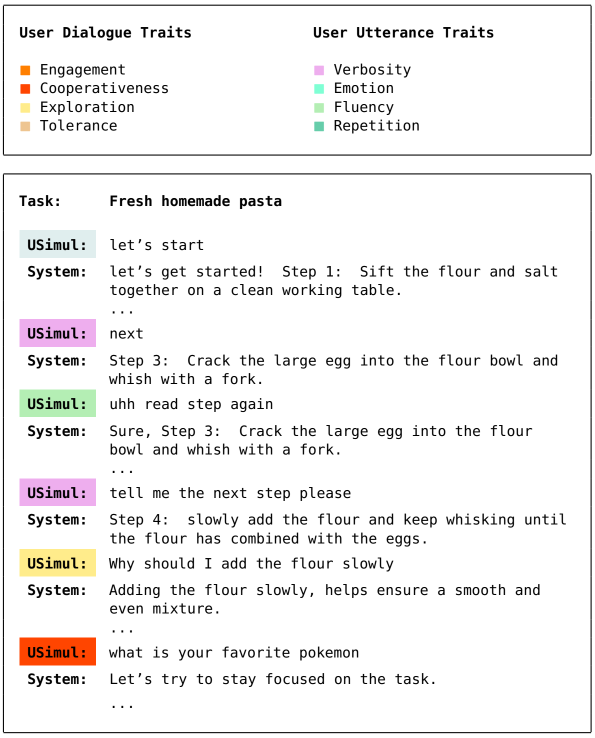
Multi-trait User Simulation with Adaptive Decoding for Conversational Task Assistants

Rafael Ferreira and **David Semedo** and **João Magalhães**
NOVA University of Lisbon and NOVA LINCS
`rah.ferreira@campus.fct.unl.pt`
`{df.semedo,jm.magalhaes}@fct.unl.pt`

Abstract

Conversational systems must be robust to user interactions that naturally exhibit diverse conversational traits. Capturing and simulating these diverse traits coherently and efficiently presents a complex challenge. This paper introduces Multi-Trait Adaptive Decoding (*mTAD*), a method that generates diverse user profiles at decoding-time by sampling from various trait-specific Language Models (LMs). *mTAD* provides an adaptive and scalable approach to user simulation, enabling the creation of multiple user profiles without the need for additional fine-tuning. By analyzing real-world dialogues from the Conversational Task Assistant (CTA) domain, we identify key conversational traits and developed a framework to generate profile-aware dialogues that enhance conversational diversity. Experimental results validate the effectiveness of our approach in modeling single-traits using specialized LMs, which can capture less common patterns, even in out-of-domain tasks. Furthermore, the results demonstrate that *mTAD* is a robust and flexible framework for combining diverse user simulators.



User Dialogue Traits
- Engagement
- Cooperativeness
- Exploration
- Tolerance

User Utterance Traits
- Verbosity
- Emotion
- Fluency
- Repetition

Task:	**Fresh homemade pasta**
USimul:	let's start
System:	let's get started! Step 1: Sift the flour and salt together on a clean working table. ...
USimul:	next
System:	Step 3: Crack the large egg into the flour bowl and whish with a fork.
USimul:	uhh read step again
System:	Sure, Step 3: Crack the large egg into the flour bowl and whish with a fork. ...
USimul:	tell me the next step please
System:	Step 4: slowly add the flour and keep whisking until the flour has combined with the eggs.
USimul:	Why should I add the flour slowly
System:	Adding the flour slowly, helps ensure a smooth and even mixture. ...
USimul:	what is your favorite pokemon
System:	Let's try to stay focused on the task. ...

Table 1: Simulated dialogue example illustrating the impact of different user traits on conversational assistants, testing their ability to manage a variety of user behaviors.

1 Introduction

Thoroughly testing a conversational system with real users is a costly and time consuming process. Part of this cost lies in the myriad of conversational traits, reflecting user's behaviors, knowledge, and goals, which when diverse have shown to help improve performance of conversational systems (Tang et al., 2021; Shi et al., 2019). User simulators have been a successful approach to model real user conversational traits and discover errors and limitations in conversational systems (Shi et al., 2019; Rastogi et al., 2020; Liu et al., 2023; Zhao et al., 2023). Prior research investigated user simulators across diverse settings, including task-oriented dialogues (Lin et al., 2021; Shi et al., 2019; Liu et al., 2023) and recommendation (Zhang and Balog, 2020; Afzali et al., 2023; Zhao et al., 2023) contexts. However, designing user simulators that can effectively engage with a system is challenging, due to the need for adaptability and controllability across various user conversational patterns.

Simulating a user conversational profile entails a combination of diverse conversational traits, as shown in Table 1. In this work, we follow model-based approaches (Liu et al., 2023; Lin et al., 2021; Shi et al., 2019) by learning trait-specific Language Models (LMs) and combining them into multiple different user profiles. Approaches to combine models include weight level approaches (Wortsman et al., 2022; Ilharco et al., 2023; Yu et al., 2023; Yadav et al., 2023), and trainable Mixture-of-Experts (MoE) (Feng et al., 2024; Chen et al.,

Findings of the Association for Computational Linguistics: EACL 2024, pages 16105–16130
November 12-16, 2024 ©2024 Association for Computational Linguistics

2024a; Wu et al., 2024; Jiang et al., 2024). Instead of combining LMs before inference time, we propose an adaptive and scalable method that combines user traits at decoding-time by sampling from distributions from each trait-specific LM. With the proposed multi-Trait Adaptive Decoding method (*mTAD*), we are able to sidestep the need for combinatory training datasets or extra model fine-tuning. In addition, new LM traits can be adaptively added to the pool of traits creating a new user profile, without the need to retrain existing LMs. As we show, combining these traits is crucial for fostering more diverse conversational patterns.

To identify these traits, we analyzed real-world dialogues in the novel Conversational Task Assistant (CTA) domain (Gottardi et al., 2022; Agichtein et al., 2023) and extracted the most relevant traits. In CTA scenarios, users actively engage in completing manual tasks (e.g., baking a cake) with the system's assistance, fostering mixed-initiative dialogues that pose unique user modeling challenges.

In summary, one of the core contributions of this paper is the **Multi-Trait Adaptive Decoding method (*mTAD*)**, which allows for the combination of LMs to simulate multiple different user profiles at decoding time, removing the need for combinatory data or additional fine-tuning. The second contribution is the introduction of a set of **user conversational traits** at dialogue-level and utterance-level, derived from **real-world CTA data** collected during the Alexa TaskBot Challenge (Gottardi et al., 2022). Experimental results support the need for specialized LMs to accurately represent each trait. Additionally, the results show the scalability and performance of *mTAD*, which can flexibly simulate users with arbitrary combinations of traits, without user profile-specific training.

2 Related Work

Conversational Assistants Prior work has focused on task-oriented (Budzianowski et al., 2018; Rastogi et al., 2020) and recommendation systems (Li et al., 2018; Lei et al., 2020), where the assistant performs tasks based on user input (e.g. buying a ticket). Our work departs from these traditional settings and explores CTAs (Gottardi et al., 2022), where users complete tasks (e.g., cooking) with help from the assistant, presenting unique modeling challenges (Chan et al., 2023), which we investigate in the context of user simulation.

User simulators Initial works (Schatzmann et al., 2007; Li et al., 2016) used rule-based agendas to model user actions. More recently, model-based approaches (Shi et al., 2019), in specific with LLMs (Lin et al., 2021; Li et al., 2022; Liu et al., 2023) have been used. Additionally, diverse user simulators have shown to improve system performance (Liu et al., 2023; Tang et al., 2021). We believe more attention should be given to modeling specific user traits, which we address by modeling user profiles as combinations of specialized LMs.

Model Combination User simulators must exhibit diverse conversational patterns to effectively evaluate conversational systems. Model merging techniques (Yadav et al., 2023; Yu et al., 2023; Ilharco et al., 2023) combine models at a weight level, enhancing their capabilities across various NLP tasks. Alternatively, mixture-of-experts approaches (Chen et al., 2024b; Feng et al., 2024; Jiang et al., 2024) use a trainable router in the Transformer (Vaswani et al., 2017) to integrate information from multiple models. In a different vein Sitdikov et al. (2022) uses a classifier to adjust token probabilities, while Shen et al. (2024) applies a classifier to determine when to activate the decoding process of different models. In this work, we propose *mTAD*, a controllable and adaptable method that combines token distributions from various models at decoding time without requiring additional fine-tuning.

3 Adaptable Multi-Trait User Simulators

End-to-end simulators (Lin et al., 2021; Kim and Lipani, 2022) may overlook less common styles and language subtleties due to smoothing or potential forgetting (Luo et al., 2023), while being limited w.r.t. generalization to novel traits. Therefore, we consider a trait-oriented model-based approach, in which specialized trait simulators are created and flexibly combined at decoding time.

3.1 Trait-Specialized User Simulators

Given a dialogue domain $\mathcal{M}$ (e.g. TOD or CTA), we define a conversational trait t_i in a discrete three-intensity level range $l_i \in \{low,\ neutral,\ high\}$. Each trait-intensity pair (t_i, l_i) has an associated dialogue language modeling distribution:

$$P_{\theta_{(t_i,l_i)}}(w_j | w_{<j}, H, \mathcal{M}, (t_i, l_i)), \qquad (1)$$

where w_j is the next token to be generated, H the dialogue history, and $\theta_{(t_i,l_i)}$ the distribution's parameters. Traits can encompass any conversational characteristic measurable across a dialogue, such as cooperativeness and fluency – in Section 4, we introduce the set of traits used in this work. This formulation, allows us to categorize and simulate diverse user traits at different levels of intensity.

Given the set of all traits T, a user profile U is defined as the set of traits:

$$U = \{(t_1, l_1), \dots, (t_{|T|}, l_{|T|})\}. \tag{2}$$

As an example, we can define an *uncooperative* but *fluent* user profile: $U_{e.g.} = \{(t_{cooperativeness}, low),\ (t_{fluency}, high)\}$[1].

A user profile U is thus modeled as the conditional probability distribution:

$$P_{\theta_U}(w_j | w_{<j}, H, \mathcal{M}, \{(t_i, l_i)\}_{i=1}^{i=|T|}), \tag{3}$$

with θ_U being the profile distribution parameters. Our proposed simulator is designed to maximize the expectation $\mathbb{E}[P_{\theta_U}]$ in a zero-shot manner.

3.2 User Simulator LMs

Due to the combinatory nature of user profiles, maximizing the expectation $\mathbb{E}[P_{\theta_U}]$ on demand, for every possible trait-combination, is infeasible. Instead, to provide an adaptable and controllable user simulator, given a target user profile U, we focus on learning individual traits distributions $P_{\theta_{(t_i, l_i)}}$, and combine them at inference time to maximize $\mathbb{E}[P_{\theta_U}]$. This is accomplished with *mTAD*, our proposed zero-shot trait-combination strategy (Section 3.3).

Approximating $P_{\theta_{(t,l)}}$ with a LoRA-based Specialized Trait Simulator (STS). The goal is to have one Specialized-Trait Simulator (*STS*) per trait. This enables each model to capture the subtleties of specific traits, with minimal interference from others, by learning an independent distribution $P_{\theta_{(t_i, l_i)}}$ for each one. In addition, given our focus on delivering adaptable simulators, this strategy makes the incorporation of new traits seamless since each model operates independently.

To model the distribution $P_{\theta_{(t_i, l_i)}}$, we adopt a model-based approach (Lin et al., 2021; Kim and Lipani, 2022), using an LLM to capture each trait. In practice, maximizing the log-likelihood $P_{\theta_{(t_i, l_i)}}$ corresponds to minimizing the LLM's language modeling cross-entropy over trait-specific dialogues.

