RUBICON: Rubric-Based Evaluation of Domain-Specific Human AI Conversations

Param Biyani*
t-pbiyani@microsoft.com
Microsoft, India

Yasharth Bajpai*†
ybajpai@microsoft.com
Microsoft, India

Arjun Radhakrishna
arradha@microsoft.com
Microsoft, USA

Gustavo Soares
gustavo.soares@microsoft.com
Microsoft, USA

Sumit Gulwani
sumitg@microsoft.com
Microsoft, USA

ABSTRACT

Evaluating conversational assistants, such as GitHub Copilot Chat, poses a significant challenge for tool builders in the domain of Software Engineering. These assistants rely on language models and chat-based user experiences, rendering their evaluation with respect to the quality of the Human-AI conversations complicated. Existing general-purpose metrics for measuring conversational quality found in literature are inadequate for appraising domain-specific dialogues due to their lack of contextual sensitivity.

In this paper, we present RUBICON, a technique for evaluating domain-specific Human-AI conversations. RUBICON leverages large language models to generate candidate rubrics for assessing conversation quality and employs a selection process to choose the subset of rubrics based on their performance in scoring conversations. In our experiments, RUBICON effectively learns to differentiate conversation quality, achieving higher accuracy and yield rates than existing baselines.

CCS CONCEPTS

• **General and reference** → **Metrics**; **Evaluation**; • **Human-centered computing** → **HCI design and evaluation methods**; • **Software and its engineering** → *Collaboration in software development*; • **Computing methodologies** → *Natural language processing*.

KEYWORDS

Conversational AI, AI-assisted Programming, Conversation Evaluation, Human-AI interaction, Evaluation Metrics, User Satisfaction

ACM Reference Format:
Param Biyani, Yasharth Bajpai, Arjun Radhakrishna, Gustavo Soares, and Sumit Gulwani. 2024. RUBICON: Rubric-Based Evaluation of Domain-Specific Human AI Conversations. In *Proceedings of the 1st ACM International Conference on AI-Powered Software (AIware '24), July 15–16, 2024, Porto de Galinhas, Brazil.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3664646.3664778

*Equal Contribution; more junior author listed earlier
†Corresponding Author

AIware '24, July 15–16, 2024, Porto de Galinhas, Brazil

ACM ISBN 979-8-4007-0685-1/24/07
https://doi.org/10.1145/3664646.3664778

1 INTRODUCTION

AI assistants for software developers, such as Github Copilot [10], have started to adopt chat-based interfaces as one of the primary means for enabling interactive AI assistance for developers. However, evaluating the quality of these chat-based AI assistants has proven to be challenging due to the diverse range of tasks developers seek help with and the complex, multi-turn nature of Human-AI conversations, making it difficult to determine the success of these interactions. As a result, developers who build applications with chat interfaces often lack sufficient insight into the performance and quality of the interactions facilitated by their tools [22].

Recently, Lin et al. [16] introduced SPUR, a technique designed to estimate User Satisfaction for open-domain conversations conducted with Bing Copilot. SPUR leverages a large language model (LLM) to analyze past labelled conversations, capturing user satisfaction signals from upvoted conversations and user dissatisfaction signals from downvoted conversations. These signals are then utilized to generate a set of rubrics iteratively. A large language model can assess the quality of new conversations by scoring them against this learned set of rubrics. While user satisfaction rubrics learned by SPUR can be effective in assessing Human-AI conversations that contain user prompts indicating satisfaction or dissatisfaction, such as compliments or rephrased questions, they may be less effective in conversations where such signals are absent.

Figure 1 shows two sample conversations and a varied set of rubrics automatically learned from them by our technique. The rubrics highlight the wide gamut of signals, such as the repetition of information, progress towards task, adherence to user requests, provision of requisite information, etc., that determine the quality of conversations with AI assistants. Note that most of the generated rubrics are domain-aware, i.e., are specific to a debugging AI assistant. While rubrics like 'provides a code snippet as a solution' and 'does not provide new or specific debugging guidance' can be approximated by generic domain-agnostic ones like 'provides a solution with an example' and 'does not help the user' that may be generated by a generic technique like SPUR [16], the domain-aware ones are more contextual, definitive, as well as easier to evaluate. Additionally, the learned rubrics in Figure 1 go beyond the idea of user satisfaction, and instead address progress towards the completion of the task at hand. The aim of this paper is to tackle the task of automatically generating such high-quality, domain- and task-aware rubrics for task-oriented conversational AI assistants.

Developing rubrics that capture specific nuances from conversations necessitates more than just data. We employ a targeted

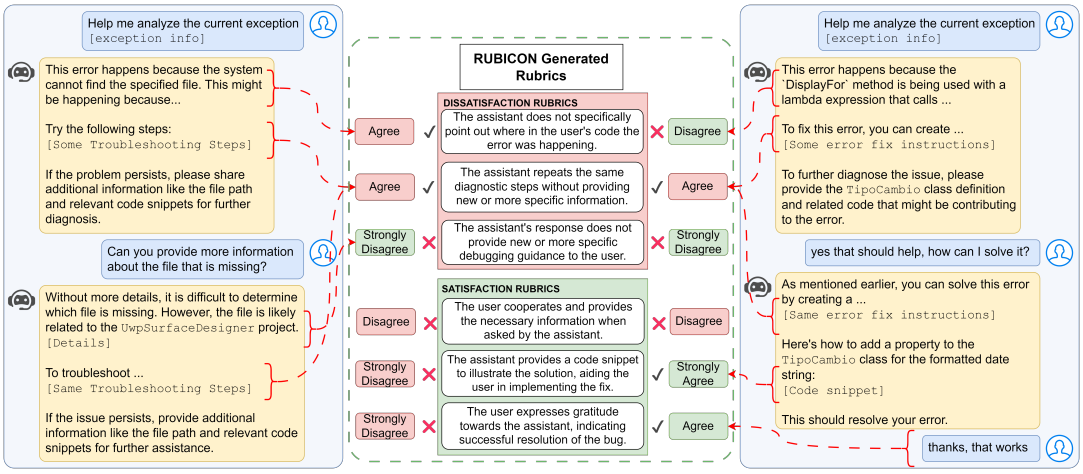


Figure 1: Two analogous conversations facilitated by the Debugger AI assistant evaluated against some representative rubrics. The conversation on the right was deemed better, while the one on the left was considered subpar.

and guided task-specific approach to rubric generation to learn these insights. Evaluating the quality of a task-oriented conversation presents challenges as it must satisfy multiple criteria, such as amiability, task progression, and collaboration, unlike a generic rubric generation technique like SPUR, which tends to produce overly broad rubrics. We aim for a more refined comprehension of conversations and their goals, enabling us to generate appropriate rubrics with higher specificity to the nature of the conversation.

