingam, M. Wattenberg, and E. L. Glassman. Chainforge: A visual toolkit for prompt engineering and llm hypothesis testing. In *CHI '24*, article no. 304, 18 pages. ACM, New York, 2024. doi: 10.1145/3613904.3642016 2

[7] M. Bhandari, P. N. Gour, A. Ashfaq, P. Liu, and G. Neubig. Re-evaluating evaluation in text summarization. In B. Webber, T. Cohn, Y. He, and Y. Liu, eds., *EMNLP '20*, pp. 9347–9359. ACL, Online, 2020. doi: 10.18653/v1/2020.emnlp-main.751 2, 3

[8] S. Brade, B. Wang, M. Sousa, S. Oore, and T. Grossman. Promptify: Text-to-image generation through interactive prompt exploration with large language models. In *UIST '23*, article no. 96, 14 pages. ACM, New York, 2023. doi: 10.1145/3586183.3606725 3

[9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds., *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020. 1

[10] D. Deutsch, R. Dror, and D. Roth. Re-examining system-level correlations of automatic summarization evaluation metrics. In M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, eds., *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 6038–6052. ACL, Seattle, 2022. doi: 10.18653/v1/2022.naacl-main.442 1, 2, 3

[11] J. Douglas Carroll and P. Arabie. Chapter 3 - multidimensional scaling. In M. H. Birnbaum, ed., *Measurement, Judgment and Decision Making*, Handbook of Perception and Cognition (Second Edition), pp. 179–250. Academic Press, San Diego, 1998. doi: 10.1016/B978-012099975-0.50005-1 6

[12] E. Durmus, F. Ladhak, and T. Hashimoto. Spurious correlations in reference-free evaluation of text generation. In S. Muresan, P. Nakov, and A. Villavicencio, eds., *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1443–1454. ACL, Dublin, 2022. doi: 10.18653/v1/2022.acl-long.102 2

[13] A. Fabbri, C.-S. Wu, W. Liu, and C. Xiong. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, eds., *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2587–2601. ACL, Seattle, 2022. doi: 10.18653/v1/2022.naacl-main.187 2

[14] D. Golovin, B. Solnik, S. Moitra, G. Kochanski, J. Karro, and D. Sculley. Google vizier: A service for black-box optimization. In *KDD '17*, 9 pages, p. 1487–1495. ACM, New York, 2017. doi: 10.1145/3097983.3098043 2

[15] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau. Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE TVCG*, 25(8):2674–2693, 2019. doi: 10.1109/TVCG.2018.2843369 2

[16] Q. Huang, M. Lu, J. Lanir, D. Lischinski, D. Cohen-Or, and H. Huang. Graphimind: Llm-centric interface for information graphics design, 2024. doi: 10.48550/arXiv.2401.13245 1

[17] C. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, vol. 8, pp. 216–225, 2014. doi: 10.1609/icwsm.v8i1.14550 4

[18] E. Jiang, K. Olson, E. Toh, A. Molina, A. Donsbach, M. Terry, and C. J. Cai. Promptmaker: Prompt-based prototyping with large language models. In *CHI EA '22*, CHI EA '22, article no. 35, 8 pages. ACM, New York, 2022. doi: 10.1145/3491101.3503564 1, 2, 3

[19] T. S. Kim, Y. Lee, M. Chang, and J. Kim. Cells, generators, and lenses: Design framework for object-oriented interaction with large language models. In *UIST '23*, article no. 4, 18 pages. ACM, New York, 2023. doi: 10.1145/3586183.3606833 2

[20] T. S. Kim, Y. Lee, J. Shin, Y.-H. Kim, and J. Kim. Evallm: Interactive evaluation of large language model prompts on user-defined criteria. In *CHI '24*, article no. 306, 21 pages. ACM, New York, 2024. doi: 10.1145/3613904.3642216 1, 2, 3, 5, 9

[21] J. P. Kincaid, R. P. Fishburne Jr, R. L. Rogers, and B. S. Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. 1975. doi: 10.21236/ada006655 4

[22] O.-H. Kwon, T. Crnovrsanin, and K.-L. Ma. What would a graph look like in this layout? a machine learning approach to large graph visualization. *IEEE TVCG*, 24(1):478–488, 2018. doi: 10.1109/TVCG.2017.2743858 9

[23] P. Laban, T. Schnabel, P. N. Bennett, and M. A. Hearst. SummaC: Revisiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177, 2022. doi: 10.1162/tacl_a_00453 4

[24] B. Li, G. Fang, Y. Yang, Q. Wang, W. Ye, W. Zhao, and S. Zhang. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness, 2023. doi: 10.48550/arXiv.2304.11633 9

[25] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81. ACL, Barcelona, 2004. 1, 2

[26] J. Liu, T. Yang, and J. Neville. Cliqueparcel: An approach for batching llm prompts that jointly optimizes efficiency and faithfulness, 2024. doi: 10.48550/arXiv.2402.14833 4