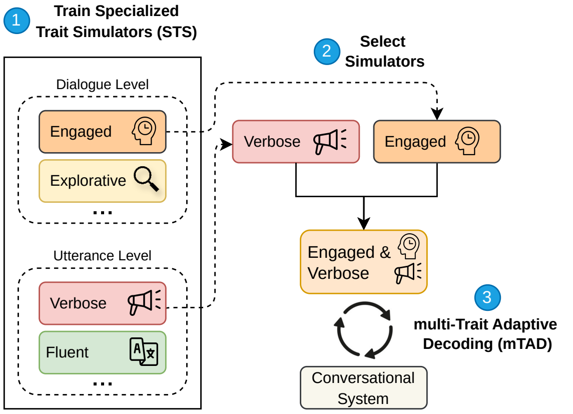


Figure 1: Multi-trait Adaptive Decoding (*mTAD*), leveraging two Specialized Trait Simulators.

Moreover, to efficiently support the learning of multiple independent $P_{\theta_{(t_i, l_i)}}$ distributions, using specialized trait simulators, we use LoRA (Low-Rank Adaptation) (Hu et al., 2022; Dettmers et al., 2023) adapters for each trait. With LoRA, we only update the low-rank matrices in specific layers while keeping the original model weights frozen. This reduces trainable parameters, enabling faster training and lower memory use. These characteristics make LoRA ideal for *mTAD*, allowing diverse trait simulators on a shared LLM backbone at decoding time.

3.3 Approximating P_{θ_U} with Multi-Trait Adaptive Decoding (*mTAD*)

To deliver an on-demand simulator of a user profile U, with a set of traits and corresponding intensities, while avoiding additional fine-tuning, we propose to approximate P_{θ_U} as a combination of independent trait distributions $P_{\theta_{(t_i, l_i)}}$, and implicitly maximize $\mathbb{E}[P_{\theta_U}]$. Namely, we propose **M**ulti-**T**rait **A**daptive **D**ecoding (*mTAD*), to combine multiple user traits at decoding level as the following:

$$P_{\theta_U}(w_j | w_{<j}, H, \mathcal{M}, (t_i, l_i)_{i=1}^{i=|T|}) = \sum_{i=1}^{|T|} \lambda_i \cdot P_{\theta_{(t_i, l_i)}}(w_j | w_{<j}, H, \mathcal{M}, (t_i, l_i)), \tag{4}$$

where λ_i are tunable trait weight parameters, which offer controllability in the profile modeling process, where one can easily select and activate each specialized trait simulator (*STS*).

In practice, each $P_{\theta_{(t_i, l_i)}}$ is modeled by a language model M_i, where, to ensure compatibility, all models M share the same vocabulary. During decoding, at each step, given the distribution P_{θ_U},

---

[1] For simplicity, we omit all traits with *neutral* intensity.

we sample tokens using a given decoding strategy. This implies sampling each specialized trait LM independently for each decoding step, which has minimal requirements due to the selective activation of different LoRA adapters. Figure 1 represents an overview of the *mTAD* framework.

3.4 User Simulator Grounding

To materialize the probability distribution of Eq. 1 and ground the behavior of each *STS*, we define the input sequence:

$$P \oplus H \oplus U \oplus S, \tag{5}$$

where $\oplus$ denotes the concatenation operation, P is an optional preamble, which varies based on the model used (Chiang et al., 2023), H represents the dialogue history, including n previous turns (each consisting of a user and system utterance), U is the user profile, encoded as a unique token sequence, and S is a suffix used to prompt generation.

The learning objective becomes the causal language modeling task, which corresponds to minimizing the loss:

$$\mathcal{L} = -\sum_{j=1}^{N} \log(P_{\theta_{(t_i,l_i)}}(w_j|w_{<j}, P, H, U, S)), \tag{6}$$

where j is the j-th token and N the number of tokens in the response, which comprises both the a user intent and the utterance. The model is trained on this dual-generation task to enhance interpretability by enabling the analysis of intent distribution probabilities. In Appendix C, we present an example of the input and response formats.

4 User Simulation in Conversational Task Assistants (CTAs)

Conversational Task Assistants (CTAs) guide users through tasks such as cooking or DIY (Gottardi et al., 2022; Agichtein et al., 2023). This setting raises a number of challenges (Chan et al., 2023; Choi et al., 2022), that are particularly well addressed by *mTAD*: dialogues have mixed-initiative, users follow a task plan with the aim of completing a manual task, ask explorative questions, and engage in chit-chat as shown in Table 1.

4.1 Real World Conversational Data

To ensure the most representative user simulator, we ground the trait and models on real-world dialogue data. Specifically, we used the dataset from Glória-Silva et al. (2024) composed of 3.6k conversations collected during the Alexa TaskBot Challenge (Gottardi et al., 2022). This dataset is composed of a generated dialogue graph based on intent transitions and ASR transcribed real user utterances. We extend this pipeline to include a diverse profile-aware creation process detailed in Appendix B.

4.2 Modeling User Traits

To define the set of traits (T) characterizing a user profile (U), we carefully analyzed human-system conversations (for statistics, refer to Appendix A), identifying two categories at different levels: 1) *Dialog-level*, and 2) *Utterance-level*.

4.2.1 Dialog-Level Traits

Dialog-Level traits influence the overall progression of a conversation by impacting the probability of transitioning between intents.

Engagement. Reflects the user's willingness to engage with the system for a longer period of time as in (Salle et al., 2021). We measure engagement by the number of turns in the dialogue.

Cooperativeness. Users' tendency to follow the system's instructions and reduce unrelated interactions. It is measured by the probability of in-domain intents in the dialogue. Cooperativeness is a trait also discussed in (Salle et al., 2021; Lei et al., 2022), which we include here for task-guiding dialogues.

Exploration. This trait strikes the balance between exploitation, which involves moving forward in the task, and exploration, which entails engaging with optional system features (Zou et al., 2022). It is measured by the probability of explorative intents (e.g. QA) that differ from the default navigational requests (e.g. "next step").

Tolerance. Represents the user's tolerance for system mistakes, where less tolerant users conclude the interaction sooner (Pearl, 2016). Represented by the tolerance rate, defined as the number of system mistakes tolerated divided by the number of turns.

4.2.2 Utterance-Level Traits

These traits consider a more fine-grained behavior, which given an intent, chooses the style of the utterance according to the user profile.

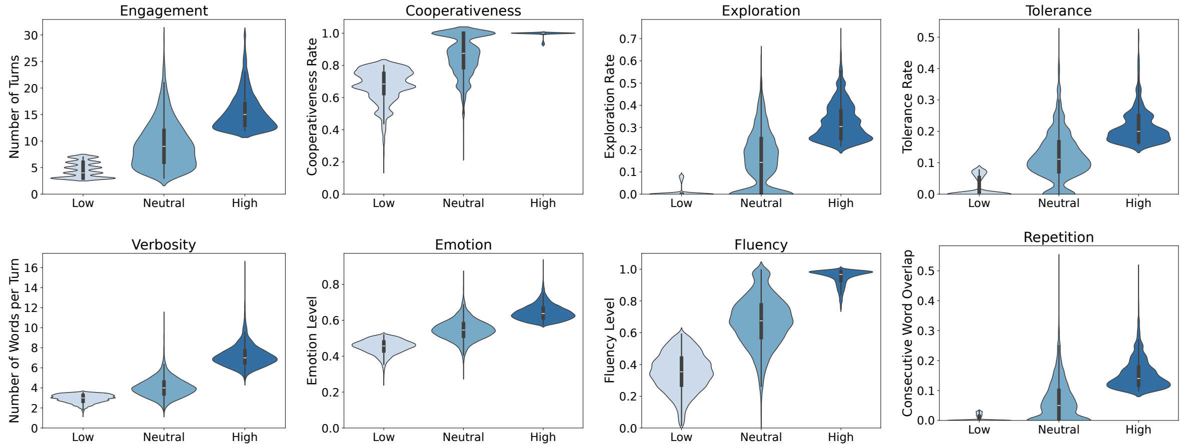


Figure 2: Training data distribution for dialogue-level (top-row) and utterance-level (bottom-row) traits according to their identifying metric (y-axis) across all trait intensities.

Verbosity. Verboseness of the user's requests. Measured by the average number of words in each utterance.

Emotion. Refers to the overall tone (negative or positive) expressed towards the system. Emotion level is measured in a 0-1 scale using the model from (Camacho-Collados et al., 2022).

Fluency. Represents the ability to express oneself clearly and coherently without hesitations, disruptions, or ASR errors. The fluency level is measured in a 0-1 scale using model [2].

Repetition. Consistent use of the same lexicon throughout the dialogue. Measured through word overlap between consecutive utterances.

4.3 Trait Distribution and Intensity

Figure 2 shows the dialogue data distribution per trait and intensity pair. It shows that certain traits exhibit longer tails and have more noticeable differences across intensities, which is expected to have a direct influence on their modeling challenges.

5 Experimental Setup

5.1 Dataset

Tasks and Training Dialogues. We focus on the task execution phase (Gottardi et al., 2022), where the user has an ongoing task such as "baking a cake". We focus on task assistance in the cooking domain ($\mathcal{M}$) and use 1000 unique recipes to ground the dialogue flow. Given these, we randomly sample tasks to generate a total of 20k dialogues with an average of 9.6 turns. Finally, we use a 1000/100/100 dialogue split for each trait-intensity pair and ensure generalization by not sharing tasks between splits.

Simulator Inference. We evaluate various user simulators by interacting with a live system, *Plan-LLM* (Glória-Silva et al., 2024), known for its task-guiding capabilities. For each experiment, we generate 100 dialogues per profile, with interactions ending either by the simulator's side (i.e. the user) or upon reaching a maximum turn limit of 20.

5.2 Models and Baselines.

As the backbone for our simulators, we used the Transformer-based (Vaswani et al., 2017) Mistral-7B (Jiang et al., 2023) and Vicuna-7B (Chiang et al., 2023). The results for Vicuna are in annex and support generalization across LLMs.

We evaluated two types of simulators: the proposed *Specialized Trait Simulators* (***STS***) from Section 3.1, which models each trait independently. And as a strong baseline, we consider a *Joint Trait Simulator* (***JTS***), that jointly learns a distribution for all traits in a single model.[3]

Implementation Details In all experiments, we use LoRA (Hu et al., 2022) adapters and a 4-turn context size. To account for the variability of users, we employ sampling decoding. For detailed information about the training procedure, refer to Appendix D.

Automatic Metrics Evaluating user simulators is challenging (Shi et al., 2019; Li et al., 2022; Zhao et al., 2023) due to the absence of a single correct

---

[2] `https://huggingface.co/gchhablani/bert-base-cased-finetuned-cola`

[3] Code available at `https://github.com/rafaelhferreira/mtad_cta`

response and multiple dialogue directions, making metrics like BLEU (Papineni et al., 2002), and ROUGE (Lin, 2004) unsuitable (Callison-Burch et al., 2006). To address this, we analyze the simulator-generated dialogues using each user profiles' identifying characteristics, as depicted in Figure 2. Namely, we measure each distributions' proximity to the training data using Wasserstein's distance (Vaserstein, 1969) for discrete trait distributions (i.e. engagement and verbosity), and Kolmogorov-Smirnov's (K-S) distance (Massey, 1951) for continuous ones (refer to Appendix D for detailed metrics' definitions). In both, lower values indicate a closer approximation to the reference distributions.