In this paper, we propose RUBICON, a technique for evaluating domain-specific Human-AI conversations. RUBICON comprises three main components: (1) rubric set generation, (2) rubric selection, and (3) conversation evaluation.

In the first step, RUBICON follows the SPUR approach to identify signals in conversations labelled as *negative* and *positive*, generating a set of rubrics. However, RUBICON extends SPUR by instructing the model regarding domain-specific signals (DS) and conversation design principles (CDP) in the form of *Gricean maxims* [11, 33], which capture four dimensions of conversation effectiveness: quantity, quality, relevance, and manner. Additionally, while SPUR combines the rubrics produced in each batch of examples without evaluating the effectiveness of the final rubric set, RUBICON generates fresh rubrics for each batch to create a pool. In the second step, RUBICON, inspired by prompt optimization using multi-arm bandit selection in Pryzant et al. [25], iteratively selects rubrics based on two data-driven loss functions. This yields the final rubric set and a score threshold for classifying conversations as negative or positive. In the final step, RUBICON employs a Large Language Model (LLM) to grade and classify the conversation under test using the selected rubric set and a calculated threshold.

For our evaluation, we instantiated RUBICON to assess conversations between developers and a widely used chat-based assistant designed for C# developers. To create a labelled dataset, we collected 100 conversations where developers sought assistance from the chat-based assistant to resolve an exception while debugging their code. To power RUBICON, we utilized GPT-4[20], a state-of-the-art language model. GPT-4 was employed to generate rubrics, select the most relevant ones, and assess the conversations based on the learned rubric set and scoring threshold.

The rubrics generated using RUBICON exhibited superior performance, creating a 9x higher delta in the score of negative and positive conversations, compared to three alternative sets of rubrics: the original set from [16], a manually adapted set, and a set learned from our debugging dataset using SPUR. We are able to predict conversation labels with an almost perfect (> 0.9) precision for 84% of conversations on unlabelled data. Additionally, we conduct ablation studies to show the effect of each component of our technique. Overall, we present the following contributions:

- We propose RUBICON, a technique to automatically generate rubrics and score conversations for domain-specific AI Assistants;
- We compare the performance of rubrics generated by RUBICON and SPUR, using a dataset of 100 real conversations about debugging exceptions between developers and an AI assistant;
- We evaluate the impact and effectiveness of the different components within RUBICON.

2 RELATED WORK

Natural language conversations have become the staple interface for modern AI applications [7, 12, 29]. Despite the proliferation of traditional NLP metrics, their maturity hasn't kept pace in the era of Large Language Models (LLMs) [6, 15, 21]. Metrics like BLEU [21], RoBERTA [17], and Perplexity are often found short on being able to measure long-form conversations, particularly due to lack of ideal references and underlying intent. [18, 26, 34, 36].

In the broader sense, conversation quality analysis is not a new problem, and domains, particularly sales and marketing, have employed various ways of measuring the quality of conversation their end-users experience when interacting with agents [2]. User satisfaction estimation has been used as a proxy for evaluating conversational quality from a user experience-led approach [16]. User satisfaction through surveys and manual post-analysis of the conversation has been the most important of all other indicators. However, manual post-analysis of conversations has both privacy and

resource implications, thus difficult to carry out at scale [9, 14, 31]. Human annotations of conversations themselves are subjective and are known to be prone to bias [5, 8, 13, 19].

Given that data annotation and large-scale user study are tedious and resource consuming tasks, recent works have explored using language models to replace human subjects in answering evaluation related questions, proposing scoring systems built around judging conversations against some assertions [9, 16, 19, 27]. These natural language assertions are framed to ask specific questions about any conversations, detecting the presence of themes ranging from user experience - frustration, curiosity, satisfaction, impatience, etc., to more sophisticated/abstract themes like engagement, inconsistency, interestingness, understanding, etc.

This paper also discusses SPUR's [16] proposed automatic technique, which uses general open domain conversations with thumbs up/down user response signals to generate rubrics for user satisfaction estimation. However, SPUR's approach and other domain-agnostic satisfaction estimation policies like [35] are only designed for open-ended conversations. SPUR's assertions measure satisfaction signals, serving as a proxy for user satisfaction. However, for a complete conversational quality evaluation, the literature in HCI and other relevant fields focusing on Human-AI conversations advocate for a holistic approach to designing conversational AI systems, emphasizing naturalness, engagement, trust, empathy, and context awareness to create meaningful interactions that go beyond mere user satisfaction. Cathy [23] and Semnick [28] emphasize the importance of understanding user needs, expectations, and conversational norms to create engaging and satisfying experiences. A holistic conversational evaluation, especially for task-oriented interactions, should consider understanding expectations and the progress the interaction enables in that direction [3, 6].

Evaluating multi-turn conversations typically uses scoring over a Likert scale after a conversation is over [4, 9, 27, 31, 37]. Some works also explore using continuous scales to rate conversations per assertion [16]. However, we find that LLMs are usually inconsistent and biased when giving numeric ratings to natural language assertions. In this paper, we briefly discuss this trade-off and other than for comparison purposes, we only use the Likert scale throughout.

In this work, we also explore the problem of selecting an optimal subset of prompts from a larger set of prompts and compare our approach with some bandit selection methods [25]. The selection policy is optimised for two metrics related to binary classification: distance between the means of the distribution of the two classes and the percentage of the conversations that can be evaluated with some minimum confidence threshold. Since it is challenging to perform binary classification on dialogues due to their subjectivity, we only consider conversations classified with high confidence after the scoring. Similar techniques have been applied in problems like credit default prediction or fraud detection where the precision requirements of classification are very strict [30, 32].

3 TECHNIQUE

We propose RUBICON to estimate conversation quality for domain-specific conversational assistants. Given a set of conversations labelled as positive and negative interactions, we learn rubrics capturing Satisfaction (SAT) & Dissatisfaction (DSAT). Fig 2 outlines three major components in our technique - (1) Generating a diverse rubric set from conversation data; (2) Selecting an optimized set of rubrics for online evaluation; (3) Scoring conversations and predicting labels. We refer to (1) and (2) as the Generate & Select paradigm of learning rubrics, which are carried out in an offline setting.

The Generation step is inspired from SPUR to support learning of attributes and patterns of Satisfaction/Dissatisfaction from the available training conversations C_{train}. We adapt *Supervised Extraction* proposed in SPUR to include domain sensitization as well as conversation design insights, while the manipulated *Rubric Summarization* adheres to set standards of conversation while accumulating diverse assertions. Secondly, our novel selection policy optimizes the number of assertions required to help converge to rubric subsets of restricted sizes, making online evaluation feasible. The policy incorporates components to address concerns in a data-scarce environment while providing a better opportunity for labelling with high precision and coverage.