[27] M. Liu, J. Shi, K. Cao, J. Zhu, and S. Liu. Analyzing the training processes of deep generative models. *IEEE TVCG*, 24(1):77–87, 2018. doi: 10.1109/TVCG.2017.2744938 2

[28] S. Lloyd. Least squares quantization in pcm. *IEEE TIT*, 28(2):129–137, 1982. doi: 10.1109/TIT.1982.1056489 6

[29] S. MacNeil, A. Tran, J. Kim, Z. Huang, S. Bernstein, and D. Mogil. Prompt middleware: Mapping prompts for large language models to ui affordances, 2023. doi: 10.48550/arXiv.2307.01142 3

[30] P. M. McCarthy and S. Jarvis. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392, 2010. doi: 10.3758/BRM.42.2.381 4

[31] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020. doi: 10.48550/arXiv.1802.03426 6

[32] A. Mishra, U. Soni, A. Arunkumar, J. Huang, B. C. Kwon, and C. Bryan. Promptaid: Prompt exploration, perturbation, testing and iteration using visual analytics for large language models, 2023. doi: 10.48550/arXiv.2304.01964 3

[33] A. Muñoz-Ortiz, C. Gómez-Rodríguez, and D. Vilares. Contrasting linguistic patterns in human and llm-generated text, 2023. doi: 10.48550/arXiv.2308.09067 4

[34] J. Novikova, O. Dušek, A. Cercas Curry, and V. Rieser. Why we need new evaluation metrics for NLG. In M. Palmer, R. Hwa, and S. Riedel, eds., *EMNLP '27*, pp. 2241–2252. ACL, Copenhagen, 2017. doi: 10.18653/v1/D17-1238 2, 3

[35] G. Palubinskas. Image similarity/distance measures: what is really behind mse and ssim? *International Journal of Image and Data Fusion*, 8(1):32–53, 2017. doi: 10.1080/19479832.2016.1273259 9

[36] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In P. Isabelle, E. Charniak, and D. Lin, eds., *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318. ACL, Philadelphia, 2002. doi: 10.3115/1073083.1073135 2

[37] S. Petridis, N. Diakopoulos, K. Crowston, M. Hansen, K. Henderson, S. Jastrzebski, J. V. Nickerson, and L. B. Chilton. Anglekindling: Supporting journalistic angle ideation with large language models. In *CHI '23*, article no. 225, 16 pages. ACM, New York, 2023. doi: 10.1145/3544548.3580907 1

[38] N. Pezzotti, T. Höllt, J. Van Gemert, B. P. Lelieveldt, E. Eisemann, and A. Vilanova. Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE TVCG*, 24(1):98–108, 2018. doi: 10.1109/TVCG.2017.2744358 2

[39] R. Scheepens, C. Hurter, H. Van De Wetering, and J. J. Van Wijk. Visualization, selection, and analysis of traffic flows. *IEEE TVCG*, 22(1):379–388, 2016. doi: 10.1109/TVCG.2015.2467112 7

[40] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, eds., *Artificial Neural Networks — ICANN'97*, pp. 583–588. Springer Berlin Heidelberg, Berlin, 1997. doi: 10.1007/BFb0020217 6

[41] T. Scialom, P.-A. Dray, S. Lamprier, B. Piwowarski, J. Staiano, A. Wang, and P. Gallinari. QuestEval: Summarization asks for fact-based evaluation. In M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds., *EMNLP '21*, pp. 6594–6604. ACL, Online and Punta Cana, 2021. doi: 10.18653/v1/2021.emnlp-main.529 2

[42] M. Seo, J. Baek, J. Thorne, and S. J. Hwang. Retrieval-augmented data augmentation for low-resource domain tasks, 2024. doi: 10.48550/arXiv.2402.13482 9

[43] C. Shen, L. Cheng, X.-P. Nguyen, Y. You, and L. Bing. Large language models are not yet human-level evaluators for abstractive summarization. In H. Bouamor, J. Pino, and K. Bali, eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 4215–4233. ACL, Singapore, 2023. doi: 10.18653/v1/2023.findings-emnlp.278 2, 5

[44] A. Srinivasan, M. Brehmer, B. Lee, and S. M. Drucker. What's the difference? evaluating variations of multi-series bar charts for visual comparison tasks. In *CHI '18*, 12 pages, p. 1–12. ACM, New York, 2018. doi: 10.1145/3173574.3173878 6

[45] S. L. Star and J. R. Griesemer. Institutional ecology, 'translations' and boundary objects: Amateurs and professionals in berkeley's museum of vertebrate zoology, 1907-39. *Social Studies of Science*, 19(3):387–420, 1989. doi: 10.1177/030631289019003001 9

[46] M. Strathern. 'improving ratings': audit in the british university system. *European Review*, 5(3):305–321, 1997. doi: 10.1017/s1062798700002660 3