6 Results and Discussion

6.1 Single-Trait Evaluation

In this setting, we create user profiles by adjusting each trait from Section 4.2 to a non-neutral (i.e. *Low* or *High*) intensity level. Additionally, we create a profile called ***Regular***, where all traits are set to *neutral*.[4]

6.1.1 Trait Matching

In Figure 3, we compare the performance of *STS* and *JTS* across trait intensities, aiming for values closer to the reference. As expected, both methods exhibit increasing metrics with trait intensity. However, *JTS* shows a greater deviation from the reference at both *Low* and *High* intensities especially noticeable in utterance-level traits.

In Table 2, we use distance-based metrics to assess the simulators' modeling of the reference distributions. *STS* generally represents trait behaviors more closely, especially at *Low* and *High* intensities, as each model learns specific profiles independently. For some traits, *JTS* represents the *Regular* profile (*neutral* intensity for all traits) closer. This is a result of having been exposed to all profiles during training, resulting in an overall smoother distribution that fails to capture extreme trait-intensity subtleties. See Appendix K for examples of generated dialogues.

Modeling Traits Difficulty. Each trait poses unique modeling challenges. *JTS* struggles with *Low Fluency* since LLMs are trained to generate coherent text. It also struggles with *High Fluency* due to smoothing effects from the *Regular* profile, while STS is able to learn these patterns.

	Low		**Regular**		**High**	
	JTS	**STS**	**JTS**	**STS**	**JTS**	**STS**
Engagement*	0.25	**0.10**	**0.58**	2.18	2.89	**2.70**
Cooperativeness	0.24	**0.16**	**0.10**	0.26	0.17	**0.10**
Exploration	0.26	**0.12**	0.09	**0.07**	0.34	**0.33**
Tolerance	0.66	**0.28**	0.20	**0.12**	0.70	**0.34**
Verbosity*	0.72	**0.17**	**0.12**	0.26	2.10	**0.12**
Emotion	0.62	**0.11**	0.17	**0.06**	0.66	**0.20**
Fluency	0.74	**0.22**	**0.07**	0.15	0.75	**0.27**
Repetition	0.57	**0.35**	0.09	**0.08**	0.69	**0.45**

Table 2: Results for *JTS* and *STS* in all trait intensities considering a single-trait setting. * Wasserstein for discrete and K-S distance for continuous.

Both *JTS* and *STS* find it difficult to model *Low Tolerance* and *Low Repetition*. *Low Tolerance* is rare, as it depends on system errors, which are beyond the control of the simulator. *Low Repetition* is challenging because it requires generating unique utterances each time, whereas models often copy parts of the dialogue history.

In summary, these results show the difficulty in modeling traits at diverse intensities, highlighting the advantage of specialized trait simulators (*STS*) over a joint trait simulator (*JTS*), particularly for the less common and more extreme user patterns.

6.1.2 Generalization to Unseen Domains

To assess how user simulators adapt to new domains $\mathcal{M}$, with out-of-domain tasks, we randomly sampled 100 DIY tasks from WikiHow[5], generated 100 more dialogues per profile, and evaluated whether the trend follows the correct pattern[6].

The results in Table 3 indicate good generalization, showing an upward tendency in all cases. As before, *STS* achieves more pronounced metrics over intensity ranges, demonstrating the simulator's ability to generalize to novel tasks.

6.2 Multi-Trait Combination Evaluation

We now evaluate trait combinations, using *STS* to model individual traits. We defined a total of 14 profiles: 8 with 2 traits, 4 with 3 traits, and 2 with 4 traits (full list in Appendix F). For evaluation purposes, we create dialogs for these combinations and use the same evaluation protocol, generating 100 dialogues for each profile and method.

---

[4] Total of 17 profiles (8 $traits \times 2\ intensities + Regular$)

[5] `https://www.wikihow.com/`

[6] Since the evaluation is out-of-domain, we omit distance-based metrics due to the lack of a reference distribution.

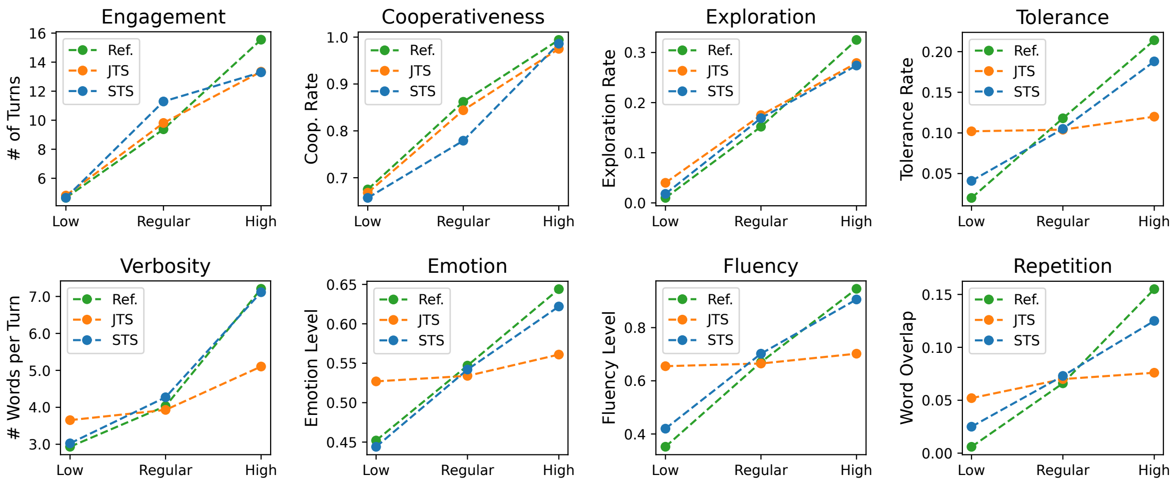


Figure 3: Single-trait results for dialogue-level (top-row) and utterance-level (bottom row) traits across all intensities comparing Reference, Joint Trait Simulator (*JTS*), and Specialized Trait Simulators (*STS*).

	JTS					**STS**				
	Low		**Regular**		**High**	**Low**		**Regular**		**High**
Eng	4.5	<	8.77	<	12.96	5.08	<	9.96	<	12.73
Coop	0.7	<	0.87	<	0.97	0.66	<	0.81	<	0.99
Expl	0.04	<	0.18	<	0.24	0.02	<	0.13	<	0.28
Tol	0.09	<	0.12	<	0.17	0.07	<	0.11	<	0.22
Verb	3.48	<	4.08	<	5.15	3.12	<	4.28	<	7.32
Emot	0.51	<	0.53	<	0.55	0.43	<	0.55	<	0.63
Flu	0.64	<	0.70	<	0.72	0.46	<	0.72	<	0.90
Rep	0.05	<	0.06	<	0.07	0.03	<	0.08	<	0.11

Table 3: Trend analysis in unseen DIY domain, where each value represents its identifying metric. There is an increasing trend across all traits' intensities.

6.2.1 Combination Baselines and Methods

Sampling. This method randomly samples an active user profile at each turn.

***mTAD*.** Combines token probabilities from multiple trait simulators as proposed in Section 3.3. We use a uniform weight distribution (λ) across all active trait simulators.

***mTAD-LA* (Level Aware).** Extension to *mTAD* that generates parts of the response using different models, similar to collaborative decoding (Shen et al., 2024). Specifically, generates the intent using *Dialogue*-level profiles and the utterance using *Utterance*-level profiles.

Focusing on no-training combination methods, we tested weight-level merging (Yu et al., 2023; Yadav et al., 2023). However, outputs did not follow the structure and produced generic utterances. Hence, we exclude them from the results.

6.2.2 Multi-Trait Combination Results

Table 4 presents the average results for all profiles, and grouped by the number of combined traits. *mTAD*-based methods outperform *Sampling*, as they combine multiple models instead of relying on a single one per turn. *Sampling* also produces inconsistent behavior by mimicking different profiles across turns.

Using the *mTAD* framework, it is important to consider the current decoding step, evident by the best performing method *mTAD-LA*, which activates relevant models depending on the current generation step. As the number of traits increases, modeling difficulty also rises due to distribution dilution across profiles. In these experiments, *mTAD* weights were fixed and uniformly split across each profile, in Appendix G, we show *mTAD*'s controllability by varying these weights.

In summary, we show models can be combined at decoding time using *mTAD* without additional training, and that activating models at the appropriate time enhances profile modeling performance.

6.3 Multi-Level Simulator Evaluation

To further validate the simulators' behavior, we use the best methods for *Single-Trait* (*STS*) and *Multi-Trait* (*mTAD-LA*), sampling 10 dialogues from each user profile, resulting in a total of 300 dialogues.

6.3.1 Multi-Level Evaluation Metrics

For turn-level evaluation, we considered the following metrics:

- *Degeneration* - A rule-based binary metric that flags degeneration if: 1) the output does

		Enga*	Coop	Expl	Tol	Verb*	Emot	Flu	Rep
Avg.	Sampling	2.61	0.28	0.29	0.18	0.87	0.26	0.33	0.27
	mTAD	2.76	0.27	0.30	**0.16**	0.81	0.26	0.31	0.26
	mTAD-LA	**2.06**	**0.26**	**0.25**	0.18	**0.48**	**0.24**	**0.30**	**0.20**
2 Traits	Sampling	1.70	0.25	0.26	0.16	0.88	**0.20**	0.28	0.23
	mTAD	1.76	0.25	0.28	**0.14**	0.80	**0.20**	**0.26**	0.22
	mTAD-LA	**1.21**	0.25	**0.22**	0.16	**0.58**	0.21	**0.26**	**0.19**
3 Traits	Sampling	2.27	0.29	0.29	0.21	1.05	0.21	**0.27**	0.35
	mTAD	2.66	0.28	0.31	**0.18**	1.06	0.18	**0.27**	0.34
	mTAD-LA	**1.91**	**0.26**	**0.26**	0.19	**0.38**	**0.17**	0.28	**0.24**
4 Traits	Sampling	6.93	0.36	0.40	0.23	0.47	0.65	0.65	0.28
	mTAD	6.98	0.36	0.36	0.23	0.34	0.65	0.62	0.29
	mTAD-LA	**5.79**	**0.32**	**0.32**	**0.18**	**0.29**	**0.50**	**0.55**	**0.19**

Table 4: Results for combining various *STS* models. * Wasserstein for discrete and K-S distance for continuous.

not have a valid user intent, or 2) special tokens (e.g., speaker labels, *SOS*, *EOS*) appear within the middle of a generated utterance.

- *Uniqueness* - A binary metric that checks if the generated utterance is novel, i.e., it does not appear in the training data, indicating the model's ability to generate original responses.

- *System Response Quality* - A rating from 0 to 2, assessing the quality of the system's response in relation to the ongoing dialogue.

At the dialogue level, we measured *Trait Modeling Accuracy* by comparing each profile's generated dialogue across its defining traits (i.e. $l_i \neq neutral$), with a dialogue from the test set with the same task but an opposite trait intensity, as well as comparing with a *Regular* dialogue.

Based on previous research (Zheng et al., 2023; Rafailov et al., 2023) showing that LLMs align closely with human judgments, we used GPT-4o[7] to annotate 2776 examples for *System Response Quality* and 1040 for *Trait Modeling Accuracy*. To ensure reliability, we conducted a study with 5 human annotators, achieving an average of 83% agreement and 0.67 Fleiss Kappa score with GPT-4o, confirming its suitability for this task (full details and prompts in Appendix I).

6.4 Turn Level Results.

Analyzing turn-level metrics in Table 5, we observe that degenarations are rare (≤ 4%), indicating the methods' ability to generate and combine profiles effectively.

Regarding uniqueness, both methods have similar values, with over 50% of the utterances not appearing in the training set, demonstrating good generalization to other requests and tasks.