3.1 Rubrics and Conversation Quality

The core component of our system is a *rubric* r, i.e., a natural language assertion that denotes some property of a Human-AI assistant conversation. For example, "the user explicitly thanks the assistant" and "the information provided by the AI assistant was generic, and did not consider the user's current problem" are both rubrics. We classify rubrics into *satisfaction rubrics* and *dissatisfaction rubrics* (denoted SAT and DSAT) to say whether the rubric expresses a positive or a negative sentiment about the conversation. We use the symbols s and d to represent SAT and DSAT rubrics respectively.

Given a conversation c, we can *evaluate* c with respect to a rubric r by providing both of them to a language model and asking it to rate the match on a 5-point Likert scale. This 5 point score ranging from *Strongly Disagree* to *Strongly Agree* facilitates the LLM to lexically articulate response on a uniform scale. The response is then converted into a normalized score in the $[0, 10]$ range to remain consistent with SPUR. For ease of discussion, the score is negated (i.e., in the range $[-10, 0]$) for DSAT rubrics to ensure that a larger score always indicates a better conversation. We denote this normalized score as $\text{eval}(r, c)$.

Rather than using a generic prompt for evaluation, we provide domain & task-sensitized instructions to evaluate rubrics with respect to a conversation with an instruction-tuned LLM. Given input size limitations for the LLM and varying sizes of conversation to be scored, the evaluations are carried out in batches of up to 10 rubrics with SAT & DSAT scored separately.

For a set of Rubrics R_S comprising of SAT and DSAT rubrics (N of each), we define the conversation score, which we refer to as $NetSAT$. The score measures the net satisfaction by adding the sum of DSAT rubrics (negated) to the sum of SAT rubrics:

$$NetSAT(R_S, c) = \sum_{s \in R_S} \text{eval}(s, c) + \sum_{d \in R_S} \text{eval}(d, c)$$

3.2 Generation of Rubrics

The first component of the technique, as shown in Figure 2, is to generate a rubric pool from a given training set of conversations C, which is divided into positive and negative conversations C_+ and C_-. Inspired by SPUR [16], we use a two-phase strategy. First, we perform a *supervised extraction* to identify conversation quality

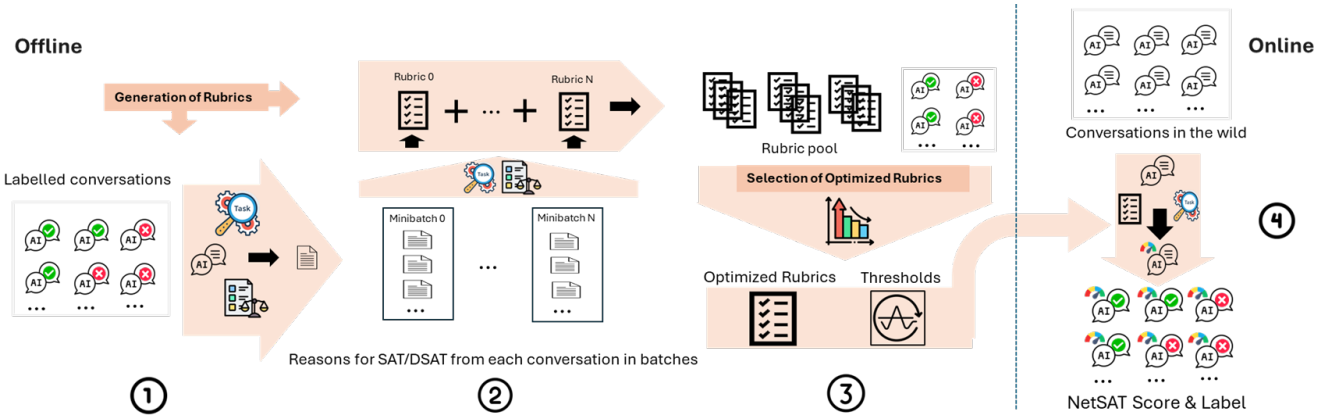


Figure 2: Representative Overview of the RUBICON.
(1) Extract task-specific patterns aligned with conversation design principles from each conversation. (2) Generate a large rubric pool from extracted reasons in the context of Gricean Maxims & the task. (3) Use data-driven loss to select optimal rubrics over conversation data (4) Score quality of each conversation over the learnt rubrics and thresholds

reasonings and patterns. Then, we summarize these reasonings in batches and aggregate them into a large rubric pool. Both the supervised extraction step and the rubric summarization steps are modified significantly from the versions in [16], and we compare the modifications experimentally in Section 4.

3.2.1 Supervised Extraction. Given a conversation $c \in C_{\text{train}}$, the supervised extraction step attempts to capture patterns $\hat{r}_i$ for particular satisfaction and dissatisfaction aspects of the conversation. If the conversation is annotated as positive in the ground-truth training set, we extract satisfaction patterns $\hat{s}_i$ (and dissatisfaction patterns $\hat{d}_i$ otherwise). Like in [16], we extract the top-k patterns from each conversation ($k = 3$ in experiments). Figure 3(a) presents the outline of the prompt used for supervised extraction.

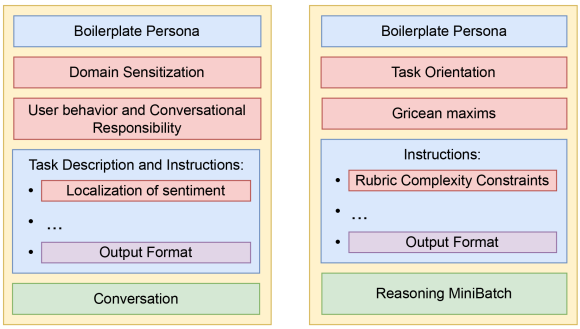
Example 3.1. Sample Reasonings for conversations from Fig. 1.
When the user asked for further clarification, the assistant provided a comprehensive and actionable plan in form of a detailed step-by-step guide on how to implement the solution, including code examples. (SAT, right conversation)
The assistant was unable to provide more specific information about the missing file, which was the user's direct question, due to lack of additional information from the user. (DSAT, left conversation)
These reasonings are picked from localized patterns in the interaction that could indicate why the user might be satisfied/dissatisfied with this conversation.

User behaviour and Conversational Responsibility(CDP). The field of classical conversation analysis assigns responsibility for conversation quality to both parties involved [23, 24]. In a conversational AI setting, it is easy to overlook the user's responsibility as a first-class participant and assign full responsibility of driving a positive conversation towards completing the task to the assistant and, in turn, the tool builder. In fact, prior works have done so [19, 27]. However, the prompt shown in Figure 3(a) presents a dedicated block for conversation responsibility. In our study, we explicitly ask the model to consider how both the user and the assistant take steps to progress or hinder the conversation towards task completion. Figure 1 (left) shows a redacted conversation where the user fails to provide relevant information requested by the assistant.