[47] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE TVCG*, 24(1):667–676, 2018. doi: 10.1109/TVCG.2017.2744158 2

[48] H. Strobelt, A. Webson, V. Sanh, B. Hoover, J. Beyer, H. Pfister, and A. M. Rush. Interactive and visual prompt engineering for ad-hoc task adaptation with large language models. *IEEE TVCG*, 29(01):1146–1156, 2023. doi: 10.1109/TVCG.2022.3209479 2

[49] S. Sučik, D. Skala, A. Švec, P. Hraška, and M. Šuppa. Prompterator: Iterate efficiently towards more effective prompts. In Y. Feng and E. Lefever, eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 471–478. ACL, Singapore, 2023. doi: 10.18653/v1/2023.emnlp-demo.43 2

[50] L. Tjuatja, V. Chen, S. T. Wu, A. Talwalkar, and G. Neubig. Do llms exhibit human-like response biases? a case study in survey design, 2024. doi: 10.48550/arXiv.2311.04076 2, 5

[51] L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. 6

[52] C. Wang, J. Thompson, and B. Lee. Data formulator: Ai-powered concept-driven visualization authoring. *IEEE TVCG*, 30(1):1128–1138, 2024. doi: 10.1109/TVCG.2023.3326585 1

[53] P. Wang, L. Li, L. Chen, Z. Cai, D. Zhu, B. Lin, Y. Cao, Q. Liu, T. Liu, and Z. Sui. Large language models are not fair evaluators, 2023. doi: 10.48550/arXiv.2305.17926 9

[54] Q. Wang, Y. Ming, Z. Jin, Q. Shen, D. Liu, M. J. Smith, K. Veeramacha-neni, and H. Qu. Atmseer: Increasing transparency and controllability in automated machine learning. In *CHI '19*, 12 pages, p. 1–12. ACM, New York, 2019. doi: 10.1145/3290605.3300911 2

[55] Y. Wang, Z. Yu, Z. Zeng, L. Yang, C. Wang, H. Chen, C. Jiang, R. Xie, J. Wang, X. Xie, W. Ye, S.-B. Zhang, and Y. Zhang. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. In *ICLR 2024*, 2024. doi: 10.48550/arXiv.2306.05087 2

[56] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds., *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824–24837. Curran Associates, Inc., 2022. doi: 10.48550/arXiv.2201.11903 3

[57] K. Wongsuphasawat, D. Smilkov, J. Wexler, J. Wilson, D. Mané, D. Fritz, D. Krishnan, F. B. Viégas, and M. Wattenberg. Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE TVCG*, 24(1):1–12, 2018. doi: 10.1109/TVCG.2017.2744878 2

[58] T. Wu, H. Zhu, M. Albayrak, A. Axon, A. Bertsch, W. Deng, Z. Ding, B. Guo, S. Gururaja, T.-S. Kuo, J. T. Liang, R. Liu, I. Mandal, J. Milbauer, X. Ni, N. Padmanabhan, S. Ramkumar, A. Sudjianto, J. Taylor, Y.-J. Tseng, P. Vaidos, Z. Wu, W. Wu, and C. Yang. Llms as workers in human-computational algorithms? replicating crowdsourcing pipelines with llms, 2023. doi: 10.48550/arXiv.2307.10168 9

[59] Y. Ye, J. Hao, Y. Hou, Z. Wang, S. Xiao, Y. Luo, and W. Zeng. Generative ai for visualization: State of the art and future directions. *Visual Informatics*, 8(2):43–66, 2024. doi: 10.1016/j.visinf.2024.04.003 1

[60] S.-J. Yen and Y.-S. Lee. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3, Part 1):5718–5727, 2009. doi: 10.1016/j.eswa.2008.06.108 6

[61] J. Zamfirescu-Pereira, R. Y. Wong, B. Hartmann, and Q. Yang. Why johnny can't prompt: How non-ai experts try (and fail) to design llm prompts. In *CHI '23*, article no. 437, 21 pages. ACM, New York, 2023. doi: 10.1145/3544548.3581388 1, 2, 3

[62] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert, 2020. doi: 10.48550/arXiv.1904.09675 2

[63] T. Zhang, F. Ladhak, E. Durmus, P. Liang, K. McKeown, and T. B. Hashimoto. Benchmarking Large Language Models for News Summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57, 2024. doi: 10.1162/tacl_a_00632 4

[64] W. Zhao, M. Peyrard, F. Liu, Y. Gao, C. M. Meyer, and S. Eger. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In K. Inui, J. Jiang, V. Ng, and X. Wan, eds., *EMNLP-IJCNLP*, pp. 563–578. ACL, Hong Kong, 2019. doi: 10.18653/v1/D19-1053 2

[65] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds., *Advances in Neural Information Processing Systems*, vol. 36, pp. 46595–46623. Curran Associates, Inc., 2023. doi: 10.48550/arXiv.2306.05685 2