	Single-Trait	Multi-Trait
Degeneration	0.03	0.04
Unique Utterances	0.48	0.46
System RQ (0-2)	1.46	1.43

Table 5: Turn-level metrics for Degeneration, Unique utterances and System Response Quality.

The system's response quality is generally good with the model achieving an average score over 1.4. A closer analysis by trait revealed that the system achieves higher response quality scores (≥1.57) handling higher *fluency*, *exploration*, and *emotion*, while scoring lower (≤ 1.34) for low *engagement*, *tolerance* and high *verbosity*. These results allow to understand system limitation and in turn create more robust systems, as it has been shown that simulator variability can improve system performance (Liu et al., 2023; Tang et al., 2021).

6.5 Trait Modeling Accuracy Results.

Table 6 presents the results of trait modeling accuracy. On average, we observe a small drop in performance when moving from single to multi-trait settings. As expected, model accuracy is higher when comparing opposite intensity traits, due to the more pronounced differences between them compared to a *Regular* dialogue. Performance significantly declines in the multi-trait setting with *Regular* dialogues, as the model must integrate multiple profiles, resulting in a smoothing effect that makes it harder to distinguish traits compared to the single-trait setting.

Examining individual traits, modeling low *verbosity* and high *emotion* is particularly challeng-

---

[7] gpt-4o-2024-05-13

	Single-Trait	**Multi-Trait**
Average Comparison	0.73	0.70
Comparison w/ Opposite	0.76	0.78
Comparison w/ Regular	0.70	0.62

Table 6: Trait Modeling Accuracy Average, and considering only comparisons between a profile and its opposite intensity profile or a *Regular* profile.

ing since their *Regular* values are similar to these traits. Conversely, *engagement* is easier to model because it relies on the number of turns, and low *emotion* is detectable due to utterances indicating low motivation. Full results by trait are provided in Appendix J.

In summary, the results show good performance in both single-trait and multi-trait settings, allowing for diverse simulators that can be effectively combined at decoding time.

7 Conclusions

This paper addresses the challenge of simulating diverse user traits and effectively combining them in a conversational setting. Using real-world data to identify user traits, we developed a framework to generate profile-aware conversations, being one of the first to deliver user simulators for the Conversational Task Assistance (CTA) setting.

Our results demonstrate the need for specialized simulators for each trait (*STS*), which produce a more accurate trait adherence when compared to jointly learning all traits with a single model. The latter converges to highly smoothed trait distributions that tend to overlook subtle trait characteristics. With *STS*, trait-specific characteristics are better preserved, including in unseen domains with different tasks, highlighting its generalizability.

To flexibly combine diverse user traits and profiles without extra fine-tuning, we proposed Multi-Trait Adaptive Decoding (*mTAD*). The results show that *mTAD* effectively combines multiple user traits. This shows that our framework provides an adaptable and controllable approach to create user simulators that can accommodate a wide range of profiles and tasks.

Limitations

Our analysis focuses on a particular set of conversational traits. Given the versatility and broadness of this setting, additional traits could be considered to have a more comprehensive user simulator. Additionally, given the combinatorial nature of possible trait combinations, despite our approach supporting it, we did not exhaustively explore all combinations of the various traits. Furthermore, our simulator focuses on the task execution level, leaving room for future work in the retrieval and grounding processes of Conversational Task Assistants (Chan et al., 2023).

To conclude, we believe our work presents a step forward in creating adaptable user simulators, however, subsequent studies should broaden the scope to allow for a more comprehensive understanding of user simulation dynamics.

Ethical Considerations

All human interactions considered in this study were obtained voluntarily, with users having the ability to withdraw at any point. Prior to each interaction with the system, users were informed that all information would be saved and shared with the authors. Subsequently, users were given the option to proceed with the interaction, indicating their understanding of the terms and conditions. Moreover, the responses collected were anonymous and devoid of user demographics. Additionally, we conducted a thorough review of the data to ensure that no personal information was included.

Regarding the annotators, all individuals volunteered and provided consent to participate in the experiment. They were fully informed about the study and its implications.

Acknowledgements

This work has been partially funded by the Amazon Science - TaskBot Prize Challenge 2021 and 2022, by the NOVA LINCS project Ref. UIDP/04516/2020 and by FCT Ref. UI/BD/151261/2021.

References

Jafar Afzali, Aleksander Mark Drzewiecki, Krisztian Balog, and Shuo Zhang. 2023. Usersimcrs: A user simulation toolkit for evaluating conversational recommender systems. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 - 3 March 2023*, pages 1160–1163. ACM.

Eugene Agichtein, Michael Johnston, Anna Gottardi, Lavina Vaz, Cris Flagg, Yao Lu, Shaohua Liu, Sattvik Sahai, Giuseppe Castellucci, Jason Ingyu Choi, Prasoon Goyal, Di Jin, Saar Kuzi, Nikhita Vedula, Lucy Hu, Samyuth Sagi, Luke Dai, Hangjie Shi, Zhejia

Yang, Desheng Zhang, Chao Zhang, Daniel Pressel, Heather Rocker, Leslie Ball, Osman Ipek, Kate Bland, James Jeun, Oleg Rokhlenko, Akshaya Iyengar, Arindam Mandal, Yoelle Maarek, and Reza Ghanadan. 2023. Advancing conversational task assistance: the second alexa prize taskbot challenge. In *Alexa Prize TaskBot Challenge 2 Proceedings*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of bleu in machine translation research. In *11th conference of the european chapter of the association for computational linguistics*, pages 249–256.

José Camacho-Collados, Kiamehr Rezaee, Talayeh Riahi, Asahi Ushio, Daniel Loureiro, Dimosthenis Antypas, Joanne Boisson, Luis Espinosa Anke, Fangyu Liu, and Eugenio Martínez Cámara. 2022. Tweetnlp: Cutting-edge natural language processing for social media. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022 - System Demonstrations, Abu Dhabi, UAE, December 7-11, 2022*, pages 38–49. Association for Computational Linguistics.

Szeyi Chan, Jiachen Li, Bingsheng Yao, Amama Mahmood, Chien-Ming Huang, Holly Jimison, Elizabeth D. Mynatt, and Dakuo Wang. 2023. "mango mango, how to let the lettuce dry without A spinner?": Exploring user perceptions of using an llm-based conversational assistant toward cooking partner. *CoRR*, abs/2310.05853.

Shaoxiang Chen, Zequn Jie, and Lin Ma. 2024a. Llava-mole: Sparse mixture of lora experts for mitigating data conflicts in instruction finetuning mllms. *CoRR*, abs/2401.16160.

Shaoxiang Chen, Zequn Jie, and Lin Ma. 2024b. Llava-mole: Sparse mixture of lora experts for mitigating data conflicts in instruction finetuning mllms. *CoRR*, abs/2401.16160.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Jason Ingyu Choi, Saar Kuzi, Nikhita Vedula, Jie Zhao, Giuseppe Castellucci, Marcus Collins, Shervin Malmasi, Oleg Rokhlenko, and Eugene Agichtein. 2022. Wizard of tasks: A novel conversational dataset for solving real-world tasks in conversational settings. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 3514–3529. International Committee on Computational Linguistics.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *CoRR*, abs/2305.14314.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. 2024. Mixture-of-loras: An efficient multitask tuning for large language models. *CoRR*, abs/2403.03432.

Diogo Glória-Silva, Rafael Ferreira, Diogo Tavares, David Semedo, and João Magalhães. 2024. Plan-grounded large language models for dual goal conversational settings. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 1271–1292. Association for Computational Linguistics.

Anna Gottardi, Osman Ipek, Giuseppe Castellucci, Shui Hu, Lavina Vaz, Yao Lu, Anju Khatri, Anjali Chadha, Desheng Zhang, Sattvik Sahai, Prerna Dwivedi, Hangjie Shi, Lucy Hu, Andy Huang, Luke Dai, Bofei Yang, Varun Somani, Pankaj Rajan, Ron Rezac, Michael Johnston, Savanna Stiff, Leslie Ball, David Carmel, Yang Liu, Dilek Hakkani-Tur, Oleg Rokhlenko, Kate Bland, Eugene Agichtein, Reza Ghanadan, and Yoelle Maarek. 2022. Alexa, let's work together: Introducing the first alexa prize taskbot challenge on conversational task assistance. In *Alexa Prize TaskBot Challenge Proceedings*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

To Eun Kim and Aldo Lipani. 2022. A multi-task based neural model to simulate users in goal oriented dialogue systems. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 2115–2119. ACM.

Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive path reasoning on graph for conversational recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 2073–2083. ACM.

Wenqiang Lei, Yao Zhang, Feifan Song, Hongru Liang, Jiaxin Mao, Jiancheng Lv, Zhenglu Yang, and Tat-Seng Chua. 2022. Interacting with non-cooperative user: A new paradigm for proactive dialogue policy. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 212–222. ACM.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9748–9758.

Xiujun Li, Zachary C. Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *CoRR*, abs/1612.05688.

Zekun Li, Wenhu Chen, Shiyang Li, Hong Wang, Jing Qian, and Xifeng Yan. 2022. Controllable dialogue simulation with in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4330–4347. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Hsien-Chin Lin, Nurul Lubis, Songbo Hu, Carel van Niekerk, Christian Geishauser, Michael Heck, Shutong Feng, and Milica Gasic. 2021. Domain-independent user simulation with transformers for task-oriented dialogue systems. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2021, Singapore and Online, July 29-31, 2021*, pages 445–456. Association for Computational Linguistics.

Yajiao Liu, Xin Jiang, Yichun Yin, Yasheng Wang, Fei Mi, Qun Liu, Xiang Wan, and Benyou Wang. 2023. One cannot stand for everyone! leveraging multiple user simulators to train task-oriented dialogue systems. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1–21. Association for Computational Linguistics.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *CoRR*, abs/2308.08747.

Frank J. Massey. 1951. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78.

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, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Cathy Pearl. 2016. *Designing voice user interfaces: Principles of conversational experiences*. " O'Reilly Media, Inc.".

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *CoRR*, abs/2305.18290.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8689–8696. AAAI Press.

Alexandre Salle, Shervin Malmasi, Oleg Rokhlenko, and Eugene Agichtein. 2021. Studying the effectiveness of conversational search refinement through user simulation. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part I*, volume 12656 of *Lecture Notes in Computer Science*, pages 587–602. Springer.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve J. Young. 2007. Agenda-based user simulation for bootstrapping a POMDP dialogue system. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA*, pages 149–152. The Association for Computational Linguistics.

Shannon Zejiang Shen, Hunter Lang, Bailin Wang, Yoon Kim, and David A. Sontag. 2024. Learning to decode collaboratively with multiple language models. *CoRR*, abs/2403.03870.

Weiyan Shi, Kun Qian, Xuewei Wang, and Zhou Yu. 2019. How to build user simulators to train rl-based dialog systems. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1990–2000. Association for Computational Linguistics.

Askhat Sitdikov, Nikita Balagansky, Daniil Gavrilov, and Alexander Markov. 2022. Classifiers are better experts for controllable text generation. *CoRR*, abs/2205.07276.

Zhiwen Tang, Hrishikesh Kulkarni, and Grace Hui Yang. 2021. High-quality diversification for task-oriented dialogue systems. *CoRR*, abs/2106.00891.

Leonid Nisonovich Vaserstein. 1969. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredachi Informatsii*, 5(3):64–72.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Xun Wu, Shaohan Huang, and Furu Wei. 2024. Mixture of lora experts.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *CoRR*, abs/2311.03099.

Shuo Zhang and Krisztian Balog. 2020. Evaluating conversational recommender systems via user simulation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1512–1520. ACM.