Reasonings may point out issues with the user experience of the tool and in the user's understanding of its capabilities. For example, based on the following user-related reasoning - *"The assistant was unable to access the file linked by the user, which hindered the progress of the conversation and likely caused frustration for the user."*, the builders of the debugging assistant we test in our experiments are able to identify two shortcomings in the design of the tool: (a) there were insufficient UX affordances for users to provide new files to the assistant and (b) the interface design sometimes leads to users confusing the debugging assistant with other IDE assistants and trying to ask non-debugging related questions of the assistant.

Domain Sensitization and Task Orientation (DS). We aim to produce rubrics that measure the quality of a conversation that is specifically about achieving a domain-specific goal (for example, debugging to fix an exception). That is, it does not matter if the assistant and user have an engaging conversation about the weather in Berlin if it is not goal-directed. Hence, it is important that the prompt to extract reasonings specifically mentions aspects of what the task is, expectations, and desired goals of the interaction. For example, in the debugging domain, our prompt contains specific expectations from the assistant described as *"designed to hold a conversation to understand, ask for more information, and investigate the bug, then provide solutions to aid the user in their debugging tasks."* This matches the intuition that for a task-oriented Human-Human conversation to be positive – it needs to both have good "conversational" qualities (e.g., participants know their roles, neither participant is frustrated or dominating, etc.) and task-specific qualities (e.g., progress made towards fixing the bug, etc.).

3.2.2 Rubric Summarization. Here, we consolidate the patterns generated in the Supervised Extraction step into SAT and DSAT rubrics. Unlike in [16], we do not aim to produce a small and usable set of rubrics updated at each step but instead to produce a large



(a) Supervised Extraction **(b) Rubric Summarization**

Figure 3: Rubric Generation Prompt Outlines.

pool of diverse rubrics. The next step will select a smaller optimized subset of good-quality rubrics from this pool.

This step incrementally generates new rubrics by processing a mini-batch of patterns identified in the step. Let the i^{th} minibatch be denoted as $\{\hat{r}_{m*(i-1)}, ..., \hat{r}_{m*i}\}$. For every minibatch, we prompt the LLM to generate a rubric set R_i based on its reasoning minibatch while also providing rubrics generated until the previous minibatch $\{R_1, ..R_{i-1}\}$. The goal is to have the generated rubric set R_i cover all the patterns in the minibatch *while* diversifying over and beyond the previously generated rubrics. The SAT and DSAT patterns are batched separately, with corresponding changes in the prompts to generate either SAT or DSAT rubrics, respectively. An important point of difference with SPUR is that we *do not update* a restricted set of N rubrics R_i at each turn and instead augment the rubric pool with fresh rubrics at each batch. A restricted set is then selected from a larger pool in a data-driven manner, instead of asking the LLM to update the rubrics based on reasonings in each batch.

After obtaining the pool of rubrics R consisting of all SAT/DSAT rubrics, we conduct a post-processing step to *semantically de-duplicate* the set, removing rubrics that exhibit semantic similarity to others. This is done to avoid any redundancy and bias in the final rubric set after selection. If r and r' are two well-performing semantically equivalent rubrics in the pool, they might likely end up in the final rubric set after selection (section 3.3), which may overpower and skew the resultant scores for the conversations using this set. To execute this, we use the reasoning capabilities of GPT-4 with a curated instruction prompt (details in [1]) to remove duplicate rubrics that cover similar reasonings, ideas or questions.

Gricean Maxims (CDP). We provide Grice's maxims as a framework of a cooperative principle for an effective conversation. These rational principles for improving conversation quality are encapsulated within four maxims: quantity, quality, relevance, and manner. These enable the summarizer to gauge different aspects of the conversation while generating the rubrics, particularly aspects like lapse on the part of any user, digression, confusion and lack of a proper structure in the conversation. The prompt for rubric summarization (Figure 3b) presents an ideal conversation as adhering to Grice's Maxims and stipulates that rubrics be crafted accordingly.

Example 3.2. The following rubrics below express the same idea:
Without CDP: *The assistant does not adhere to the user's specific request for the format or structure of the response.*
With CDP: *The assistant's responses are not aligned with the user's expectations.* The first rubric highlights the mismatch between user requests and AI responses without explaining why this is problematic or how it impacts the interaction. In contrast, the second rubric, aligned with the maxim of *relevance*, provides a more thorough and fundamental explanation of the communication breakdown.

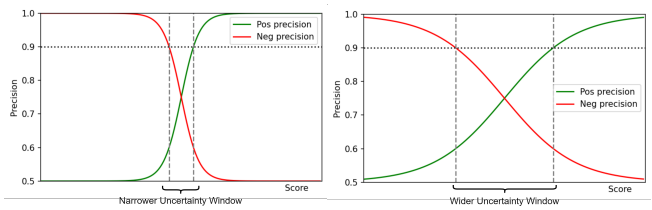


Figure 4: Positive & Negative Precision vs threshold for two hypothetical rubric sets. A narrower uncertainty window provides for larger ranges for high-confidence labels.

Domain Sensitization & Rubric Complexity (DS). The prompt incorporates instructions for the LLM to focus on the aspects that make a conversation positive/negative in the cognizance of the domain. For the Debugging use case, this includes progress towards the debugging task, localization of the bug, answering user queries, or providing an outright code snippet as the solution. Given that complex rubrics are difficult to reason about and often lead to non-determinism in response, even at low-temperature levels, we restrict the assertions to be short and clean. The instruction prompt limits the generation of complex rubrics by a natural language insight that simple assertive statements tend to have a single verb.

3.3 Selecting Optimized Rubrics

From the large pool of rubrics generated as per Section 3.2, we select a subset $\mathrm{R_S}$ of a practical size.

Correctness and correctness loss. Given a set of rubric R and a conversation c, we first define the score $\mathrm{Score}(\mathrm{R}, c)$ to be the sum $\sum_{r \in \mathrm{R}} \mathrm{Eval}_r(c)$ where $\mathrm{Eval}_r(c)$ is the normalized $[0, 10]$ Likert score for a single rubric and conversation described in Section 3.2. Intuitively, R should separate positive and negative conversations, i.e., positive conversations should be scored significantly higher than negative conversations. Hence, we define the *correctness loss* to be:

$$\mathrm{Loss_C}(\mathrm{R}) = \mathrm{mean}_{c \in \mathrm{C_+}}\ \mathrm{Score}(\mathrm{R}, c) - \mathrm{mean}_{c \in \mathrm{C_-}}\ \mathrm{Score}(\mathrm{R}, c)$$

Sharpness and Sharpness loss. We want the rubric-set R to be easily convertible into a binary classifier using a threshold. Correctness loss measures the average separation between positive and negative conversations but does not account for the *sharpness of the separation.* Given a threshold θ, we call a conversation c *positive* if $\mathrm{Score}(\mathrm{R}, c) > \theta$. The *positive precision* $\mathrm{Prec_+}(\mathrm{R}, \theta)$ (resp. *negative precision* $\mathrm{Prec_-}(\mathrm{R}, \theta)$) are the fractions of ground truth positive and negative conversations correctly labelled using threshold θ, respectively. There is a trade-off between $\mathrm{Prec_+}$ and $\mathrm{Prec_-}$: we can get very high positive precision by setting a high threshold (i.e., labelling almost all conversations negative) and vice-versa.

