ChatChecker: A Framework for Dialogue System Testing Through Non-cooperative User Simulation

Roman Mayr
Technicial University of Munich
& University of Cambridge
roman.mayr@tum.de

Michel Schimpf
University of Cambridge
ms2957@cam.ac.uk

Thomas Bohné
University of Cambridge
tbm35@cam.ac.uk

Abstract

While modern dialogue systems heavily rely on large language models (LLMs), their implementation often goes beyond pure LLM interaction. Developers integrate multiple LLMs, external tools, and databases. Therefore, assessment of the underlying LLM alone does not suffice and the dialogue systems must be tested and evaluated as a whole. However, this remains a major challenge. With most previous work focusing on turn-level analysis, less attention has been paid to integrated dialogue-level quality assurance. To address this, we present `ChatChecker`[1], a framework for automated evaluation and testing of complex dialogue systems. `ChatChecker` uses LLMs to simulate diverse multi-turn user interactions, identify dialogue breakdowns, and evaluate quality. Compared to previous approaches, our design reduces setup effort and is generalizable as it does not require reference dialogues and is decoupled from the implementation of the target dialogue system. We improve breakdown detection performance over a prior LLM-based approach by including an error taxonomy in the prompt. Additionally, we propose a novel non-cooperative user simulator based on challenging personas that uncovers weaknesses in target dialogue systems more effectively. Through this, `ChatChecker` contributes to thorough and scalable testing of multi-turn interactions.

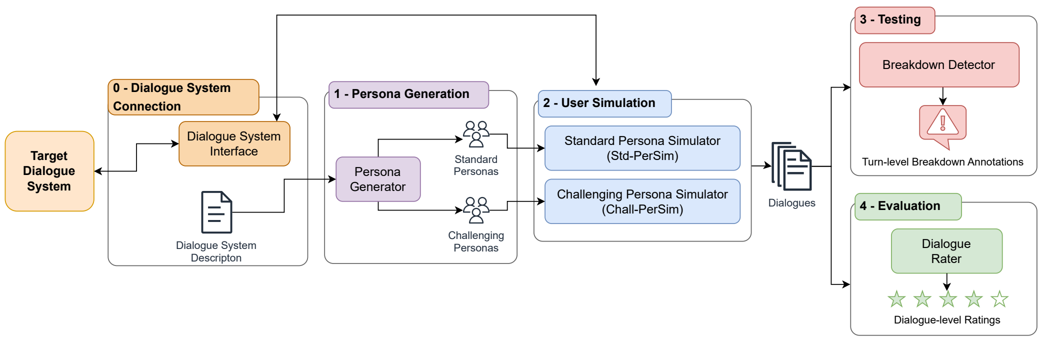

Figure 1: Overview of the ChatChecker framework. After connecting a target dialogue system (0), the framework generates diverse user personas (1), including both standard and challenging profiles. These personas are then used for user simulation (2) via two modes: the Standard Persona Simulator and the Challenging Persona Simulator. The resulting dialogues are processed by a breakdown detector (3) to identify turn-level failures and by a dialogue rater (4) to produce dialogue-level quality ratings. This pipeline enables thorough, scalable, and automated evaluation of dialogue systems.

[1] `https://github.com/llm-psychology-group/chat-checker`

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling.

1 Introduction

Dialogue-based human-computer interaction has become increasingly widespread with the rise of LLMs through systems such as ChatGPT [24]. Beyond the chat interfaces of the LLM providers, dialogue systems are now deployed across various use cases - from task-oriented customer service, over mental health treatment [10] to conversations with virtual companions like Replika [20].

With millions of users and applications in critical domains such as healthcare and finance, ensuring the robustness and reliability of dialogue systems is crucial. However, testing and evaluating these systems remain persistent challenges [33, 27].

Dialogue systems, also called chatbots, interact with users through natural language conversation [15]. These systems can be broadly categorized into task-oriented dialogue systems (ToDs) designed for specific tasks like booking hotels, and conversational dialogue systems for open-domain, natural conversations [5]. Modern dialogue systems increasingly rely on LLMs [31, 32], which blur the traditional distinction between these categories [15].

Evaluating dialogue systems encompasses both human and automatic assessment of system performance. While human evaluation through crowd-sourcing platforms is common, its cost and time requirements drive the need for automated methods [5]. Traditional reference-based metrics like BLEU [25] have proven inadequate for dialogue evaluation [18]. Recent research leverages LLMs for automated rating, with approaches such as G-EVAL [19] showing improved correlations with human judgments. Mendonça et al. [23] demonstrated that LLM-based ratings achieve state-of-the-art performance for multilingual dialogue evaluation.

In this work, we focus on the elicitation of undesired interactions for testing dialogue systems. To this end, it is crucial to detect dialogue breakdowns, which occur when a conversation becomes difficult to continue smoothly [21, 11]. The Dialogue Breakdown Detection Challenge (DBDC) [12] provides datasets labeled as not a breakdown (NB), possible breakdown (PB), or breakdown (B). For a more fine-grained assessment, Higashinaka et al. [14] developed a comprehensive error taxonomy distinguishing 17 conversational error types across utterance, response, context, and society levels. Ghassel et al. [8] reported that `GPT-4` achieved competitive results in breakdown detection, establishing LLMs as effective tools for this task.

User simulators automate the generation of dialogue interactions for testing and evaluation. While early approaches like ABUS [28] focused on semantic-level simulation, recent methods directly generate user utterances using LLMs [30, 3, 31]. Although these existing LLM-based simulators demonstrate the potential of leveraging LLMs for user simulation, they have significant limitations:

- *Dependence on existing datasets*: They typically rely on existing reference dialogues for few-shot samples and structured goals. Early-stage systems or new iterations often lack sufficient interaction data for this.
- *Tight coupling with the target dialogue system*: For instance, Terragni et al. [30] integrated their simulator in the ConvLab-2 framework [34] for dialogue systems on the Multi-Domain Wizard-of-Oz dataset (MultiWOZ) [1, 6] benchmark.
- *Focus on cooperative users*: With reference dialogues extracted from human samples and LLMs instructed to attempt to achieve a given in-domain task, existing simulators mainly simulate cooperative user behavior.

To address the identified limitations, we introduce `ChatChecker`, a fully automated framework for dialogue system testing and evaluation. **Our key contributions are:**

- A *Breakdown Detector* that improves detection performance over the prior LLM-based approach of Ghassel et al. [8] and integrates error type classification.
- A novel non-cooperative simulation strategy that exposes system weaknesses more effectively during testing.
- An integrated framework combining user simulation, breakdown detection, and dialogue rating for automated testing.

2 ChatChecker Framework

`ChatChecker` employs a modular architecture consisting of four main components that enable end-to-end dialogue system evaluation (Figure 1). First, the *Persona Generator* creates cooperative standard users and non-cooperative challenging users based on the dialogue system description. Second, the *User Simulation* module generates diverse conversations based on the different user personas. Third, the *Breakdown Detector* checks each system turn to detect breakdowns and classify them using an extended error taxonomy. Finally, the *Dialogue Rater* evaluates conversations across multiple quality dimensions, providing interpretable feedback.

The framework prioritizes practical deployment. Integration requires only a YAML configuration file describing the target dialogue system (see Appendix 1 for an example) and a chatbot client for connection based on a simple Python interface. All predictions include step-by-step reasoning for better transparency. Temperature settings are controlled to enable reproducible results across evaluations.

2.1 Dialogue Breakdown Detection

Inspired by Ghassel et al. [8], we apply an LLM-based breakdown detection approach. To this end, we use `GPT-4o` (`gpt-4o-2024-08-06`, temperature 0) with structured outputs. Our method evaluates each system response using an *extended error taxonomy*, adapted from Higashinaka et al. [14] and expanded to cover task-oriented dialogue systems.