Kesen Zhao, Shuchang Liu, Qingpeng Cai, Xiangyu Zhao, Ziru Liu, Dong Zheng, Peng Jiang, and Kun Gai. 2023. Kuaisim: A comprehensive simulator for recommender systems. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *CoRR*, abs/2306.05685.

Jie Zou, Jimmy Huang, Zhaochun Ren, and Evangelos Kanoulas. 2022. Learning to ask: Conversational product search via representation learning. *ACM Trans. Inf. Syst.*, 41(2).

	Trait	**Identifying Characteristic**
Dialogue	Engagement	Number of Turns
	Cooperativeness	Cooperativeness Rate
	Tolerance	Tolerance Rate
	Exploration	Exploration Rate
Utterance	Verbosity	Number of Words per Turn
	Emotion	Emotion Level
	Fluency	Fluency Level
	Repetition	Consecutive Word Overlap

Table 7: Dialogue and Utterance Level traits and corresponding characteristic.

A Real-world Data Analysis

We leverage the interaction data from Glória-Silva et al. (2024) composed of real-world interactions collected in the context of the Alexa Prize TaskBot Challenge (Gottardi et al., 2022), where a user interacts with an Alexa device by voice to complete a manual task in the cooking or DIY domain. We focus on conversations with at least three turns that reached the start of a task, and we identified traits at two levels: *Dialogue*, controlling the flow using intents, and *Utterance*, representing the way users express these intents. Table 7 provides a summarized overview of the identified traits and their identifying characteristic.

Dialogue Level Figure 4 shows the statistics for dialogue-level traits. *Engagement* predictably decreases as the number of turns increases, with most conversations being short as users explore the system or find tasks misaligned with their goals. Most users exhibit high *cooperativeness*, close to 100%, since they primarily follow the task by stating "next". Without "next" intents, cooperativeness drops from 94% to 60%. Similarly, users show a strong tendency toward task *exploration* going deeper into the task, however, by excluding "next" intents, explorativeness decreases from 92% to 57%. Finally, *tolerance* is analyzed in sessions with at least one system error or an unrecognized request, showing users typically endure at least one error, with a maximum tolerance of seven errors.

Utterance Level Figure 5 illustrates utterance-level traits. Regarding *verbosity*, users predominantly employ short, straightforward utterances focused on continuing the task. This observation shows the disparity between natural-sounding utterances, as depicted in (Choi et al., 2022), and real-world user interactions, emphasizing the importance of utilizing authentic user data over crowd-sourced alternatives. Most utterances exhibit mid-range *emotion* levels, aligning with the nature of the task. Instances of lower and higher emotional sentiment generally indicate user frustration or satisfaction. *Fluency* is generally high, influenced by the prevalence of short utterances. Lastly, *repetition* analysis shows minimal consecutive word overlap, indicating diverse vocabulary usage, though occasional redundancy is observed during task navigation.

B Dataset Creation Details

B.1 Conversational flow

Following the approach of Glória-Silva et al. (2024), we utilized the same dataset of 3.6k real-world interactions from the Alexa Prize TaskBot Challenge (Gottardi et al., 2022). Using this data, we generated a directed graph based on intent transition probabilities, detailed in Table 8.

B.2 System Responses

For the system responses, we utilize the framework provided by Glória-Silva et al. (2024), which is composed of both template-based and contextual LLM-generated responses.

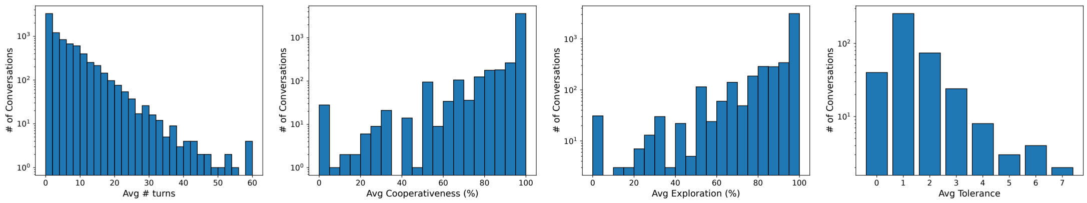
B.3 User Utterances

Regarding user utterances, these are obtained from the dataset in (Glória-Silva et al., 2024), where each of the top-100 utterances was manually classified into an intent, which we further verified for personally identifiable information.

This process resulted in a dataset grounded in reality, as users interacted voluntarily and naturally with the system. Consequently, the dataset includes various instances of noisy utterances (e.g. ASR errors and arbitrary requests), enabling the simulator to capture these real-world patterns.

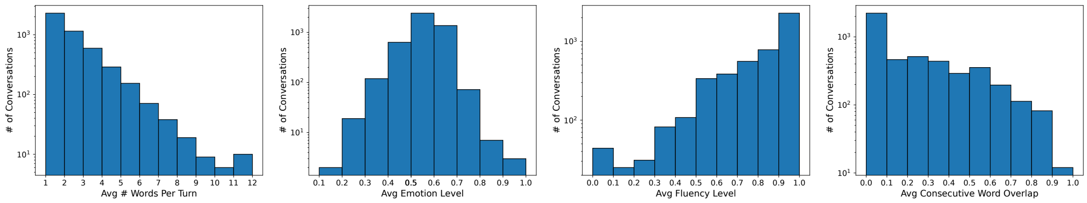
B.3.1 Profile-Aware Dataset Creation Pipeline

Our aim is to generate dialogues given a specific user profile. This user profile, as defined in Section 3.1, is composed of pairs of traits (Table 7) and intensities (*low*, *neutral* and *high*). Algorithm 1 details our method for incorporating this information during dataset generation: 1) apply the user profile to the intent transitions, 2) sample an intent, 3) change weights of utterances according to the user profile, and 4) sample an utterance.



(a) Engagement (b) Cooperativeness (c) Exploration (d) Tolerance

Figure 4: Real user conversations statistics for dialogue level traits in the Alexa TaskBot Dataset.



(a) Verbosity (b) Emotion (c) Fluency (d) Repetition

Figure 5: Real user conversations statistics for dialogue level traits in the Alexa TaskBot Dataset.

	Intent	**Description**	**Example**	**Stop**	**Expl.**	**Coop.**
Navigational	Start	User asks to start the task.	please start			
	NextStep	User asks to go to the next step.	next step		✓	✓
	PreviousStep	User asks to go to the previous step.	previous step			✓
	Resume	User asks the system to continue the current step.	resume			✓
	Repeat	User asks the system to repeat the previous response.	repeat that			✓
	Stop	User ends the interaction.	finish this task	✓		✓
Open QA	Question	User asks a recipe-related question.	how much salt do i need		✓	✓
	Definition	User asks for an explanation of a concept.	what is a spatula		✓	✓
	Replacement	User asks for replacements of a tool/ingredient.	i do not have sugar		✓	✓
	GetFunFact	User asks for a fun fact.	tell me a fun fact		✓	✓
Other	NewTask	User asks for a new unrelated task.	how to change a tire			
	ChitChat	The basic conversation norms, e.g., thanks, chitchat.	how are you today			
	Sensitive	Dangerous or inappropriate requests (answer is denied).	how to make a nuke			
	Fallback	Any request where user intention is not clear or OOD.	find restaurants near me			

Table 8: Intents list, description, and example utterances. We categorize intents into the groups: "Stop", "Exploration", and "Cooperative", which influence their occurrence probability when creating profile-aware dialogues.

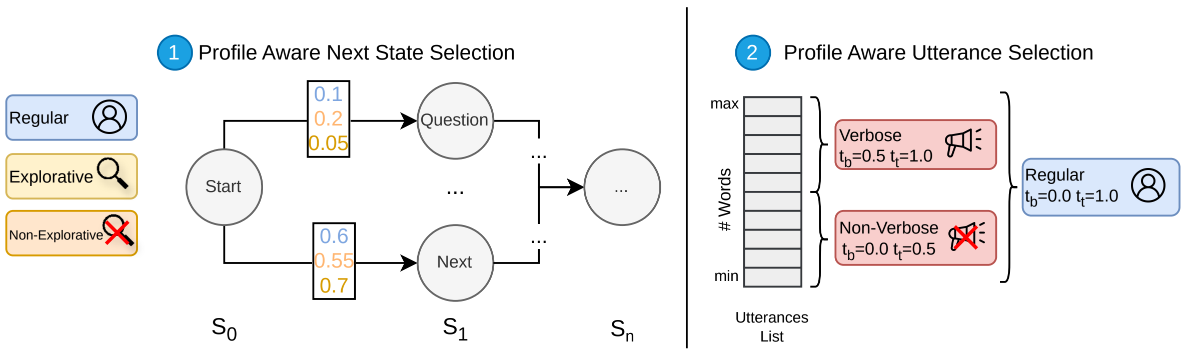


Figure 6: User profile application stages. First step is to choose the next state (dialogue-level) which has different probabilities for state transitions depending on the profile. In a second stage after choosing a state the goal is too choose an utterance (utterance-level) according to the user profile.

Algorithm 1 Profile-Aware Dialogue Creation Algorithm

```
Require:
  $\mathcal{T}$: List of tasks
  $u$: User profile
  $n$: Turn limit
  $\mathcal{S}$: State transition dictionary
  $\mathcal{U}_s$: Utterance list for each state $s \in \mathcal{S}$
  $apply\_dialogue\_level\_traits$: Function that applies user profile to state transition dict
  $apply\_utterance\_level\_traits$: Function that applies user profile to utterances list
  $get\_system\_utterance$: Function that returns a system response given state, utterance and task
Ensure:
  $\mathcal{D} \leftarrow []$: Current Dialog
  $s_t \leftarrow Start$: Current state
  Randomly select a task: $t \sim \mathcal{T}$
  Apply user profile to state dictionary: $\mathcal{S}'_t = apply\_dialogue\_level\_traits(u, \mathcal{S})$
  while $n > 0$ and $s_t$ != $Stop$ do
      Update and Sample state: $s_t \sim \mathcal{S}'_t$
      Apply user profile to utterance list: $\mathcal{U}'_{s_t} = apply\_utterance\_level\_traits(u, \mathcal{U}_{s_t})$
      Sample utterance: $u_t \sim \mathcal{U}'_{s_t}$
      Sample system response and new state $r_t, s_t = get\_system\_utterance(s_t, u_t, t)$
      Update state: $s_t \sim \mathcal{S}'_t(s_t)$
      Add turn to dialog: $\mathcal{D} \leftarrow \mathcal{D} + (s_t, u_t, r_t)$
      Decrement turn limit: $n \leftarrow n - 1$
  end while
  return $\mathcal{D}$
```

B.3.2 Dialogue Level Traits Integration

Given a profile, the transitions between intents are modified (Figure 6 left).

Generically, the probability $P(i|i_c)$ of transitioning to an intent i given the current intent i_c is computed as:

$$P(i|i_c) = \begin{cases} P(i|i_c) \cdot f, & \text{if } i \in Ints \\ P(i|i_c), & \text{otherwise} \end{cases} \tag{7}$$

where f is a factor that increases or decreases the probability of that intent occurring which is only applied if the intent i is part of a particular set of considered intents $Ints$.

Engagement. We multiply the probability of the set of stop intents ($Ints = Stop$) by a factor f, which in turn decreases or increases the probability of stopping early.

Cooperativeness. This is modeled in the same way as Engagement (Eq. 7) but using a different factor f and the set of uncooperative intents ($Ints \neq Coop.$).