Figure 4 shows two hypothetical plots of threshold against positive and negative precision for two hypothetical rubric sets $\mathrm{R_1}$ and $\mathrm{R_2}$. The left plot shows a narrow range of candidate thresholds yielding high positive and negative precision, unlike the right plot. The *sharpness loss* quantifies this idea of sharpness of a rubric-set

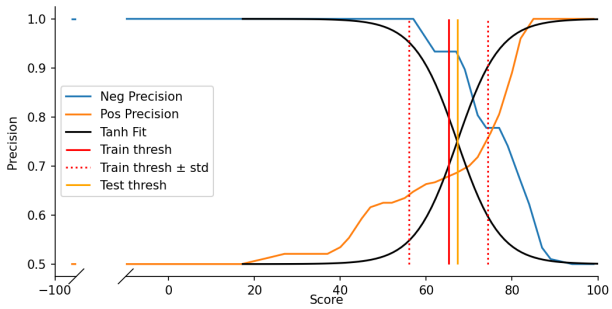


Figure 5: Positive & Negative Precision v Score(Threshold) plot, comparing the *Tanh* curves centred at the threshold and the corresponding precision curves post selection.

precisely: we fit a tanh function for both positive and negative precision curves while ensuring the mid-point θ_c for both curves is the same. Given a rubric-set R, let

$$\theta_c = \frac{1}{2} \cdot \left(\frac{\sum_{c \in C_+} \text{Score}(R, c)}{|C_+|} + \frac{\sum_{c \in C_-} \text{Score}(R, c)}{|C_-|} \right)$$

We define the sharpness loss as:

$$\begin{aligned} \text{Loss}_S(R) = \ & \text{Error}_\theta[\tanh(\theta - \theta_c), \text{Prec}_+(R, \theta)] \\ & + \text{Error}_\theta[\tanh(-(\theta - \theta_c)), \text{Prec}_-(R, \theta)] \end{aligned}$$

Note that we elide from the notation the scaling factors that ensure that both the precision curves and tanh are in the same range.

Iterative selection of rubrics. Algorithm 1 depicts an algorithm for selecting a rubric set R_S from a rubric pool R_P. It starts with an empty set R_S (line 1) and iteratively grows the set by adding the rubric r^* that minimizes the (weighted) sum of correctness and sharpness loss (lines 2- 4). The weight α (line 11) is a hyperparameter selected based on the best average performance in a 5-fold cross-validation over the training conversation data. In practice, rather than a combined budget n, we instead have separate n_{SAT} and n_{DSAT} budgets for the SAT & DSAT rubrics. We stop selecting either SAT or DSAT rubrics as soon as the respective budget is hit.

Algorithm 1 Selecting rubrics from rubric pool

Require: Conversations $C = C_+ \cup C_-$ partitioned into pos. and neg. subsets
Require: Rubric pool R_P, Budget n, and Sharpness loss weight $\alpha \in \mathbb{R}^{>0}$
Ensure: Selected rubric set R_S and Threshold θ

1: $R_S \leftarrow \emptyset$
2: **while** $|R_S| \neq n \wedge |R_P \setminus R_S| > 0$ **do**
3: $\quad r^* \leftarrow \arg\max_{r \in R_P \setminus R_S} \text{Loss}(R_S \cup \{r\}, C_+, C_-)$
4: $\quad R_S \leftarrow R_S \cup \{r^*\}$
5: **return** $\langle R_S, \text{GetThreshold}(R_S, C_+, C_-) \rangle$
6:
7: **procedure** Loss(R, C_+, C_-)
8: $\quad LC \leftarrow \text{mean}_{c \in C_+} \text{Score}(R, c) - \text{mean}_{c \in C_-} \text{Score}(R, c)$
9: $\quad \theta_c \leftarrow \text{GetThreshold}(R, C_+, C_-)$
10: $\quad LS \leftarrow \text{Loss}_S(R, \theta_c, C_+, C_-)$
11: $\quad$ **return** $LC + \alpha \cdot LS$
12:
13: **procedure** GetThreshold(R, C_+, C_-)
14: $\quad$ **return** $\frac{1}{2} \cdot (\text{mean}_{c \in C_+} \text{Score}(R, c) + \text{mean}_{c \in C_-} \text{Score}(R, c))$

4 EVALUATION

To evaluate RUBICON, we aim to answer the following questions:

- **RQ1:** How does the effectiveness of RUBICON compare to other rubric-backed baseline methods?
- **RQ2:** What is the extent of the impact of Domain Sensitization (DS) and Conversation Design Principles (CDP) instructions on the performance of RUBICON?
- **RQ3:** How does the effectiveness of the proposed selection policy compare to other baselines in selecting the final set of rubrics?

4.1 Data

Collection & Sanitization. The conversation data was sourced from a C# Debugger Copilot assistant deployed in an IDE at a large software company that was mined over 30 calendar days of deployment. All conversations have a unique Conversation ID and were collected anonymously. Of 127 conversations, after addressing logging & data corruption issues, we were left with 100 conversations. The deployed assistant is aware of the context around the exceptions like the error message and stack trace, and source code details like exception location and current file. The deployment was controlled and limited to maintain user access across the spectrum of proficiency and experience in software development in C#.
Annotation. The filtered set of 100 conversations are then annotated for ground truth of being *Positive* or *Negative* in reference to concrete aid to the user with the debugging task. Two authors with experience of over 2 & 4 years of professional software development in the industry annotate the data in the binary classification task. In the initial phase, the first 20 conversations were annotated together while discussing themes for annotation. Post these insights, both engineers annotated another set of 20 conversations independently, which were then tested for Inter-Rater Reliability (IRR). We found that the annotators agreed on 17 conversations, giving us an IRR of 0.85, which is an 'almost perfect' agreement. We split the remaining set between both the raters to label independently while communicating in case of doubts or need for discussion.
Split. Given the limited data availability, we carry out a 50:50 train-test split on the filtered conversation data. Both classification labels are homogeneously represented in the splits.