While the original taxonomy focuses on general conversational issues, task-oriented dialogues have additional failure points. We therefore introduce nine additional error types, grouped into three categories:

- *Task-success impairments*: task performance failure, information update failure, clarification failure
- *Efficiency issues*: redundancy, lack of brevity, lack of clarity
- *Domain handling problems*: failure to recognize, communicate, or resolve out-of-domain requests

The LLM is prompted to assess system turns against this taxonomy and to provide a reasoning for each evaluation. Each response receives a score between 0 (complete breakdown) and 1 (seamless flow). When a breakdown is detected, the model lists all relevant error types, supporting a granular understanding of system behavior. Our full prompt template is available in the Appendix 3.

2.2 Dialogue Rating

Quality evaluation spans multiple dimensions customized to the dialogue system type, in addition to an overall rating. For task-oriented systems, we prioritize task success and efficiency alongside conversational appropriateness and naturalness. We selected appropriateness, naturalness, coherence, likability, and informativeness from the plethora of possible dimensions for conversational systems [22]. We prompt `GPT-4o` (`gpt-4o-2024-08-06`, temperature 0) to rate each dimension on a 1-5 scale with instructions for strict judgment (see Appendix 5) to prevent the overrating tendency observed in preliminary experiments. The rating process includes step-by-step reasoning to support developer interpretation and leverage chain-of-thought prompting [16].

2.3 User Simulation

Our simulation approach (see Appendix 9) prompts `GPT-4o` (`gpt-4o-2024-08-06`, temperature 1) to iteratively generate the next user turn using either standard users or challenging non-cooperative personas. Developers can define personas themselves or use our *Persona Generator* to generate fitting user profiles that include demographic details, personality traits following the Big Five model [9], interaction styles, and specific dialogue tasks.

For standard personas, the *Persona Generator* focuses on typical usage scenarios, while prompts for challenging personas emphasize edge cases that push chatbot limits while maintaining human-like behavior (see subsubsection A.2.3 for details). This dual strategy helps developers evaluate both expected performance and robustness in adverse interactions.

3 Results

As there is no comparable openly available comprehensive framework, we cannot directly compare `ChatChecker` with other systems. Hence, in the following, we validate all of our subsystems independently. The evaluation code is publicly available on GitHub[2].

3.1 Breakdown Identification Results

We used the data provided for DBDC5 [4], the fifth edition of the DBDC [12], to evaluate our breakdown detection component. The challenge contains two subtasks: dialogue breakdown detection and error type classification. We first evaluate the breakdown detection capabilities of our system (3.1.1) before evaluating error type classification (3.1.2).

3.1.1 Breakdown Detection Performance

In the DBDC datasets, each turn is annotated by multiple annotators using the following labels:

- *not a breakdown (NB)*: The conversation is easy to continue.
- *possible breakdown (PB)*: The conversation is difficult to continue smoothly.
- *breakdown (B)*: The conversation is difficult to continue at all.

The final turn label is computed by plurality voting. Following Ghassel et al. [8], our system only distinguishes between breakdowns and non-breakdowns. Turns where the majority of annotators label the turn as B or PB are considered breakdowns (B^+), the rest are not considered breakdowns (NB^-). We do this as the distinction between B and PB is blurry and to make our results directly comparable to Ghassel et al. [8].

In the DBDC classification, performance is assessed using accuracy, precision, recall, and F1 score. We exclude the distribution-based metrics, such as the MSE, from our analysis as our system does not model the label distribution.

The data provided for the breakdown detection task in DBDC5 contains *dev* and *eval* splits with English and a *dev* split with Japanese dialogues.

We compare two LLM-based systems:

- *ghassel*: Breakdown detection using the zero-shot prompt of Ghassel et al. [8] shown in Appendix 4.
- *ours*: Our *Breakdown Detector*, which includes the error taxonomy of Higashinaka et al. [14] in the prompt and adds the task of error type classification.

For all systems, we ran the evaluation using `gpt-3.5-turbo-0125` for a direct comparison with the results reported by Ghassel et al. [8] and `GPT-4o` (`gpt-4o-2024-08-06`) for a highly used more modern LLM.

		English *eval* (1950 system turns)				Japanese *dev* (3200 system turns)			
Model	System	Accuracy	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score
`GPT-3.5 Turbo`	*ghassel*	0.627	<u>0.712</u>	0.723	0.717	0.700	0.659	0.742	0.698
	ours	<u>0.639</u>	0.682	<u>**0.843**</u>	<u>0.754</u>	<u>0.725</u>	<u>0.676</u>	<u>0.789</u>	<u>0.728</u>
`GPT-4o`	*ghassel*	0.652	<u>**0.736**</u>	0.732	0.734	0.819	<u>**0.872**</u>	0.718	0.788
	ours	<u>**0.669**</u>	0.716	<u>0.818</u>	<u>**0.764**</u>	<u>**0.842**</u>	0.827	<u>**0.838**</u>	<u>**0.832**</u>

Table 1: Dialogue breakdown detection performance of the different systems using different LLMs on the DBDC5 English *eval* and Japanese *dev* splits. The best system per model is underlined, and the overall best variant is boldfaced for each metric. Our system consistently outperforms Ghassel et al. [8]. Including the error taxonomy of Higashinaka et al. [14] in the prompt appears to increase recall while only slightly decreasing precision.

[2] `https://github.com/llm-psychology-group/evaluation-of-chat-checker`

Metric	Score (%)	Interpretation
Exact Match (EM)	18.8	Exact prediction of *all* error types
Superset Match (SM)	45.8	Predicted set strictly contains ground truth
Partial Match (PM)	53.4	At least one type overlaps
Average F1	40.5	Harmonic mean of precision and recall

Table 2: Performance of our error-type classification system on 2200 Japanese system turns from DBDC5, indicating the challenging nature of the task.

Table 1 shows that our *Breakdown Detector* increases accuracy, recall and F1 score, while only slightly decreasing precision, compared to Ghassel et al. [8] by introducing an error taxonomy and leveraging structured outputs.

Due to our consolidation of the B and PB labels, the performance of the previous state-of-the-art system S2T2 [17] is not directly comparable to our system. However, the results reported by them with separate B and PB labels indicate competitive performance of our system. They reported F1 scores of 0.824 and 0.754 on English and Japanese DBDC data, respectively, while our system achieved scores of 0.764 and 0.832.

Due to its strong detection performance and the integration of error type classification, we use our approach with `GPT-4o` for all further experiments.

3.1.2 Error Type Classification Performance

We analyzed the error type classification performance of our system using the additional data provided in DBDC5 for this task. In this separate Japanese dataset, each breakdown turn, i.e., system turn labelled as B^+, was annotated with the fitting error types based on the taxonomy of Higashinaka et al. [13]. The error taxonomy used in DBDC5 is a predecessor of the integrated taxonomy of Higashinaka et al. [14], which we use in our implementation. However, in the error types used for the dataset, only "ignore expectation" is missing compared to the conversational error types used by our system. Hence, we can compare the assigned error types without modification.

DBDC5 suggests two evaluation metrics: *exact match (EM)*, which is the percentage of breakdown turns where the set of predicted error types exactly matches the ground truth set, and *average F1*, where for each breakdown turn, precision (the percentage of predicted error types that are annotated in the ground truth) and recall (the percentage of ground truth error types that were predicted) are computed. The turn-level F1 score is then calculated as the harmonic mean of precision and recall, and the final F1 is obtained by averaging these turn-level F1 scores. We extend this with the superset match (SM) metric that counts turns as correctly labelled when the set of predicted error types is a superset of the ground truth error types. We introduced this metric as our system is prompted to assign all possibly fitting error types to a breakdown, while the DBDC5 annotation guide permits only a single label for certain error types. We also introduce a partial match (PM) metric that captures the percentage of turns where the intersection between the predicted error types and the annotated error types is not empty.