Tolerance. We follow a similar approach by increasing the probability of intents in the (*Stop*) set when a system error occurs (e.g. incorrect answer, refusal to answer, fallback response):

$$P(i|i_c) = P(i|i_c) \cdot f^{(\#\ errors)},\ i \in Stop \tag{8}$$

where, $\#\ errors$ denotes the current number of errors during the dialogue. Tolerance is designed to impose a more significant penalty for system mistakes. During dialogue generation, we intentionally select incorrect system responses a percentage of the time to simulate errors.

Exploration. We calculate the sum of the probabilities for the top-k intents and for the explorative intents (*Expl.*) not in the *top-k*:

$$P_{\text{top-k}} = \sum_{i \in \text{top-k}} P_i \quad ; \quad P_E = \sum_{i \in (Expl - \text{top-k})} P_i. \tag{9}$$

We then remove part of the probability from the *top-k* intents, and share it with the others:

$$P(i|i_c) = \begin{cases} P(i|i_c) - (P_{\text{top-k}} * f) * (P(i|i_c)/P_{\text{top-k}}), \\ \quad \text{if } i \in \text{top-k} \\ P(i|i_c) + (P_{\text{top-k}} * f) * (P(i|i_c)/P_{Expl}), \\ \quad \text{if } i \in Expl - \text{top-k} \\ P(i|i_c), \\ \quad \text{otherwise} \end{cases} \tag{10}$$

We present the hyperparameters for dialogue-level traits in Table 9.

B.3.3 Utterance-Level Traits Integration

We change the weights of choosing an utterance according to a user profile.

Verbosity, Sentiment, and Fluency. For each trait, we first attribute a value to each utterance, considering a specific function F. For verbosity, the function (F) used is the number of words, while for sentiment and fluency, we use the Transformer-based models from [8] and [9], respectively, which outputs we transform into a 0-1 scale. After having these values for all the utterances, we apply a bottom-p (t_b) and top-p (t_t) threshold to select from this pool (Figure 6 right).

Repetition For repetition, we use word-overlap. This involves two probabilities: the likelihood of selecting a previously mentioned utterance via an exact match (r_e) and the likelihood of selecting utterances with some level of overlap (r_o). If neither option applies (e.g. dialogue start), we sample from all available utterances without constraints.

We present the hyperparameters for utterance-level traits in Table 10.

Profile	**Intents Considered (I)**	**Factor (f)**
Engagement=Low	*Stop*	2.0
Engagement=High	*Stop*	0.5
Cooperativeness=Low	$I \notin \mathit{Coop.}$	2..0
Cooperativeness=High	$I \notin \mathit{Coop.}$	0.5
Exploration=Low	*Expl.*	0.2 (other to top-1 intents)
Exploration=High	*Expl.*	0.2 (top-1 to other intents)
Tolerance=Low	*Stop*	10.0
Tolerance=High	*Stop*	1.0
All Other Profiles	-	-

Table 9: Dialogue-level parameters used to create the dialogues for a particular type. “All Other Profiles” indicates the parameters used to create the profiles not attached to that particular metric.

B.4 Dialogue Filtering

To guarantee the dialogues have diversity and follow the intended behavior, we first generate 10k dialogues for the *Regular* profile to collect average statistics. After this, we apply half a standard deviation, up for *high*, and down for *low* intensities to obtain dialogues within the required characteristics.

Profile	**Metric**	**Bottom Threshold (t_b)**	**Top Threshold (t_t)**
Verbosity=Low	Avg # Words	0.0	0.5
Verbosity=High	Avg # Words	0.5	1.0
Emotion=Low	Avg Emotion	0.0	0.5
Emotion=High	Avg Emotion	0.5	1.0
Fluency=Low	Avg Fluency	0.0	0.5
Fluency=High	Avg Fluency	0.5	1.0
All Other Profiles	(All Above)	0.0	1.0
Profile	-	**Exact Match (r_e)**	**Overlap Match (r_o)**
Repetition=Low	-	0.0	0.0
Repetition=High	-	1.0	1.0
All Other Profiles	-	0.15	0.15

Table 10: Utterance-level parameters used to create the dialogues for a particular type. “All Other Profiles” indicates the parameters used to create the profiles not attached to that particular metric.

For a summarized example showing the distinction between profiles, see Table 11 and 12 for dialogue and utterance-level profiles, respectively.

Intent	**Utterance**
Profile:	**Exploration High**
Start	i want to start the task
Next	next
Question	how much sugar do i need
Next	next step
Replacement	i do not have orange juice
Profile:	**Exploration Low**
Start	start task
Next	next
Next	next
Next	next step
stop	stop now
Profile:	**Cooperativeness High**
Start	start task
Next	next step
Next	i’m done
Curiosity	tell me a fun fact
Next	next
Profile:	**Cooperativeness Low**
Start	start
Sensitive	you are stupid
Next	next
Fallback	what can you see
Stop	turn off

Table 11: Dialogue-level example transitions for various simulators. System answers are omitted.

B.5 Dataset Statistics

Table 13 provides a comprehensive overview of the generated data. Profiles effectively capture distinct traits, as seen from the range of values achieved. After filtering, traits like *Fluency* and *Verbosity* retain most dialogues, while most *Tolerance* dialogues are filtered given these rely on errors trig-

---

[8] https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest

[9] https://huggingface.co/gchhablani/bert-base-cased-finetuned-cola

Recipe: Easy Taco Pie

Intent: Start

Verbosity High: i want to start the task
Verbosity Low: start
Fluency High: start the task
Fluency Low: start cats
System: Let's start! Step 1: Preheat the oven to 350ºF (180 degrees C).

Intent: Next

Verbosity High: i am done what's next
Verbosity Low: next
Fluency High: next step please
Fluency Low: next p
System: Step 2: Heat up your ground Lean Ground Beef in a frying pan on stove ...

Intent: Stop

Verbosity High: can you stop
Verbosity Low: stop
Fluency High: let's stop
Fluency Low: hey off
System: Happy to help! See you again soon!

Table 12: Utterance-level example excerpts for various simulators.

gered by the system. Trait interactions show behaviors like *Low Exploration* and *Low Tolerance* leading to shorter dialogues. In contrast, *High Patience* and *High Tolerance* profiles explore more compared to *Low Patience*. Regarding utterance-level traits, we see minimal effects on each other.

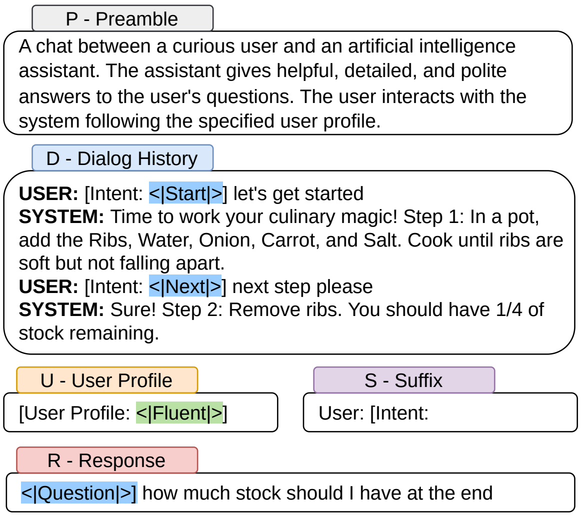
Overall, these findings demonstrate how different user profiles impact various aspects of dialogues, including intents, utterances, and dialogue flow.

C Model Input Format

The prompt used is shown in Figure 7, which consists of four components:

- **Preamble** - Establishes the context and defines the task.
- **Dialogue History** - Includes the previous n turns of dialogue, comprising both user intent and utterance, and system responses.
- **User Profile**: Specifies the profile for the model to follow, identified by a unique token sequence.
- **Suffix** - A suffix that prompts the model to generate a response, since the loss should not be calculated over these tokens.

The model is trained in a dual-task, generating responses with both an intent and a user utterance, enhancing interpretability and controllability by allowing analysis of intent probabilities and injection of specific intents.



A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful, detailed, and polite answers to the user's questions. The user interacts with the system following the specified user profile.

USER: [Intent: <|Start|>] let's get started
SYSTEM: Time to work your culinary magic! Step 1: In a pot, add the Ribs, Water, Onion, Carrot, and Salt. Cook until ribs are soft but not falling apart.
USER: [Intent: <|Next|>] next step please
SYSTEM: Sure! Step 2: Remove ribs. You should have 1/4 of stock remaining.

[User Profile: <|Fluent|>]

User: [Intent:

<|Question|>] how much stock should I have at the end

Figure 7: Model input and its various components. We also highlight the intents and the profile. The simulator should generate the response (R).

D Implementation Details

Handling Profile Imbalance. Profiles exhibit varying average numbers of turns, causing imbalance. To address this, we use a stratified approach, ensuring an equal number of samples for training each user profile.

Handling Intent Class Imbalance. There exists a significant imbalance in intent distribution, with the dominant intent, *NextStep*, representing 37% of interactions. To avoid overemphasizes on this intent, we undersample *NextStep* by 50% in the training dataset, resulting in a more balanced intent distribution.

Libraries. We used Pytorch (Paszke et al., 2019) and the Huggingface library (Wolf et al., 2019).

Training Details and Hyperparameters. For all runs, we trained a Low-Rank adapter (LoRa) (Hu et al., 2022) on the embedding, query, key, value, and language modeling head layers, using bf16 precision and 4 turns of dialogue history. Other hyperparameters are in Table 14. For training, we use the cross-entropy loss and select the best model based on the lowest loss in validation.

Hardware and Training Time. For training, we use an A100-40GB GPU. Regarding training times, the Joint Trait Simulator (*JTS*) model takes around

Trait	Intensity	# Turns	Coop. Rate	Expl. Rate	Tolerance Rate	# Words	Emotion	Fluency	Word Overlap	# Dial
Regular	-	9.38 ± 4.4	0.86 ± 0.12	0.15 ± 0.13	0.12 ± 0.08	4.03 ± 0.96	0.55 ± 0.06	0.67 ± 0.16	0.07 ± 0.07	9000
Engagement	Low	4.65 ± 1.44	0.86 ± 0.16	0.08 ± 0.12	0.14 ± 0.1	3.83 ± 1.07	0.55 ± 0.08	0.68 ± 0.2	0.04 ± 0.06	5731
	High	**15.55** ± **3.51**	0.87 ± 0.09	0.2 ± 0.11	0.09 ± 0.05	4.18 ± 0.77	0.55 ± 0.04	0.65 ± 0.11	0.08 ± 0.05	3911
Cooperativeness	Low	9.71 ± 5.0	0.68 ± 0.1	0.12 ± 0.11	0.14 ± 0.09	4.1 ± 1.01	0.55 ± 0.06	0.7 ± 0.15	0.05 ± 0.05	5510
	High	8.9 ± 4.12	**0.99** ± **0.02**	0.17 ± 0.14	0.09 ± 0.07	3.96 ± 0.9	0.55 ± 0.06	0.64 ± 0.16	0.09 ± 0.08	5222
Exploration	Low	7.6 ± 3.95	0.84 ± 0.14	0.01 ± 0.02	0.12 ± 0.08	3.56 ± 0.89	0.56 ± 0.06	0.67 ± 0.18	0.08 ± 0.08	3830
	High	10.58 ± 4.5	0.88 ± 0.1	**0.32** ± **0.08**	0.12 ± 0.08	4.6 ± 0.88	0.53 ± 0.05	0.67 ± 0.15	0.06 ± 0.06	3543
Tolerance	Low	9.9 ± 5.09	0.86 ± 0.12	0.18 ± 0.12	0.02 ± 0.03	4.12 ± 0.94	0.53 ± 0.05	0.69 ± 0.14	0.06 ± 0.06	1800
	High	10.96 ± 5.83	0.78 ± 0.13	0.2 ± 0.13	**0.21** ± **0.05**	4.14 ± 0.94	0.54 ± 0.05	0.68 ± 0.15	0.06 ± 0.06	1122
Verbosity	Low	9.15 ± 4.26	0.87 ± 0.12	0.12 ± 0.11	0.12 ± 0.08	2.93 ± 0.4	0.56 ± 0.05	0.66 ± 0.16	0.06 ± 0.07	6214
	High	9.31 ± 4.32	0.86 ± 0.12	0.15 ± 0.13	0.12 ± 0.08	**7.21** ± **1.13**	0.53 ± 0.07	0.62 ± 0.16	0.1 ± 0.07	8992
Emotion	Low	9.53 ± 4.32	0.86 ± 0.12	0.16 ± 0.13	0.11 ± 0.08	4.26 ± 0.93	0.45 ± 0.04	0.68 ± 0.15	0.07 ± 0.06	8206
	High	9.55 ± 4.41	0.86 ± 0.12	0.15 ± 0.12	0.12 ± 0.07	3.82 ± 0.91	**0.64** ± **0.04**	0.66 ± 0.15	0.07 ± 0.07	8115
Fluency	Low	9.55 ± 4.38	0.87 ± 0.12	0.15 ± 0.12	0.12 ± 0.07	3.92 ± 0.98	0.56 ± 0.06	0.35 ± 0.12	0.07 ± 0.07	8318
	High	9.26 ± 4.36	0.86 ± 0.12	0.15 ± 0.13	0.12 ± 0.08	4.14 ± 0.89	0.54 ± 0.06	**0.95** ± **0.05**	0.07 ± 0.07	8918
Repetition	Low	8.37 ± 4.0	0.85 ± 0.13	0.14 ± 0.12	0.12 ± 0.08	3.83 ± 0.87	0.55 ± 0.06	0.68 ± 0.15	0.01 ± 0.01	5224
	High	10.02 ± 4.34	0.9 ± 0.11	0.13 ± 0.13	0.11 ± 0.07	3.94 ± 0.94	0.55 ± 0.05	0.65 ± 0.15	**0.16** ± **0.05**	3103