4.2 Metrics

Accuracy, Precision, Recall and F1 Score are calculated on the test set C_{test} based on threshold θ learned from the C_{train} for each final set of rubrics. We define two additional metrics in our evaluation - **ΔNetSAT** score measures how well a rubric set separates the clusters of positive and negative conversations across the score axis. The metric is the difference between the means of the NetSAT score of positive and negative conversations, i.e. $\Delta NetSAT = \overline{NetSAT}_+ - \overline{NetSAT}_-$. The higher the $\Delta NetSAT$, the higher the separation between the two distributions, meaning that the rubric differentiates between positive and negative conversations better. *Note:* In our experiments, which we describe in Section 4.5, we observe a −0.69 Pearson correlation coefficient between the SAT and DSAT scores across conversations on our final rubric set generated by RUBICON, indicating as scores on SAT increase, the scores on DSAT tends to

decrease, and the difference is actually representative of the separation, making it unlikely for a positive conversation and a negative conversation to have similar scores, in general.

Yield Rate measures what fraction of the test set C_{test} can be labelled with a certain precision. Classifying conversations as distinctly positive or negative is a difficult and subjective task, even for humans. To be confident about our generated labels, we only consider conversations which can be classified with a minimum acceptable precision. $YieldRate@P$ is the percentage of conversations that fall above the P precision threshold. For the purpose of our experiments, we fix P to be 0.9 and refer to the metric as $YieldRate@90$. We calculate the Prec_- and Prec_+ as a function of threshold θ. $\text{Prec}_-(\theta)$ is the percentage of negative labels classified with a threshold θ, similarly for Prec_+. By setting a 0.9 precision threshold, we divide the scores into three windows. The first window with $\text{Prec}_- > 0.9$, the second with $\text{Prec}_- < 0.9$ and $\text{Prec}_+ < 0.9$, which we call the uncertainty window, and the third with $\text{Prec}_+ > 0.9$. Only the conversations with scores outside of the uncertainty window are considered, and the yield rate measures the ratio of these conversations. Figure 4 shows the positive and negative precision curves based on a 0.9 precision cutoff. *Note*: A high overall precision does not translate to a high Yield Rate, as the latter depends on the spread of the score and the uncertainty window. One may have high precision values for more data with low confidence in the uncertainty interval; however, this performance does not translate to practical deployment scenarios.

4.3 Baselines

We compare RUBICON against the other baselines for the end-to-end task of generating scoreable rubric sets. We further compare dedicated baselines for the Rubric Selection policy separately.

BING. Authors of SPUR investigate, generate and share rubrics for the Bing Copilot [16] generated with their technique. These SAT/DSAT rubrics were learned from training data for open-domain question-answering of Bing Copilot.

BING (Domain Adapted). For this baseline, we manually adapted the *BING* rubrics to be more inclusive of our domain and expectations. This was done by an independent software engineer who works on the Visual Studio Copilot with over 6 years of professional development experience. We ask them to *completely* replace rubrics that did not apply to the use case with *domain-aware* rubrics for both SAT/DSAT based on their intuition and understanding. The BING_{DA} rubric set (provided in Supplementary [1]) finally received had 3 replacements in SAT and 4 replacements in DSAT.

SPUR. We re-implemented the SPUR technique [16] using the prompts and strategies provided in the paper. We ran this pipeline on our training data to get the final set of rubrics for evaluation.

4.3.1 Selection Policy. Given a rubric pool and some validation data, the selection policy chooses a subset of rubrics optimised for performance on the validation dataset.

UCB Bandits. This policy treats rubric selection as a bandit selection problem, with UCB used to select prompts based on performance [25]. We define the reward score of a rubric as the sum of its scores over a batch of conversations. Reinforcement Learning methods balance exploration and exploitation, assuming prompt evaluations are limited and costly. This may not necessarily be the case in domain-specific conversations where data may be scarce, rendering evaluating all rubrics over a smaller data size manageable.

Brute Force. We also compare with a brute force approach where the score of each rubric is defined the same as in UCB Bandits. This assumes that complete evaluation over the rubric and conversation is manageable, akin to RUBICON.

4.4 Experimental Setup

Throughout our experiments, we use GPT-4 to power RUBICON and SPUR. Two authors of the paper created the proposed prompt instructions as discussed in Section 3 (exact details in [1]) For all these configurations, we maintain a consistent selection policy for the final rubrics, as proposed in Algo. 1, with both SAT & DSAT rubric set to $N = 10$. For *RQ1*, we use the final rubric sets obtained from all baselines and our technique, which are then analyzed for the rating and classification task. To evaluate *RQ2*, we conduct an ablation study to differentiate the contribution of each component in the prompts. We establish four configurations by systematically excluding instructions related to DS, CDP, and both from the final proposed prompts. Finally, to answer *RQ3* - for all selection policies, we fix the rubric pool to the one acquired on executing the augmentation as described in 3.2. Thereafter, the ability of these techniques to select an optimized rubric set is studied.

4.5 Results

4.5.1 RQ1: How does the effectiveness of RUBICON compare to the other rubric-backed baseline methods? Table 1 summarises the results. The improvement of RUBICON over other methods is readily apparent in $\Delta NetSAT$. This highlights the effectiveness of our tool in distancing positive conversations from negative ones in terms of $NetSAT$ which creates the highest separation as compared to other baselines. SPUR, in particular, is unable to generalize over our data to create differences in scores of positive and negative conversations. Furthermore, we observed that RUBICON also outperforms in terms of $YR@90$, scoring 20 points (abs.) higher than the next best. This indicates that our tool offers a high precision (>0.9) in classifying 84% of conversations. SPUR only classifies 28% of conversations with this level of precision. $BING_{DA}$ achieves a 6.2% precision score improvement over *BING* just through the substitution of a few rubrics with domain-specific ones, underscoring the significance of domain specificity. While $BING_{DA}$ achieved a slightly higher precision score, it classified far fewer conversations (64%) with the minimum acceptable level of precision. Our results suggest that RUBICON rubrics perform better than SPUR rubrics and rubrics manually written by experts.

Table 1: Rubric performance across different techniques. $\Delta NS = \Delta NetSAT$, $YR@90 = YieldRate@90\%$, $A = Accuracy$, $P = Precision$, $R = Recall$, $F1 = F1score$

Technique	ΔNS	$YR@90$	A	P	R	$F1$
BING	11.0	58.0	70.0	64.7	88.0	74.6
$BING_{DA}$	11.3	64.0	76.0	**70.9**	88.0	78.6
SPUR	3.0	28.0	58.0	55.9	76.0	64.4
RUBICON	**27.4**	**84.0**	**76.0**	68.6	**96.0**	**80.0**

4.5.2 RQ2: What is the extent of the impact of Domain Sensitization (DS) and Conversation Design Principles (CDP) instructions on the performance of RUBICON?. Table 2 showcases the various configurations for the augmentation step ablation. We observe that domain sensitization is crucial in recognizing patterns and significantly improves metrics across the board. Notably, the removal of both DS and CDP resulted in the steepest decline, highlighting their pivotal role. This also suggests that DS and CDP are complementary, optimizing the technique's performance when combined. The prompts are engineered to elicit domain-specific responses, identify points of satisfaction/progress signals, and clarify the joint responsibility of the user and assistant in accomplishing the task objectives. CDP establishes a framework for the desired structure and content of an ideal conversation, guiding the generation of rubrics that encapsulate singular ideas that check off the most relevant conversational attributes. A higher recall in the absence of DS could indicate over-generalization, increasing both true and false positives. Thus, DS helps recognize more nuanced and domain-specific patterns, enhancing selectivity for positive instances.