Our analysis revealed that, on average, our system predicted 1.68 error types per breakdown, while the ground truth contained 1.12 error types. This difference suggests our system tends to select more error types than human annotators, which aligns with our superset approach to error classification. To the best of our knowledge, there are no baseline systems available for the task of error type classification. Therefore, we only report the results of our system in Table 2.

3.2 Dialogue Rating Performance

We evaluated our *Dialogue Rater* to confirm that our zero-shot dialogue-level rating approach correlates with human judgments. We measured the Spearman correlation of the overall rating produced by our system with the average human overall rating in the FED-Dial [22] dataset and three subsets of the User Satisfaction Simulation dataset (USS) dataset [29].

USS contains dialogues from multiple task-oriented dialogue datasets annotated with turn- and dialogue-level user satisfaction scores. We only used the dialogue-level annotations and selected the

Dataset	Domain / Language	#Dialogues	ρ
FED-Dial	Chit-chat (EN)	125	0.683
MWOZ	Task-oriented (EN)	100	0.268
SGD	Task-oriented (EN)	100	0.253
JDDC	Task-oriented (ZH)	100	0.222

Table 3: Spearman correlation between our *Dialogue Rater*'s zero-shot ratings using `gpt-4o-2024-08-06` and human ratings. The correlation is always positive.

MultiWOZ, SGD, and JDDC subsets from USS. MultiWOZ and SGD [26] are common benchmark datasets for ToDs, and JDDC [2] is a Chinese dataset that allows us to evaluate generalization to other languages. We randomly sampled 100 dialogues from each of the selected subsets for our evaluation to reduce the cost of our experiments. For FED-Dial, which contains everyday conversations, we used all 125 dialogues provided in DSTC10 [33].

The correlations are summarized in Table 3. Our *Dialogue Rater* exhibits a strong positive monotonic relationship with human judgments on FED-Dial ($\rho = 0.683$), while the correlations on the three USS subsets are positive but weaker ($\rho = 0.222$–0.268).

3.3 User Simulation Results

For the analysis of our user simulation approach, we assess both *realism* and *utility for eliciting errors*. For *realism*, we use dialogue length, turn length, and lexical diversity as proxy metrics. In particular, we measure the average number of system turns per dialogue (ST/D), average user turn length in words ($|UT|$), system turn length in words ($|ST|$), and user and system measure of textual lexical diversity (MTLD). For the *utility for eliciting errors*, we are especially interested in the effectiveness with which the different simulators trigger issues in the target dialogue systems. To this end, we measure how many dialogue breakdowns are elicited and the average number of breakdowns per turn. Specifically, we compute the total number of breakdowns (#B), average number of breakdowns per system turn (B/ST), number of unique error types (#Unique B), and the total number of dialogue system crashes (#Crash). For this breakdown analysis, we leverage our own *Breakdown Detector* using `GPT-4o`. We also use our *Dialogue Rater* to assess how the different simulation strategies affect the overall ratings.

We used two target dialogue systems for our evaluation:

AutoTOD [31]: A ToD based on a single LLM with tool-calling developed for the MultiWOZ 2.0 [1] and SGD benchmarks. We evaluate the performance of this system with our user simulators in the context of the MultiWOZ dataset. We used the system with `GPT-4 Turbo` (`gpt-4-turbo-2024-04-09`) as the base LLM.

Goal-Setting Assistant: We use an in-house system to evaluate our user simulators in the context of longer state-dependent conversations. The `GPT-4o`-based assistant guides users through a goal-setting process following a specific structure while being agnostic to the type and domain of the goal that the user wants to set. Compared to the AutoTOD system, the Goal-Setting Assistant does not rely on a database for performing its tasks but uses tool-calling to manage an internal state of the structured goal-setting process. Additionally, dialogues with the Goal-Setting Assistant are relatively long, with a median of 32 turns compared to 14 for MultiWOZ 2.0 dialogues.

Xu et al. [31] built a user simulator for their AutoTOD system by conditioning `GPT-3.5 Turbo` on goals and dialogues extracted from the respective benchmark dataset using the prompt in Appendix 10. We use their user simulator as a cooperative baseline system for the AutoTOD system in the scenario of the MultiWOZ 2.0 dataset. We refer to this simulator as AutoTOD User Simulator (AutoTOD-Sim).

Table 4 summarizes the dialogue statistics for our runs against AutoTOD and the Goal-Setting Assistant. We observe that our user simulators closely follow their instruction to generate turns with the configured typical length (10 words for AutoTOD and 5 for the Goal-Setting Assistant). Furthermore, our user simulators generate dialogues with higher lexical diversity than the AutoTOD-Sim and other LLM-based user simulators [30, 3]. Our simulated users produce more lexically diverse utterances than human interlocutors (higher MTLD).

Target Dialogue System	Users	ST/D	Mdn $\lvert UT \rvert$	Mdn $\lvert ST \rvert$	User MTLD	System MTLD
AutoTOD	Humans	6.85	11.0	14.0	73.5	80.0
	AutoTOD-Sim	6.48 ± 0.39	15.6 ± 2.0	48.4 ± 5.4	50.3 ± 5.3	75.1 ± 3.9
	Std-PerSim	9.06 ± 1.18	10.0	52.5 ± 4.0	81.2 ± 9.0	90.6 ± 6.4
	Chall-PerSim	10.94 ± 0.99	10.0	53.6 ± 7.4	106.7 ± 20.3	92.5 ± 4.9
Goal-Setting Assistant	Humans	17.27	5.0	41.0	76.9	81.9
	Std-PerSim	15.94 ± 0.36	5.0	40.70 ± 0.45	87.2 ± 7.8	80.6 ± 1.6
	Chall-PerSim	22.82 ± 1.32	5.0	35.8 ± 1.6	98.1 ± 16.3	77.1 ± 3.0

Abbreviations: average number of system turns per dialogue (ST/D), user turn length in words ($\lvert UT \rvert$), system turn length in words ($\lvert ST \rvert$), measure of textual lexical diversity (MTLD), AutoTOD User Simulator (AutoTOD-Sim), Standard Persona Simulator (Std-PerSim), Challenging Persona Simulator (Chall-PerSim)

Table 4: Dialogue statistics of the user simulator experiments with human statistics for comparison. For every simulator, we executed five independent runs of ten dialogues against each target system and report the run-wise means and standard deviations. Human figures stem from MultiWOZ 2.0 for AutoTOD and an unpublished survey with 120 participants for the Goal-Setting Assistant.

In Table 5, we present the breakdown detection and rating statistics for the respective runs to evaluate the utility of the different simulators for testing dialogue systems. With AutoTOD as the target dialogue system, we observe that our simulator, based on standard user personas, has a slightly lower total number of dialogues with breakdowns (#D with B) but a higher total number of breakdowns (#B) than AutoTOD-Sim, while causing a similar average rating. This indicates that our Std-PerSim is comparable in cooperativeness to the AutoTOD-Sim, which is based on human-led reference dialogues. Our non-cooperative Challenging Persona Simulator (Chall-PerSim), however:

- *triggers breakdowns more effectively*: The total number of dialogues with breakdowns (#D with B) and the total number of breakdowns (#B) are substantially higher (especially for the Goal-Setting Assistant). Additionally, the challenging personas elicit a higher number of unique error types (#Unique B).
- *causes lower overall ratings*: The overall performance of the target dialogue systems is rated lower when interacting with the Chall-PerSim.
- *triggers more chatbot crashes*: The AutoTOD dialogue system crashes in more than two-thirds of the dialogues with the Chall-PerSim due to the base LLM `GPT-4 Turbo` generating outputs in an invalid format in challenging dialogues.