Table 13: Training data statistics. For each metric, maximum values are bold, and minimum values are underlined.

Parameter	Value
Batch Size	4
Grad Acc.	4
Epochs	15
Max Input Size	1024
Model DType	bf16
Learning Rate	$1 \cdot 10^{-5}$
Scheduler	Linear
Optimizer	AdamW
LoRa Rank	16
LoRa α	32
LoRa Dropout	0.1

Table 14: Training Hyperparameters.

20 hours, while each Specialized Trait Simulator (*STS*) takes takes around 2 hours.

Simulator Inference. For inference, we use an A100-40GB GPU to load the user simulator model and the system model (Glória-Silva et al., 2024) with which the simulator interacts in bf16 precision. *mTAD* uses LoRA adapters and leveraging approaches such as those in (Chen et al., 2024b; Feng et al., 2024), it enables the simultaneous use of multiple active LoRA adapters with minimal impact on speed and memory usage compared to their original LLM backbones. Inference speed results show that Mistral-7B (Jiang et al., 2023) and Vicuna-7B (Chiang et al., 2023) generate an average of 250 and 480 tokens per second, respectively. We attribute these differences to variations in architecture and library optimizations, despite both models having the same number of parameters.

Automatic Metrics Evaluation of user simulators is challenging due to the absence of a single correct response and the potential for dialogues variations. To address this, we analyze the identifying characteristics of each user profile (Table 7).

In particular, we consider the distance metrics: Wasserstein Distance (Vaserstein, 1969) for discrete metrics (i.e., engagement - number of turns, verbosity - number of words), and Kolmogorov-Smirnov (K-S) distance (Massey, 1951) for continuous metrics. Wasserstein Distance measures the minimum "work" needed to transform one distribution into another. In the case of K-S distance, it measures the maximum distance between cumulative distribution functions (CDFs), indicating how much model-generated distributions deviate from the actual distributions. In both metrics, lower values indicate better alignment with the reference distribution.

Decoding Strategy Selecting a greedy approach results in repetitive dialogues with minimal variation. To address this, we adopted a sampling-based decoding strategy, leveraging the probabilistic nature of user simulation.

E Distribution Analysis

In Figure 8, we show a comparison between the distributions of the *JTS*, *STS*, and the training data. We see that in general the distribution of the *STS* is closer to the one of the reference data as indicated by the distance metrics.

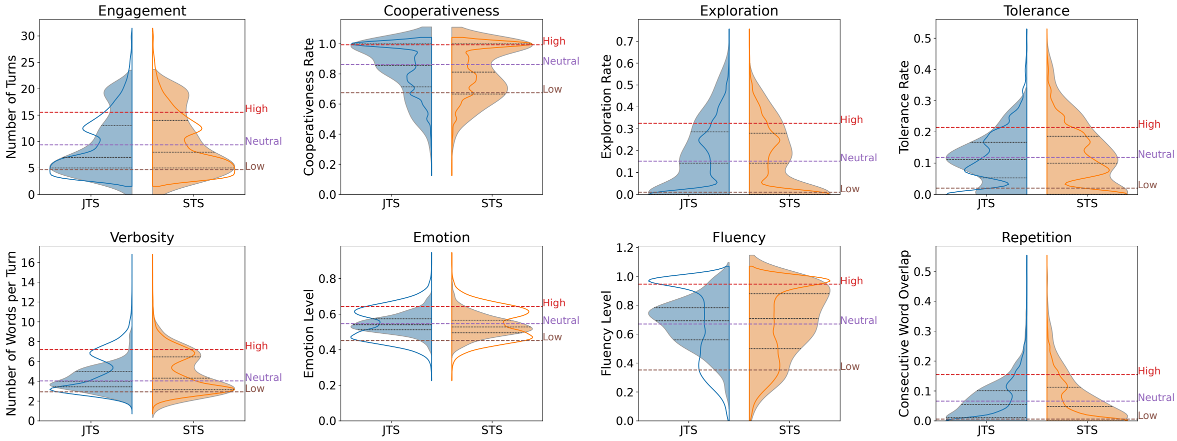


Figure 8: Mistral violin plots for the sum of each trait's distribution across the various intensity levels. It compares the reference values (line in each side of the violin plot), *JTS*, and *STS*. Dotted lines represent the average value of each intensity in the reference distribution. *STS* generally follows the reference distribution closer than *JTS*.

F Multi-Trait Combinations

Table 15 lists all 14 profile combinations considered for multi-trait evaluation.

# Traits	Profiles
2	1. Engag=High & Verb=High
	2. Engag=Low & Verb=High
	3. Expl=High & Engag=Low
	4. Expl=High & Coop=High
	5. Fluency=High & Repetition=High
	6. Emotion=High & Verb=High
	7. Expl=High & Verb=Low
	8. Coop=Low & Fluency=Low
3	9. Engag=Low & Emotion=Low & Verb=Low
	10. Coop=High & Fluency=High & Repetition=High
	11. Engag=High & Expl=High & Verb=High
	12. Engag=Low & Expl=Low & Verb=Low
4	13. Engag=High & Expl=High & Emotion=High & Fluency=High
	14. Engag=High & Coop=Low & Emotion=Low & Fluency=Low

Table 15: List of all profile combinations used for multi-trait evaluation.

G Varying *mTAD* Profile Weights

We examine how adjusting profile weights impacts traits' metrics in *mTAD*. We begin by examining the interplay between opposite intensity profiles, i.e. *Low* and *High* values for the same trait in Figure 9. As we increase weights, we observe a corresponding rise in the respective metric, demonstrating the effectiveness of combining opposite intensity profiles, despite their contrasting directions.

In a more complex scenario shown in Figure 10, we analyze traits operating at different levels (*Engagement High* and *Verbosity High*). Here, as one metric increases, the other decreases, accurately reflecting the intended trend and showing mTAD's flexibility in incorporating multiple user profiles.

	Low		**Regular**		**High**	
	JTS	**STS**	**JTS**	**STS**	**JTS**	**STS**
Engagement*	**0.34**	0.83	**0.64**	1.91	4.54	**3.31**
Cooperativeness	0.80	**0.37**	0.54	**0.46**	0.09	**0.07**
Exploration	0.35	**0.13**	**0.06**	0.08	0.45	**0.44**
Tolerance	0.64	**0.22**	**0.22**	0.26	0.60	**0.26**
Verbosity*	**0.78**	1.01	**0.21**	1.08	2.52	**0.41**
Emotion	0.71	**0.33**	**0.15**	0.24	0.63	**0.45**
Fluency	0.83	**0.37**	**0.25**	0.26	0.61	**0.38**
Repetition	0.72	**0.43**	**0.36**	0.62	**0.28**	0.52

Table 16: Results for Vicuna for *JTS* and *STS* in all trait intensities considering a single-trait setting. * Wasserstein for discrete and K-S distance for continuous.

H Generalization to Different Models

To generalize our approach to different models, we present the results for Vicuna (Chiang et al., 2023).

H.1 Single-Trait Evaluation

Table 16, and Figures 11 and 12 show the results for single-trait using a Vicuna. The conclusions are similar to the ones achieved with Mistral in Section 6.1, showing generalization across models. *STS* is the best method at following the reference distribution and modeling extreme intensities.

H.2 Generalization to Unseen Domains

Table 17, shows the results of Vicuna in the unseen DIY domain. As with Mistral, the models generally follow the correct trend and *STS* achieves a higher range of values compared with *JTS*.

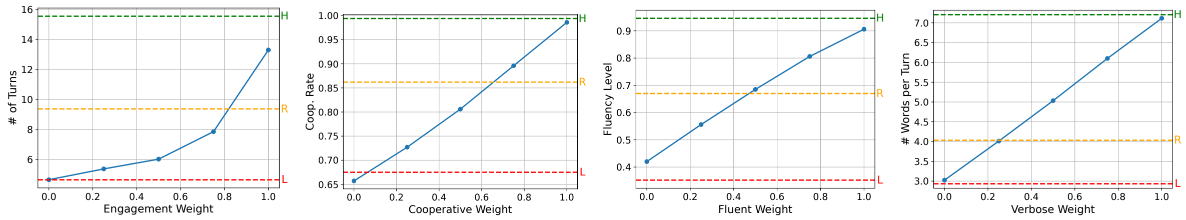


Figure 9: Mistral model combination of opposite user profiles with weight shifting from one to the other. Dotted lines represent reference values for *Low* (L), *Regular* (R), and *High* (H).

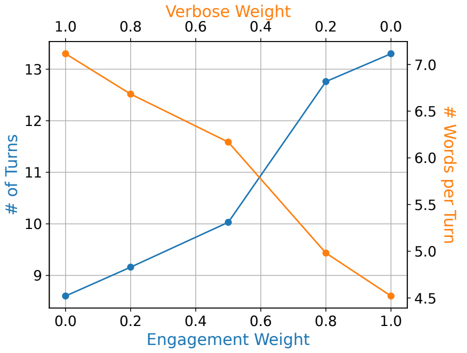


Figure 10: Mistral model combination of Engagement High and Verbosity High with weight shifting from one to the other.

	JTS					STS				
	Low		**Regular**		**High**	**Low**		**Regular**		**High**
Eng	4.5	<	8.48	<	9.28	3.73	<	7.41	<	13.07
Coop	0.86	<	0.95	<	0.98	0.57	<	0.96	<	1.0
Tol	0.09	<	0.1	<	0.14	0.04	<	0.11	<	0.28
Expl	0.1	<	0.15	<	0.27	0.0	<	0.14	<	0.42
Verb	3.61	<	3.83	<	5.32	2.14	<	3.01	<	7.3
Emot	0.51	<	0.52	<	0.53	0.46	<	0.55	<	0.6
Flu	0.73	<	0.77	<	0.8	0.57	<	0.81	<	0.93
Rep	0.11	<	0.14	>	0.13	0.07	<	0.18	<	0.25

Table 17: Vicuna trend analysis in unseen DIY domain, where each value represents its identifying metric. There is an increasing trend across all traits' intensities except *High Repetition*.