Table 2: Ablation study of Rubric Augmentation

Technique	ΔNS	$YR@90$	A	P	R	$F1$
RUBICON	**27.4**	**84.0**	**76.0**	**68.6**	96.0	**80.0**
$-CDP$	23.7	72.0	64.0	58.5	96.0	72.7
$-DS$	17.9	66.0	52.0	51.0	**100.0**	67.6
$-CDP-DS$	15.7	62.0	50.0	50.0	**100.0**	66.67

4.5.3 RQ3: How does the effectiveness of the proposed selection policy compare to that of other baselines in selecting the final set of rubrics? The experimental results (Table 3), reveal several key insights about the performance of our selection policy. Our policy outperforms baseline methods in terms of $\Delta NetSAT$ score and $YieldRate@90\%$ precision with clear contributions from both components of loss as hypothesized. These results suggest that our selection policy is particularly effective in separating positive and negative conversations while also being able to confidently label a larger proportion of conversations compared to baselines. Interestingly, though *BruteForce* and *CorrectnessLoss* consider all possibilities, the objective allows the latter to have better accuracy and separation while maintaining a similar confidence. Overall, the Generate & Select framework allows us to generate a diverse set of rubrics, followed by a selection step that can be optimised to various evaluation needs. Our implementation prioritizes both surety and correctness of the classifications, and this focus is evident in the selection policy construction. The last row suggests that *SharpnessLoss* enables us to classify confidently with high yield rates without losing on performance as compared to other baselines with respect to the $NetSAT$ separation, counterbalancing effects of overfitting tendencies from the other loss component.

Table 3: Comparative analysis of Selection policies.

Technique	ΔNS	$YR@90$	A	P	R	$F1$
UCB	23.6	68.0	72.0	65.7	92.0	76.7
BruteForce	24.2	74.0	74.0	67.7	92.0	78.0
CorrectnessLoss	**29.7**	74.0	**76.0**	**69.7**	92.0	79.3
RUBICON	27.4	**84.0**	**76.0**	68.6	**96.0**	**80.0**

5 THREATS TO VALIDITY

Internal Validity The ground truth labels for the binary classification of conversations into positive or negative were manually assigned. Despite our high inter-annotator agreement, this process is inherently subjective and prone to errors or inconsistencies, which might impact the rubric learning process.

External Validity The data used in our experiments were collected over a month from a C# debugger copilot assistant deployed in an IDE at a large software company. The limitation in the dataset's diversity in terms of the number of conversations we could collect due to ethical limitations might lead to biases in the results. Additionally, our study is conducted in the context of software debugging tasks, making the results domain-specific to software engineering (SE). We use debugging specific sensitization in our experiments, and the resultant rubrics and evaluator may not directly translate to other domains or tasks. While the concept and technique are transferable, generalizing the results to other areas within SE or other domains might require additional validation.

Construct Validity Our research leverages automated scoring of rubrics based on instruction tuned LLM. However, the performance of the automated scoring is optimized with respect to the task, its accuracy is not directly evaluated. Changes to the underlying scoring model could influence our results. The use of the Likert scale and its arithmetic manipulations makes several implied assumptions. Converting the ordinal Likert scale to a $[0, 10]$ scale assumes the difference between 'Neutral' and 'Disagree' to be the same as 'Agree' and 'Strongly Agree'. We also club 'Not Applicable' with 'Neutral' and sum evaluations from different rubrics with equal weightage for the final $NetSAT$ score, despite varying importance of rubrics for different users or conversations. Future work will explore different options for calculating the $NetSAT$ score.

6 CONCLUSION

We introduced RUBICON, a novel methodology for automated evaluation of AI-assisted debugging conversations. We showed that by incorporating domain knowledge and conversational design principles, we can considerably enhance the quality of generated rubrics. Furthermore, we demonstrated that our proposed selection policy effectively differentiates between good and bad conversations and surpasses baseline methods. A domain-adapted Likert scale scoring system also proved effective in scoring rubrics. We share the prompts used in the pipeline as well as the rubrics generated at [1].

We implement a strategy where extracting rubrics and selecting a rubric set for online evaluations are two distinct but consequent problems within the Generate & Select framework. Our technique optimises different steps for handling multi-turn task-oriented conversations, aiming to provide a powerful yet easy-to-implement technique for evaluating AI assistants and chatbots.

RUBICON has been successfully deployed in a popular IDE at a large software company to monitor two Software Engineering AI Assistant Copilots, with promising results. The insights gained from this study open new avenues for future research, emphasizing that the synergistic application of AI and human expertise can lead to robust and effective tools for evaluating and improving conversational debugging experiences.

REFERENCES

[1] Param Biyani, Yasharth Bajpai, Arjun Radhakrishna, Gustavo Soares, and Sumit Gulwani. 2024. Supplementary Material for RUBICON. https://aka.ms/rubicon-supplementary

[2] Praveen Kumar Bodigutla, Lazaros Polymenakos, and Spyros Matsoukas. 2019. Multi-domain Conversation Quality Evaluation via User Satisfaction Estimation. arXiv:1911.08567 [cs.LG]

[3] Bhavya Chopra, Yasharth Bajpai, Param Biyani, Gustavo Soares, Arjun Radhakrishna, Chris Parnin, and Sumit Gulwani. 2024. Exploring Interaction Patterns for Debugging: Enhancing Conversational Capabilities of AI-assistants. https://api.semanticscholar.org/CorpusID:267617149

[4] Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2017. Referenceless Quality Estimation for Natural Language Generation. arXiv:1708.01759 [cs.CL]

[5] Wieland Eckert, Esther Levin, and R. Pieraccini. 1998. User modeling for spoken dialogue system evaluation. , 80 - 87 pages. https://doi.org/10.1109/ASRU.1997.658991

[6] Ryan Fellows, H. Ihshaish, Steve Battle, Ciaran Haines, Peter Mayhew, and J. Ignacio Deza. 2021. Task-oriented Dialogue Systems: performance vs. quality-optima, a review. https://api.semanticscholar.org/CorpusID:245353758

[7] Nah Fiona Fui-Hoon, Zheng Ruilin, Cai Jingyuan, Siau Keng, and Chen Langtao. 2023. Generative AI and ChatGPT: Applications, challenges, and AI-human collaboration. *Journal of Information Technology Case and Application Research* 25, 3 (2023), 277–304. https://doi.org/10.1080/15228053.2023.2233814

[8] Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: core tasks, applications and evaluation. *J. Artif. Int. Res.* 61, 1 (jan 2018), 65–170.