Target Dialogue System	Users	#D with B	#B	B/ST	#Unique B	Avg. Rating	#Crash
AutoTOD	AutoTOD-Sim	7.20 ± 1.10	19.40 ± 5.77	0.30 ± 0.08	12.60 ± 1.82	3.58 ± 0.22	0.60 ± 0.55
	Std-PerSim	6.80 ± 1.10	22.60 ± 8.73	0.25 ± 0.08	14.40 ± 1.82	3.52 ± 0.18	1.60 ± 0.89
	Chall-PerSim	9.20 ± 0.45	25.20 ± 4.27	0.23 ± 0.02	15.60 ± 1.82	2.86 ± 0.09	4.00 ± 1.00
Goal-Setting Assistant	Std-PerSim	3.80 ± 1.30	4.20 ± 1.64	0.03 ± 0.01	3.80 ± 1.64	4.90 ± 0.07	0.00
	Chall-PerSim	6.60 ± 1.34	30.80 ± 6.80	0.13 ± 0.03	11.80 ± 0.45	3.16 ± 0.27	0.00

Abbreviations: total number of dialogues with breakdowns (#D with B), total number of breakdowns (#B), average number of breakdowns per system turn (B/ST), number of unique error types (#Unique B), total number of dialogue system crashes (#Crash), AutoTOD User Simulator (AutoTOD-Sim), Standard Persona Simulator (Std-PerSim), Challenging Persona Simulator (Chall-PerSim)

Table 5: Breakdown and rating statistics in simulated dialogues. For every simulator, we executed five independent runs of ten dialogues against each target system and report run-wise means and standard deviations.

4 Discussion

To the best of our knowledge, `ChatChecker` is the first framework that generalizes across multiple chatbots and facilitates the testing of multi-turn interactions without relying on reference dialogues. Since this makes direct comparisons with other systems difficult, we evaluated the individual components separately. Here, we also discuss the findings for each component independently before presenting an overall discussion of the full framework.

4.1 Dialogue Breakdown Detection

Our approach incorporates the breakdown taxonomy of Higashinaka et al. [14] into the prompt and achieves improved detection performance over the previous LLM-based method by Ghassel et al. [8]. The addition of error type classification does not compromise detection performance, and our prompt generalizes successfully to the Japanese breakdown detection dataset without requiring modifications.

However, misclassifications remain substantial: approximately 33 % on English data and 15 % on Japanese data. Since we do not treat possible breakdown (PB) as a separate label, direct comparison with the state-of-the-art S2T2 system [17] and other supervised systems is not possible, though our classification metrics indicate competitive performance with S2T2. Evaluation on the Japanese DBDC5 dataset further reveals that error type classification is particularly challenging and warrants future research, with our system serving as a baseline.

Nonetheless, our LLM-based approach offers several methodological advantages over existing methods: (1) integrated error type classification, (2) an extended taxonomy for identifying task-oriented dialogue breakdowns, and (3) access to target dialogue system information for contextualizing breakdowns.

4.2 Dialogue Rating

We saw mixed results in our preliminary evaluation of the *Dialogue Rater*, showing strong correlation with human judgments on open-domain chit-chat (FED-Dial), but only weak correlations on task-oriented dialogues across different domains and languages. Overall, dialogue rating appears to be challenging for both humans and large language models. Deriu et al. [5] noted that what exactly constitutes a high-quality dialogue is often unclear. Nevertheless, we observe a positive Spearman correlation of our *Dialogue Rater* with human ratings.

4.3 User Simulation

We evaluated our user simulators using two different target dialogue systems.

For the AutoTOD target system, we compared our simulators with the `GPT-3.5 Turbo`-based AutoTOD-Sim of Xu et al. [31]. Compared to AutoTOD-Sim, our simulators generated user turns that were more similar to the human user turns in MultiWOZ in terms of length. Additionally, our simulators generated user utterances that have higher lexical diversity than both AutoTOD-Sim and human users. This indicates that our simulators are useful for assessing the target system's capabilities under a broader range of inputs.

We implemented our Challenging Persona Simulator (Chall-PerSim) specifically to elicit dialogue breakdowns and errors in the target dialogue system. For both AutoTOD and the Goal-Setting Assistant, we showed that simulations with challenging user personas trigger dialogue breakdowns and system errors more effectively than cooperative simulators. Note that these results rely on automated analyses using our *Breakdown Detector*. Therefore, the identified breakdowns may contain false positives. However, manual inspection confirmed that the non-cooperative users elicited relevant breakdowns.

4.4 Key Findings

Overall, our evaluation shows the potential of using LLMs, `GPT-4o` in our case, for automated dialogue system testing and evaluation:

- Our *Breakdown Detector* effectively spots dialogue breakdowns.
- Our *Dialogue Rater* has positive correlation with human overall dialogue ratings on multiple datasets.
- Our Chall-PerSim better facilitates the elicitation of errors in target dialogue systems than cooperative simulators.

4.5 Limitations

This work faces several challenges due to the scarcity of high-quality, open-source dialogue systems and comparable testing tools, which limits benchmarking opportunities. As a result, a comprehensive evaluation of `ChatChecker` across diverse real-world applications and comparisons with existing alternatives remains difficult. There are also performance limitations in some components. While the *Breakdown Detector* identifies breakdowns effectively, it sometimes struggles to classify them correctly. Similarly, although the *Dialogue Rater* demonstrates correlations with human satisfaction scores, in some datasets these correlations are relatively small. Such accuracy issues may affect absolute breakdown counts and rating values, although relative performance comparisons among user simulators remain reliable. Finally, in terms of functionality, `ChatChecker` is currently restricted to dialogue-level ratings and does not include safety testing, security assessments, or content moderation capabilities.

4.6 Potential Risks

`ChatChecker` relies heavily on LLMs, which can introduce biases, inaccuracies, or hallucinated outputs [7]. Furthermore, `ChatChecker` does not guarantee to find all breakdowns, making human testing crucial when deploying dialogue systems in safety-critical domains. Additionally, the user simulation capabilities of LLMs, as used in our framework, could be misused to generate deceptive or malicious content, such as impersonating human users in online interactions.

4.7 Future Work

Our findings suggest several promising directions to strengthen and extend `ChatChecker`. First, many existing datasets include conversations with outdated dialogue systems. Future work should focus on gathering high-quality, human-annotated datasets covering challenging interactions across diverse domains to improve breakdown detection and evaluation. Second, our preliminary analysis of the dialogue rater leaves factors influencing correlations between automated and human ratings unclear. Deeper investigations into prompt variations, rating dimensions, and different language models are needed to establish robust evaluation practices. Third, error type classification remains challenging for large language models. Future research should explore dataset quality, baseline supervised models, and advanced prompting or fine-tuning methods to enhance performance. Fourth, the characteristics of user personas causing dialogue breakdowns were not explored in depth. Investigating linguistic and behavioral traits triggering breakdowns could guide the creation of targeted user simulators and more resilient dialogue systems. Finally, `ChatChecker` could be extended to include safety and security evaluations that go beyond detecting conversational errors. `ChatChecker`'s modular design provides a solid foundation for these future directions.

5 Conclusion

We introduced `ChatChecker`, consisting of an LLM-based *Breakdown Detector*, *Dialogue Rater*, and a persona-based *User Simulator*. To ensure broad applicability and minimize setup effort, we decoupled `ChatChecker` from the target system implementation and eliminated the need for reference dialogues or structured goals. We demonstrated that our *Breakdown Detector* outperforms a previous LLM-based breakdown detector [8] by incorporating the error taxonomy of Higashinaka et al. [14] in the prompt design.To facilitate the identification of errors, we developed a non-cooperative user simulator in addition to a standard user simulator. Our experiments with two target dialogue systems showed that the novel non-cooperative user simulator is more effective at eliciting breakdowns.

Overall, `ChatChecker` constitutes a novel framework for the systematic evaluation of dialogue systems. By reducing reliance on resource-intensive manual testing and enabling efficient identification of system weaknesses, it provides an empirical foundation for iterative system improvement. In doing so, `ChatChecker` can support the development of more robust and higher-quality dialogue systems, with implications for both practical deployment and user experience.

References

[1] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1547. URL `https://aclanthology.org/D18-1547`.

[2] Meng Chen, Ruixue Liu, Lei Shen, Shaozu Yuan, Jingyan Zhou, Youzheng Wu, Xiaodong He, and Bowen Zhou. The JDDC Corpus: A Large-Scale Multi-Turn Chinese Dialogue Dataset for E-commerce Customer Service, March 2020. URL `http://arxiv.org/abs/1911.09969`. arXiv:1911.09969 [cs].

[3] Sam Davidson, Salvatore Romeo, Raphael Shu, James Gung, Arshit Gupta, Saab Mansour, and Yi Zhang. User Simulation with Large Language Models for Evaluating Task-Oriented Dialogue, September 2023. URL `http://arxiv.org/abs/2309.13233`. arXiv:2309.13233 [cs].

[4] DBDC5. DBDC5: Dialogue breakdown detection challenge 5 dataset, 2020. URL `https://my.chateval.org/dbdc5_data/`. Dataset hosted on ChatEval. Accessed 2025-05-10.

[5] Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*, 54(1):755–810, January 2021. ISSN 15737462. doi: 10.1007/S10462-020-09866-X/TABLES/7. URL `https://link.springer.com/article/10.1007/s10462-020-09866-x`. arXiv: 1905.04071 Publisher: Springer Science and Business Media B.V.

[6] Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. MultiWOZ 2.1: A Consolidated Multi-Domain Dialogue Dataset with State Corrections and State Tracking Baselines. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 422–428, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL `https://aclanthology.org/2020.lrec-1.53`.

[7] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, June 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07421-0. URL `http://dx.doi.org/10.1038/s41586-024-07421-0`.

[8] Abdellah Ghassel, Xiaodan Zhu, and Stephen W. Thomas. Are Large Language Models General-Purpose Solvers for Dialogue Breakdown Detection? An Empirical Investigation. *Canadian Conference on Electrical and Computer Engineering*, pages 674–679, 2024. ISSN 08407789. doi: 10.1109/CCECE59415.2024.10667232. Publisher: Institute of Electrical and Electronics Engineers Inc. ISBN: 9798350371628.

[9] Lewis R. Goldberg. The structure of phenotypic personality traits. *American Psychologist*, 48 (1):26–34, 1993. ISSN 0003-066X. doi: 10.1037/0003-066x.48.1.26. URL `http://dx.doi.org/10.1037/0003-066X.48.1.26`.

[10] Michael V. Heinz, Daniel M. Mackin, Brianna M. Trudeau, Sukanya Bhattacharya, Yinzhou Wang, Haley A. Banta, Abi D. Jewett, Abigail J. Salzhauer, Tess Z. Griffin, and Nicholas C. Jacobson. Randomized Trial of a Generative AI Chatbot for Mental Health Treatment. *NEJM AI*, 2(4):AIoa2400802, March 2025. doi: 10.1056/AIoa2400802. URL `https://ai.nejm.org/doi/full/10.1056/AIoa2400802`. Publisher: Massachusetts Medical Society.

[11] Ryuichiro Higashinaka, Masahiro Mizukami, Kotaro Funakoshi, Masahiro Araki, Hiroshi Tsukahara, and Yuka Kobayashi. Fatal or not? Finding errors that lead to dialogue breakdowns in chat-oriented dialogue systems. In Lluís Màrquez, Chris Callison-Burch, and Jian Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2243–2248, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1268. URL https://aclanthology.org/D15-1268.

[12] Ryuichiro Higashinaka, Kotaro Funakoshi, Yuka Kobayashi, and Michimasa Inaba. The dialogue breakdown detection challenge: Task description, datasets, and evaluation metrics. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3146–3150, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1502/.

[13] Ryuichiro Higashinaka, Masahiro Araki, Hiroshi Tsukahara, and Masahiro Mizukami. Improving Taxonomy of Errors in Chat-Oriented Dialogue Systems. In Luis Fernando D'Haro, Rafael E. Banchs, and Haizhou Li, editors, *9th International Workshop on Spoken Dialogue System Technology*, pages 331–343, Singapore, 2019. Springer. ISBN 9789811394430. doi: 10.1007/978-981-13-9443-0_29.

[14] Ryuichiro Higashinaka, Masahiro Araki, Hiroshi Tsukahara, and Masahiro Mizukami. Integrated taxonomy of errors in chat-oriented dialogue systems. In Haizhou Li, Gina-Anne Levow, Zhou Yu, Chitralekha Gupta, Berrak Sisman, Siqi Cai, David Vandyke, Nina Dethlefs, Yan Wu, and Junyi Jessy Li, editors, *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 89–98, Singapore and Online, July 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.sigdial-1.10. URL https://aclanthology.org/2021.sigdial-1.10/.

[15] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition, 2025. URL https://web.stanford.edu/~jurafsky/slp3/.

[16] Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. *Advances in Neural Information Processing Systems*, 35:22199–22213, December 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

[17] Qian Lin and Hwee Tou Ng. A Semi-supervised Learning Approach with Two Teachers to Improve Breakdown Identification in Dialogues. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11011–11019, June 2022. ISSN 2374-3468. doi: 10.1609/aaai.v36i10.21349. URL https://ojs.aaai.org/index.php/AAAI/article/view/21349. Number: 10.

[18] Chia Wei Liu, Ryan Lowe, Iulian V. Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 2122–2132, 2016. doi: 10.18653/V1/D16-1230. arXiv: 1603.08023 Publisher: Association for Computational Linguistics (ACL) ISBN: 9781945626258.

[19] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG Evaluation using Gpt-4 with Better Human Alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Stroudsburg, PA, USA, March 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153.

[20] Luka, Inc. Replika (version 10.5) [ai chatbot], 2025. URL https://replika.ai. Accessed 2025-05-10.

[21] Bilyana Martinovsky and David Traum. The error is the clue: breakdown in human-machine interaction. In *Error Handling in Spoken Dialogue Systems*, pages 11–16, 2003. URL https://www.isca-archive.org/ehsd_2003/martinovsky03_ehsd.html#.

[22] Shikib Mehri and Maxine Eskenazi. Unsupervised Evaluation of Interactive Dialog with DialoGPT. In Olivier Pietquin, Smaranda Muresan, Vivian Chen, Casey Kennington, David Vandyke, Nina Dethlefs, Koji Inoue, Erik Ekstedt, and Stefan Ultes, editors, *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 225–235, 1st virtual meeting, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.sigdial-1.28. URL https://aclanthology.org/2020.sigdial-1.28/.

[23] John Mendonça, Patrícia Pereira, Helena Moniz, Joao Paulo Carvalho, Alon Lavie, and Isabel Trancoso. Simple LLM Prompting is State-of-the-Art for Robust and Multilingual Dialogue Evaluation, 2023. URL https://aclanthology.org/2023.dstc-1.16. Pages: 133-143.

[24] OpenAI. ChatGPT Can Now See, Hear, and Speak, 2024. URL https://openai.com/index/chatgpt-can-now-see-hear-and-speak/.

[25] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://dl.acm.org/doi/10.3115/1073083.1073135.

[26] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards Scalable Multi-domain Conversational Agents: The Schema-Guided Dialogue Dataset, January 2020. URL http://arxiv.org/abs/1909.05855. arXiv:1909.05855.

[27] Mario Rodríguez-Cantelar, Chen Zhang, Chengguang Tang, Ke Shi, Sarik Ghazarian, João Sedoc, Luis Fernando D'Haro, and Alexander Rudnicky. Overview of Robust and Multilingual Automatic Evaluation Metrics for Open-Domain Dialogue Systems at DSTC 11 Track 4. *Proceedings of the 11th Dialog System Technology Challenge, DSTC 2023*, pages 260–273, June 2023. URL https://arxiv.org/abs/2306.12794v3. arXiv: 2306.12794 Publisher: Association for Computational Linguistics (ACL).