[9] Marco Gerosa, Bianca Trinkenreich, Igor Steinmacher, and Anita Sarma. 2024. Can AI serve as a substitute for human subjects in software engineering research? *Automated Software Engg.* 31, 1 (jan 2024), 12 pages. https://doi.org/10.1007/s10515-023-00409-6

[10] GitHub. 2023. GitHub Copilot. https://github.com/features/copilot

[11] Herbert Paul Grice. 1989. *Studies in the Way of Words.* Harvard University Press, Cambridge.

[12] Dinesh Kalla, Nathan Smith, Dr. Sivaraju Kuraku, and Fnu Samaah. 2023. Study and Analysis of Chat GPT and its Impact on Different Fields of Study. https://doi.org/10.5281/zenodo.10250455

[13] Ilya Kulikov, Alexander H. Miller, Kyunghyun Cho, and Jason Weston. 2018. Importance of a Search Strategy in Neural Dialogue Modelling. https://api.semanticscholar.org/CorpusID:53297919

[14] Margaret Li, Jason Weston, and Stephen Roller. 2019. ACUTE-EVAL: Improved Dialogue Evaluation with Optimized Questions and Multi-turn Comparisons. https://api.semanticscholar.org/CorpusID:202538657

[15] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out.* Association for Computational Linguistics, Barcelona, Spain, 74–81. https://aclanthology.org/W04-1013

[16] Ying-Chun Lin, Jennifer Neville, Jack W. Stokes, Longqi Yang, Tara Safavi, Mengting Wan, Scott Counts, Siddharth Suri, Reid Andersen, Xiaofeng Xu, Deepak Gupta, Sujay Kumar Jauhar, Xia Song, Georg Buscher, Saurabh Tiwary, Brent Hecht, and Jaime Teevan. 2024. Interpretable User Satisfaction Estimation for Conversational Systems with Large Language Models. arXiv:2403.12388 [cs.IR]

[17] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs.CL]

[18] Clara Meister and Ryan Cotterell. 2021. Language Model Evaluation Beyond Perplexity. arXiv:2106.00085 [cs.CL]

[19] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2018. RankME: Reliable Human Ratings for Natural Language Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Marilyn Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, New Orleans, Louisiana, 72–78. https://doi.org/10.18653/v1/N18-2012

[20] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[21] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Pierre Isabelle, Eugene Charniak, and Dekang Lin (Eds.). Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, 311–318. https://doi.org/10.3115/1073083.1073135

[22] Chris Parnin, Gustavo Soares, Rahul Pandita, Sumit Gulwani, Jessica Rich, and Austin Z. Henley. 2023. Building Your Own Product Copilot: Challenges, Opportunities, and Needs. arXiv:2312.14231 [cs.SE]

[23] Cathy Pearl. 2016. *Designing Voice User Interfaces: Principles of Conversational Experiences* (1st ed.). O'Reilly Media, Inc., Sebastopol, CA.

[24] Diana Perez-Marin and Ismael Pascual-Nieto. 2011. *Conversational Agents and Natural Language Interaction: Techniques and Effective Practices.* Information Science Reference - Imprint of: IGI Publishing, Hershey, PA.

[25] Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic Prompt Optimization with "Gradient Descent" and Beam Search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 7957–7968. https://doi.org/10.18653/v1/2023.emnlp-main.494

[26] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[27] Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? How controllable attributes affect human judgments. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 1702–1723. https://doi.org/10.18653/v1/N19-1170

[28] Olga Seminck. 2023. Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots by Michael McTear. *Computational Linguistics* 49, 1 (March 2023), 257–259. https://doi.org/10.1162/coli_r_00470

[29] Sakib Shahriar and Kadhim Hayawi. 2023. Let's Have a Chat! A Conversation with ChatGPT: Technology, Applications, and Limitations. *Artificial Intelligence and Applications* 2, 1 (June 2023), 11–20. https://doi.org/10.47852/bonviewaia3202939

[30] Usneek Singh, Piyush Arora, Shamika Ganesan, Mohit Kumar, Siddhant Kulkarni, and Salil Rajeev Joshi. 2024. Comparative Analysis of Transformers for Modeling Tabular Data: A Casestudy using Industry Scale Dataset. In *Proceedings of the 7th Joint International Conference on Data Science & Management of Data (11th ACM IKDD CODS and 29th COMAD)* (Bangalore, India) *(CODS-COMAD '24)*. Association for Computing Machinery, New York, NY, USA, 449–453. https://doi.org/10.1145/3632410.3632456

[31] Eric Smith, Orion Hsu, Rebecca Qian, Stephen Roller, Y-Lan Boureau, and Jason Weston. 2022. Human Evaluation of Conversations is an Open Problem: comparing the sensitivity of various methods for evaluating dialogue agents. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, Bing Liu, Alexandros Papangelis, Stefan Ultes, Abhinav Rastogi, Yun-Nung Chen, Georgios Spithourakis, Elnaz Nouri, and Weiyan Shi (Eds.). Association for Computational Linguistics, Dublin, Ireland, 77–97. https://doi.org/10.18653/v1/2022.nlp4convai-1.8

[32] Abhinav Srivastava, Amlan Kundu, Shamik Sural, and Arun Majumdar. 2008. Credit Card Fraud Detection Using Hidden Markov Model. *IEEE Transactions on Dependable and Secure Computing* 5, 1 (2008), 37–48. https://doi.org/10.1109/TDSC.2007.70228

[33] P. F. Strawson. 2011. Review of Paul Grice, Studies in the Way of Words. https://doi.org/10.1093/acprof:oso/9780199587292.003.0015 arXiv:https://academic.oup.com/book/0/chapter/141927393/chapter-ag-pdf/45432827/book_2039_section_141927393.ag.pdf

[34] Elior Sulem, Omri Abend, and Ari Rappoport. 2018. BLEU is Not Suitable for the Evaluation of Text Simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 738–744. https://doi.org/10.18653/v1/D18-1081

[35] Stefan Ultes, Paweł Budzianowski, Iñigo Casanueva, Nikola Mrkšić, Lina Rojas-Barahona, Pei-Hao Su, Tsung-Hsien Wen, Milica Gašić, and Steve Young. 2017. Domain-Independent User Satisfaction Reward Estimation for Dialogue Policy Learning. In *Proc. Interspeech 2017.* Interspeech, Stockholm, Sweden, 1721–1725. https://doi.org/10.21437/Interspeech.2017-1032

[36] Yequan Wang, Jiawen Deng, Aixin Sun, and Xuying Meng. 2023. Perplexity from PLM Is Unreliable for Evaluating Text Quality. arXiv:2210.05892 [cs.CL]

[37] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too?. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, Melbourne, Australia, 2204–2213. https://doi.org/10.18653/v1/P18-1205

Received 2024-04-05; accepted 2024-05-04