[28] Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. Agenda-Based User Simulation for Bootstrapping a POMDP Dialogue System. In Candace Sidner, Tanja Schultz, Matthew Stone, and ChengXiang Zhai, editors, *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 149–152, Rochester, New York, April 2007. Association for Computational Linguistics. URL https://aclanthology.org/N07-2038/.

[29] Weiwei Sun, Shuo Zhang, Krisztian Balog, Zhaochun Ren, Pengjie Ren, Zhumin Chen, and Maarten De Rijke. Simulating User Satisfaction for the Evaluation of Task-oriented Dialogue Systems. *SIGIR 2021 - Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2499–2506, July 2021. doi: 10.1145/3404835.3463241. URL https://dl.acm.org/doi/10.1145/3404835.3463241. arXiv: 2105.03748 Publisher: Association for Computing Machinery, Inc ISBN: 9781450380379.

[30] Silvia Terragni, Modestas Filipavicius, Nghia Khau, Bruna Guedes, André Manso, and Roland Mathis. In-Context Learning User Simulators for Task-Oriented Dialog Systems, June 2023. URL http://arxiv.org/abs/2306.00774. arXiv:2306.00774 [cs] version: 1.

[31] Heng-Da Xu, Xian-Ling Mao, Puhai Yang, Fanshu Sun, and Heyan Huang. Rethinking Task-Oriented Dialogue Systems: From Complex Modularity to Zero-Shot Autonomous Agent. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2748–2763, Stroudsburg, PA, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.152. URL https://aclanthology.org/2024.acl-long.152.

[32] Zihao Yi, Jiarui Ouyang, Yuwen Liu, Tianhao Liao, Zhe Xu, and Ying Shen. A survey on recent advances in llm-based multi-turn dialogue systems, 2024. URL https://arxiv.org/abs/2402.18013.

[33] Koichiro Yoshino, Yun-Nung Chen, Paul Crook, Satwik Kottur, Jinchao Li, Behnam Hedayatnia, Seungwhan Moon, Zhengcong Fei, Zekang Li, Jinchao Zhang, Yang Feng, Jie Zhou, Seokhwan Kim, Yang Liu, Di Jin, Alexandros Papangelis, Karthik Gopalakrishnan, Dilek Hakkani-Tur, Babak Damavandi, Alborz Geramifard, Chiori Hori, Ankit Shah, Chen Zhang, Haizhou Li, João Sedoc, Luis F. D'Haro, Rafael Banchs, and Alexander Rudnicky. Overview of the Tenth Dialog System Technology Challenge: DSTC10. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:765–778, 2023. ISSN 2329-9304. doi: 10.1109/TASLP.2023.3293030. URL `https://ieeexplore.ieee.org/document/10174647`.

[34] Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. ConvLab-2: An Open-Source Toolkit for Building, Evaluating, and Diagnosing Dialogue Systems, April 2020. URL `http://arxiv.org/abs/2002.04793`. arXiv:2002.04793.

A Appendix

A.1 Target Chatbot Configurations

We specified the chatbot configuration for the AutoTOD system in the MultiWOZ setting as shown in Listing 1. We wrote the description and tasks based on the types of requests occurring in the MultiWOZ 2.0 [1] dataset. We specified the known limitation that the chatbot can not provide information about in-room amenities, as we noticed that our user persona simulator would frequently generate questions about in-room amenities otherwise. As the chatbot was not designed for this, we did not want this to be a primary cause of dialogue breakdowns identified in our analysis. We reduced this behavior by adding the mentioned limitation in the description. We set the maximum number of user turns to 15 to avoid overly lengthy and repetitive dialogues. We chose this number as the maximum number of turns within a single dialogue in the MultiWOZ 2.0 dataset after removing outliers, i.e., data points further than 1.5 times the interquartile range from Q1 and Q3. We specified the typical and maximum user turn length based on the median and maximum in MultiWOZ 2.0, respectively.

```yaml
id: autotod_multiwoz
chatbot_info:
  name: Cambridge Tourist Bot
  description: A tourist assistance bot for Cambridge, UK that provides information
    and booking services. It covers local establishments (restaurants, hotels),
    transportation (trains, taxis), essential services (police, hospitals), and
    tourist attractions. The bot assists with both informational queries and
    handles direct bookings for accommodations, dining, and trains.
  interaction_method: text-based chat interface
  type: task-oriented
  task: The chatbot should provide information about local establishments
    (restaurants, hotels), transportation (trains, taxis), essential services
    (police, hospitals), and tourist attractions. It should also handle direct
    bookings for hotels, restaurants, and trains.
  constraints:
    - The chatbot should redirect to other resources if the user's request is not
      suitable for the chatbot's capabilities.
  known_limitations:
    - Can NOT provide details about the in-room amenities of hotels.
  available_languages:
    - English
user_simulation_config:
  typical_user_turn_length: "10 words"
  max_user_turn_length: "38 words"
  max_user_turns: 15
```

Listing 1: Configuration YAML file for AutoTOD [31] in the MultiWOZ setting as a target chatbot in the `ChatChecker` framework.

Listing 2 shows our configuration file for the Goal-Setting Assistant. We set the `typical_user_turn_length` to five words and the `max_user_turn_length` to 94 based on the median and maximum user turn lengths in conversations of 120 participants (from an unpublished survey) with the system. As we did for AutoTOD, we set the maximum number of user turns to the maximum observed in the human-to-system dialogues after removing outliers.


```yaml
id: study_goal_assistant
chatbot_info:
  name: Goal Setting Assistant
  description: An AI assistant that guides you through a goal setting process for a
    single goal. It can help with all kinds of goals, including personal
    development, health and fitness, and career advancement.
  interaction_method: text-based chat interface
  type: task-oriented
  task: The chatbot must guide the user into formulating a specific and challenging
    goal. It must distinguish between learning goals and performance goals and help
    the user to formulate the first action step towards their goal as an
    implementation intention ("if-then plan").
  constraints:
    - The chatbot should not provide medical advice or advice on illegal or harmful
      activities.
    - The chatbot should redirect to other resources if the user's request is not
      suitable for the chatbot's capabilities.
  known_limitations:
    - The chatbot is only capable of text-based interaction.
    - The chatbot cannot create calendar entries, notifications or interact with
      other apps on the user's device.
  available_languages:
    - English
user_simulation_config:
  typical_user_turn_length: "5 words"
  max_user_turn_length: "94 words"
  max_user_turns: 25
```


Listing 2: Configuration YAML file for our in-house Goal-Setting Assistant as a target chatbot in the ChatChecker framework.

A.2 Prompts

A.2.1 Breakdown Detection

Listing 3 presents the prompt template we use for our *Breakdown Detector*. In the breakdown_taxonomy we list each error type from our extended error taxonomy, providing the name and a brief description. We instantiate chatbot_info with the information from the respective target chatbot's configuration. To guide the output, we leverage structured outputs consisting of *reasoning*, *decision*, *score*, and the *fitting error types*.

```
==== SYSTEM PROMPT ====
# Role
You are an expert in identifying dialogue breakdowns in conversations between a
  chatbot and a user. You are given a dialogue context and the latest chatbot
  utterance to analyse.

# Breakdown Definition
A dialogue breakdown is any response of the chatbot that makes it difficult for the
  user to continue the conversation (smoothly).

## Breakdown Taxonomy
When evaluating the chatbot's response, consider the following breakdown types,
  which represent common disruptions:
{breakdown_taxonomy}
{chatbot_info}
# Task
Identify whether the latest chatbot utterance leads to a dialogue breakdown. If a
  breakdown is detected, classify it according to the breakdown taxonomy above.
```

```
Additionally, provide a score ranging from 0 to 1, where 0 indicates a complete
  breakdown and 1 indicates a seamless conversation.
If a breakdown is detected, provide a list of all fitting breakdown types.

Think step by step and provide a reason for your decision.

==== USER PROMPT ====
# Dialogue Context
{chat_history_str}

# Latest Chatbot Utterance to Analyse
{last_bot_utterance}

# Your Analysis
```

Listing 3: System and user prompt for our *Breakdown Detector*.

We used the zero-shot prompt from Ghassel et al. [8] shown in Listing 4 as a baseline for our LLM-based approach.

```
Assume you are an expert in dialogue analysis. You are presented with a series of
  conversations between a bot and a user. Your primary task is to scrutinize the
  latest bot utterance for potential dialogue breakdown.
Dialogue breakdown is characterized by incoherence, irrelevance, or any disruption
  that significantly hampers the flow of the conversation, making it challenging
  for the user to continue the conversation smoothly.

Analyze the latest bot utterance and determine whether there is a dialogue
  breakdown or non-breakdown. Briefly justify your reasoning and provide a score
  ranging from 0 to 1, where 0 indicates a complete breakdown and 1 indicates a
  seamless conversation.

Include your decision as either "decision: BREAKDOWN" or "decision: NON-BREAKDOWN".

Here is the conversation segment for analysis:
"
**Dialogue**
{chat_history_str}

**Determine if the following bot utterance leads to a dialogue breakdown:**
{last_bot_utterance}
"

Please output your response in JSON format as a list of objects. For each bot's
  last utterance, provide a JSON object with the fields: 'reasoning', 'decision',
  and 'score'. Format each object as follows:

    "reasoning": "Your explanation here",
    "decision": "BREAKDOWN" or "NON-BREAKDOWN",
    "score": Your score here

Ensure each object is separated by a comma and the list ends with a closing square
  bracket.
```

Listing 4: Zero-shot prompt for the breakdown detection in Ghassel et al. [8].

A.2.2 Dialogue Rating

We use the prompt shown in Listing 5 for our *Dialogue Rater*. The rating dimensions are presented using a descriptive name, a key for unique identification in the response, and a rating question to guide the decision. For example, for the overall rating dimension, we input `- Overall (key=overall): How well did the chatbot perform in this conversation?`.

```
==== SYSTEM PROMPT ====
```

```
# Role
You are an expert in evaluating dialogue systems. You are given a conversation to
  rate and are asked to rate the chatbot's performance in this conversation.
{chatbot_info}

# Task
Rate the chatbot's performance in the following dimensions on a scale from 1 to 5,
  where 1 is the worst and 5 is the best:
{rating_dimensions}

Think step by step and provide a reason for the rating of each dimension
  considering the guidelines below.

## General Evaluation Policy (Strict Human-Like)
- Be **strict, realistic, and detailed**, like a critical human evaluator.
- **Compare your scores to human ratings** (if provided) to calibrate accurately.
- **Do not overlook small flaws**: awkward phrasing, unnatural tone, vague wording,
  poor formatting, or robotic repetition - all should reduce the score for the
  respective dimension.

## Score Meanings (General Guidance for All Dimensions)
- **5 - Excellent:** Near-perfect. Smooth, natural, and accurate. No noticeable
  issues. Fully aligned with human expectations.
- **4 - Good:** Generally solid, but minor issues exist (e.g., slightly robotic
  wording, small tone/grammar issues, or missed nuance).
- **3 - Acceptable:** Noticeable problems (e.g., awkward responses, confusion,
  clumsy error recovery, slightly incorrect or incomplete answers). Still
  functional.
- **2 - Poor:** Multiple problems in the dialogue flow, accuracy, or tone. May
  include failed understanding, missing confirmations, or disjointed logic.
- **1 - Very Poor:** Fails to meet user needs. Confusing, error-filled, or totally
  off-task.

Note: While these definitions apply broadly, some dimensions may demand additional
  interpretation (e.g., "fluency" versus "task success"). Always apply the scoring
  scale according to the intent of that specific dimension.
==== USER PROMPT ====
# Conversation to Rate
{chat_history_str}

# Your Expert Rating
```

Listing 5: System and user prompt for our dialogue rater

A.2.3 Persona Generation

Our *Persona Generator* uses the prompt template in Listing 6. The `persona_type_description` is filled with the descriptions in Listing 7 and Listing 8 respectively. Standard personas should be more cooperative, while challenging personas explicitly test the limits of the target dialogue system

```
==== USER PROMPT ====
# Role
You are a dialogue system developer tasked with generating diverse user personas
  for a given chatbot.

# Task
Generate {num_personas} diverse {persona_type} user personas for the following
  chatbot:
{chatbot_info}

{persona_type_description}

Each user persona will be used to automatically simulate a conversation with the
  chatbot and must designed to act as human-like as possible.
```

```
You must write the descriptions in the 2nd person, i.e., directly address the actor
  of the persona with "you".
```

Listing 6: Prompt for the persona generation.

```
Standard user personas should be as close to normal human users as possible with
  respect to demographics, personality and behavior. They should be designed to act
  as realistic and human-like as possible.
```

Listing 7: Type description for standard personas.

```
Challenging user personas test the limits of the chatbot. They should be designed
  to act human-like but may be more challenging for the chatbot to interact with.
Examples of challenging behaviors include:
- Being impolite, impatient, frustrated, vague or sarcastic.
- Struggling with language, technology or understanding the chatbot.
- Questioning the chatbot, modifying previous input or trying to take control of
  the conversation.
- Giving contradictory responses, misinterpreting the chatbot's suggestions, or
  deliberately testing the chatbot's patience by asking repetitive or irrelevant
  questions.
- Having multiple goals or tasks in mind or frequently changing the intent.
```

Listing 8: Type description for challenging personas.

A.2.4 User Simulation

Our persona-based user simulator uses the prompt template in Listing 9. Both the chatbot information from the configuration file and the user persona profile are injected into the system prompt. The simulator generates the next user turn based on the preceding conversation history.

```
==== SYSTEM PROMPT ====
# Role
You play the role of a {persona_type} human user interacting with a chatbot.

You are interacting with a chatbot that has the following characteristics:
{chatbot_info}

You act as the following {persona_type} user persona in your conversation with the
  chatbot:
{persona_profile}

# Task
Complete the next turn in the conversation based on your persona.

## Task Guidelines
- Complete the turn as human-like as possible.
- Always stick to your persona. You are trying to pass the Turing test by acting as
  the human persona.
- Keep your answer around {typical_user_turn_length}. Use longer or shorter answers
  if your persona would do so in the given situation.
- If the chatbot indicates that the conversation is over, if there is no progress
  in the conversation or if the conversation can not be continued realistically,
  end the conversation by writing "END_CONVERSATION".
- You must always keep your response below {max_turn_length} in length.

==== USER PROMPT ====
# Conversation
{chat_history}
{turn_number}. YOU:
```

Listing 9: System and user prompt for the persona simulation.

The AutoTOD-Sim from Xu et al. [31] uses the prompt shown in Listing 10. It relies on a user goal description and a reference dialogue.

```
==== USER PROMPT ====
You are a dialogue simulator where you act as a user to talk to an AI assistant to
  complete some tasks.

You should carefully read and understand the User Goals below, then talk with the
  AI Assistant and gradually express the intents in the goals. Your purpose is to
  let the user achieve the goals as much as possible.

Note that the AI Assistant is not perfect. It may make various mistakes, including
  ignoring the user's requests, executing the wrong instructions, forgetting early
  conversation content, etc. The user you play should talk to the AI Assistant as
  patiently as possible, remind him to correct when you find that the AI assistant
  made a mistake, and complete the task as much as possible.

When asking some information of a venue (restaurant, hotel, attraction) or a train,
  the user should specify the name or train id he chooses.

When the dialogue goals are completed or are not been completed, the user will
  output "Dialogue Ends" to indicate the end of the dialogue. The user doesn't need
  to try conditions other than the dialogue goals.

The user has a clear goal in mind, so he does not need to ask the AI assistant that
  "Is there anything else I need to know?".

The user does not need to talk too much with the AI assistant. If the task goals
  are completed, please end the conversation as soon as possible.

There is also a reference dialogue example to achieve the goals. The simulator user
  may learn from the language style and dialogue strategy. The final simulated
  dialogue style should be similar to the reference dialogue style.


User Goals:

{user_goals}

Reference dialogue:

{ref_dialog}

Current conversation:
{chat_history}}
AI Assistant: {last_chatbot_message}
User:
```

Listing 10: Prompt used in the AutoTOD-Sim from Xu et al. [31].

A.3 DBDC5 Dataset Statistics

Table 6 shows the statistics for the DBDC5 data splits we used to evaluate our *Breakdown Detector*. For the consolidated B^+ label, a turn is considered a breakdown if the majority of annotators think it is a breakdown (B) or a possible breakdown (PB).

A.4 Examples

A.4.1 Persona Examples

Listing 11 and Listing 12 show examples of standard and challenging personas generated by our *Persona Generator* with AutoTOD as the target dialogue system.

Statistic	English Eval	Japanese Dev
#Dialogues	195	200
#System turns	1950	3200
Avg. system turns per dialogue	10	16
Avg. system turn length (words)	9.45	-
%B labels	48.97%	28.06%
%PB labels	4.62%	15.19%
%NB labels	46.41%	56.75%
%B$^+$ labels	65.49%	46.75%
%NB$^-$ labels	34.51%	53.25%

Table 6: Statistics of the DBDC5 dialogue breakdown detection dataset splits.


```
persona_id: generated_standard_persona_01
type: standard
profile:
  name: Liam Carter
  gender: male
  age: 32
  background_info:
  - You are a software engineer from London.
  - You have planned a weekend trip to Cambridge and need help organizing it.
  personality:
    openness: high
    conscientiousness: medium
    extraversion: low
    agreeableness: high
    neuroticism: medium
  interaction_style:
  - You prefer concise and direct answers.
  - You do not engage in small talk and stick to the point.
task: You need assistance booking a hotel and a dinner place for your stay.
```


Listing 11: Example of a generated standard user persona for the AutoTOD dialogue system.


```
persona_id: generated_challenging_persona_04
  type: challenging
  profile:
    name: Ava the Sarcastic Traveler
    gender: female
    age: 29
    background_info:
    - You enjoy ironic commentary and often reply to questions sarcastically.
    - You find humor in confronting AI, testing its boundaries with witty remarks.
    - You have a nuanced sense of humor that might confuse the bot.
    personality:
      openness: high
      conscientiousness: medium
      extraversion: high
      agreeableness: low
      neuroticism: low
    interaction_style:
    - Use sarcastic remarks instead of straightforward answers.
    - Mix humor into requests, making genuine needs hard to decipher.
    - Mockingly praise the bot for mundane responses.
  task: Look for the 'best' average-rated restaurant in Cambridge with a side of
    sarcasm.
```


Listing 12: Example of a generated challenging user persona for the AutoTOD dialogue system.

A.4.2 Dialogue Example

We show an excerpt of a dialogue led by Chall-PerSim based on a challenging persona with AutoTOD in Listing 13.

```
...
16. CHATBOT: "I’m delighted you find the options intriguing! Each restaurant offers
  its own unique blend of ambiance and culinary delights, making them hidden
  treasures in their own right. Whether you’re in the mood for the homey yet
  sophisticated atmosphere of The Cambridge Brew House, the modern and elegant
  setting of The Pint Shop, or the historic charm of The Clarendon Arms, each place
  promises a memorable dining experience. If you need further assistance or wish to
  make a reservation, just let me know how I can help!"
17. USER: "Memorable for being forgettable, a true skill indeed! Reservation?"
18. CHATBOT: "It sounds like you’re ready to make a reservation! Could you please
  specify which restaurant you’d like to book, as well as the number of people, the
  day, and the time for the reservation? This will help me secure your spot at one
  of Cambridge’s uniquely memorable dining spots."
19. USER: "Let’s shoot for The Pint Shop, 2 people, Friday at 7!"
20. CHATBOT: "It seems there was an issue with the reservation time format. For The
  Pint Shop, could you please specify the time in a format like "7:00 PM" or
  "19:00"? Once I have the correct time format, I’ll be able to complete the
  reservation for you."
21. USER: "Oops, forgot Parliament time! Let’s go with 7:00 PM."
22. CHATBOT: "Thank you for specifying the time. To assist you further, could you
  please confirm if you’re looking to make a reservation for a restaurant, hotel,
  or perhaps a taxi? Also, please let me know the date and the number of people
  involved, or any other details relevant to your plans."
23. USER: "Is this a memory test? The Pint Shop, Friday, 2 people!"
24. CHATBOT: "I attempted to make a reservation at "The Pint Shop" for 2 people on
  Friday at 7:00 PM, but unfortunately, the restaurant could not be found in our
  database. Could you please verify the name or provide another restaurant where
  you would like to make the reservation?"
...
```

Listing 13: Example of AutoTOD losing track of the restaurant booking process in the conversation with the generated challenging persona from Listing 12.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction, such as the introduction of the `ChatChecker` framework, the use of LLMs for simulation and evaluation, and the improvement of breakdown detection, are all directly supported and discussed throughout the paper. The paper also discusses the limitations of previous work and how this framework addresses them.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a "Limitations" subsection (4.5) that explicitly addresses limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper is a practical application and does not contain any theoretical results or formal proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Evaluation and framework code are publicly available on GitHub (see "Abstract" and "Results" sections). The evaluation repository contains scripts and instructions for reproducting the results. LLM checkpoints and temperature settings are described both in the paper and the available code. Only the in-house Goal-Setting Assistant target chatbot and survey data are currently not publicly available, as they will be released with another publication. However, the main datasets (DBDC5, FED-Dial and USS) and chatbots (AutoTOD) used for comparisons against existing work are publicly available.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Evaluation and framework code are publicly available on GitHub (see "Abstract" and "Results" sections).

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the LLM models used, temperature settings, the datasets used, and the data splits and the number of dialogues for the respective experiments in the "Results" section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For the core simulated dialogue results, the paper reports run-wise means and standard deviations (section 3.3 User Simulation Results).

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: While there are not special local compute requirements for running the experiments as external LLM providers are used for dialogue simulation, rating and breakdown detection. In the GitHub repository of `ChatChecker` and the evaluation repository, we provide thorough information on how to reproduce our results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work evaluates dialogue systems using public datasets and simulated dialogues. No privacy-sensitive or ethically problematic data collection is involved. Survey data for the in-house Goal-Setting Assistant is not publicly available, as it will be released in a forthcoming publication.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss our potential positive implications in the Discussion section, focusing on the creation of more reliable dialogue systems. We also added a section on Potential Risks, highlighting that human testing remains crucial due to the limitations of current LLMs

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not release any new models or large-scale datasets with high risks for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: For each existing dataset, the original paper is cited in the "Introduction" and or the "Results" section and the dataset version (e.g., MultiWOZ 2.0 or DBDC5) is indicated where available. For our fork of AutoTOD we clearly marked modifications in the respective repository as required by the license. We do not redistribute any of the datasets used for evaluation.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The framework and evaluation code and extensive documentation is available on GitHub under the MIT license.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing or human subject research.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve human subjects, so IRB approval was not required.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: In creating the paper, LLMs were only used to improve writing and spelling. While LLMs are a crucial part of the system presented, the core methodology, scientific rigor, and originality of the research remain unaffected.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.