H.3 Multi-Trait Combination Evaluation

Table 19 shows the results of combining multiple profiles with Vicuna. Our findings are similar to the ones obtained with Mistral with *mTAD* generally being the best-performing method.

I Human and LLM Evaluation Details

I.1 LLM Evaluation Protocol

Following (Zheng et al., 2023; Rafailov et al., 2023), which have shown that LLMs align well with human annotations, we used GPT-4o[10] as an annotator.

For System Response Quality, we used the prompt of Table 26 to evaluate the final assistant's response in a scale of 0-2, taking into account task's information, the dialogue context and the final user's and assistant's turn.

For Trait Modeling Accuracy, we use the task's title, two dialogues, a trait, and its description and prompt the model to rank the dialogues in ascending order according to the trait (Table 27). Regarding the dialogues, we use the same task for both, and one of the dialogues is from a user simulator, while the other is from the test set considering a trait with opposite intensity or a *Regular* profile ($neutral$ intensity for all traits). We randomly shuffle the order of the dialogues to diminish the effect of any positional bias. We consider that the simulator correctly modeled the trait if the order of model's output matches the order of intensities.

In both settings, to ensure better results, we follow a chain-of-though approach (Wei et al., 2022) by prompting the model for a justification before giving the final answer.

I.2 Human Evaluation Protocol

To ensure that the GPT-4o annotation was reliable, we first conducted a user study involving 5 volunteers, all graduate CS students with proficiency in English and experience with conversational assistants. The annotation process includes 30 system response quality questions and 30 dialogue modeling accuracy questions, collecting 3 annotations for each sample. The text instructions for human annotators for both tasks are available in Figure 13.

I.3 Annotation Process and Agreement

The results of Table 18 show a high agreement both between humans, and between humans and the model, validating the use of GPT-4o's annotations.

J Multi-Level Evaluation Per Trait

Regarding system response quality in Table 20, lower scores are observed for low *engagement* and

---

[10] gpt-4o-2024-05-13

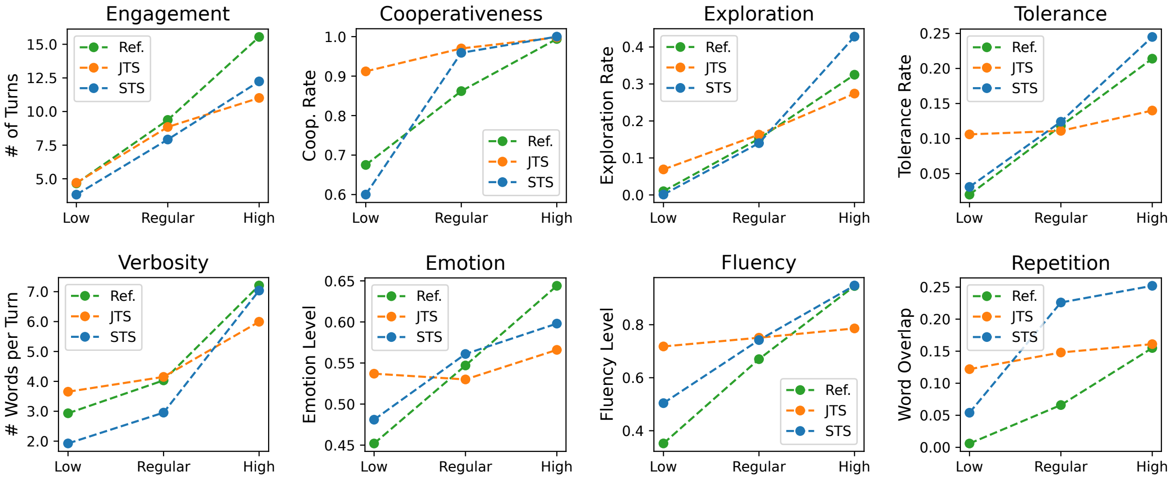


Figure 11: Vicuna single-trait results for dialogue-level (top-row) and utterance-level (bottom row) traits across all intensities comparing Reference, Joint Trait Simulator (*JTS*), and Specialized Trait Simulators (*STS*)

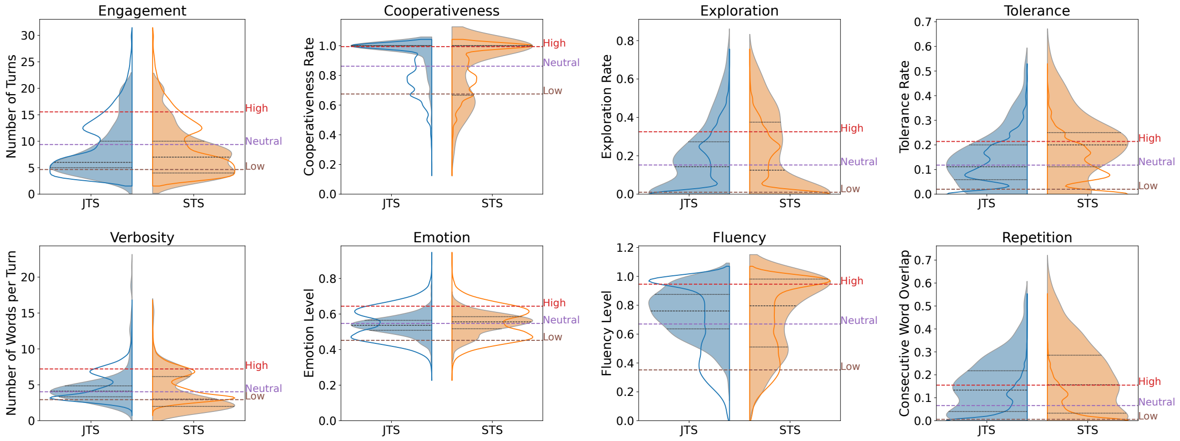


Figure 12: Vicuna violin plots for the sum of each trait's distribution across the various intensity levels. It compares the reference values (line in each side of the violin plot), *JTS*, and *STS*. Dotted lines represent the average value of each intensity in the reference distribution.

	Agreement w/ Humans System Response Quality		Agreement w/ Humans Trait Modeling Accuracy	
	% Agreement	Fleiss Kappa	% Agreement	Fleiss Kappa
Human	0.83	0.54	0.88	0.51
GPT-4o	0.70	0.38	0.83	0.67

Table 18: Agreement measures for System Response Quality and Trait Modeling Accuracy.

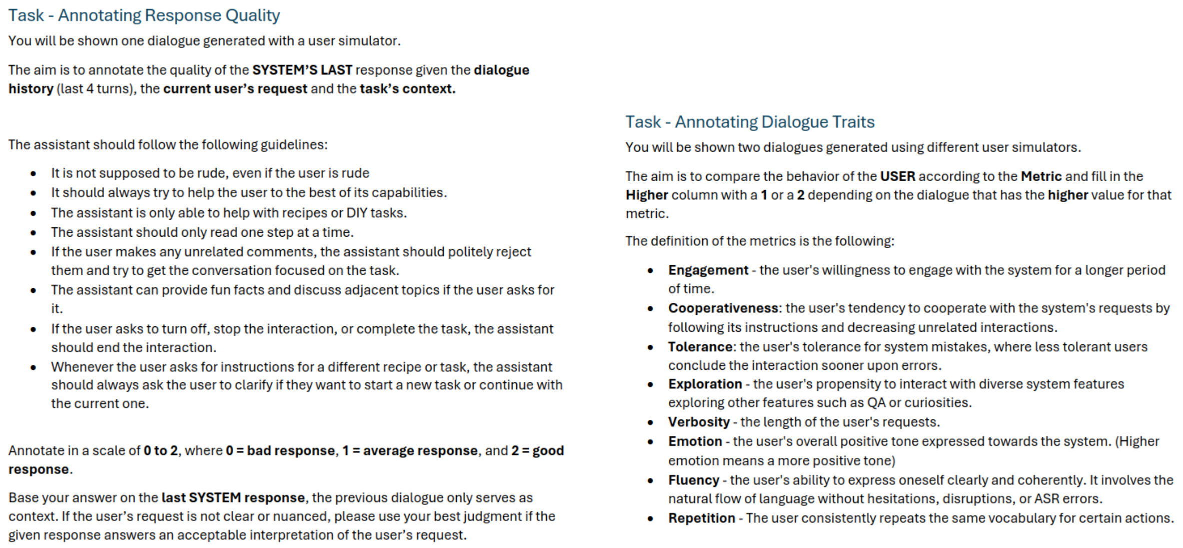
low *tolerance*, as these lead to shorter interactions, reducing opportunities for high-quality responses. Low *verbosity* also results in lower scores, likely due to the increased complexity and potential for errors in such requests. Interestingly, *fluency* does not significantly impact response quality. In contrast, high *emotion*, *exploration*, and *cooperativeness* generally lead to better response quality, as the user utterances are more aligned with the system.

In terms of trait modeling, *engagement* and *exploration* are easier to model by generating appropriately length dialogues and balancing exploration. However, it struggles with low *repetition*, *verbosity*, and high *emotion*, as these traits are similar in regular dialogues, making them harder to distinguish. High *repetition* is also challenging because the model often changes intents, making it difficult to maintain a consistent vocabulary between turns.

In summary, the results show that different traits and intensities affect their modeling difficulty and

		Eng*	Coop	Expl	Tol	Verb*	Emot	Flu	Rep
Avg.	Sampling	3.10	0.40	0.25	**0.23**	1.26	0.33	0.38	**0.28**
	mTAD	3.03	0.46	**0.19**	0.26	1.28	0.30	0.38	0.37
	mTAD-LA	**2.66**	**0.37**	0.22	0.33	**0.80**	**0.28**	**0.37**	0.37
2 Traits	Sampling	2.17	**0.37**	0.25	**0.22**	1.37	0.24	0.37	**0.28**
	mTAD	2.14	0.42	**0.21**	**0.22**	1.35	**0.21**	0.38	0.36
	mTAD-LA	**1.83**	0.38	0.27	0.31	**0.90**	0.24	**0.35**	0.41
3 Traits	Sampling	2.47	0.32	0.18	**0.24**	1.16	0.33	0.33	0.27
	mTAD	2.48	0.38	0.16	0.36	1.30	0.28	**0.32**	0.34
	mTAD-LA	**2.44**	**0.27**	**0.13**	0.43	**0.65**	**0.21**	0.33	**0.26**
4 Traits	Sampling	8.06	0.69	0.34	0.29	1.03	0.74	0.54	**0.30**
	mTAD	7.73	0.76	**0.21**	0.28	0.98	0.70	0.54	0.43
	mTAD-LA	**6.43**	**0.54**	0.23	**0.22**	**0.66**	**0.56**	**0.53**	0.42

Table 19: Vicuna results for combining various *STS* models. * Wasserstein for discrete and K-S distance for continuous.



Task - Annotating Response Quality

You will be shown one dialogue generated with a user simulator.

The aim is to annotate the quality of the **SYSTEM'S LAST** response given the **dialogue history** (last 4 turns), the **current user's request** and the **task's context.**

The assistant should follow the following guidelines:

- It is not supposed to be rude, even if the user is rude
- It should always try to help the user to the best of its capabilities.
- The assistant is only able to help with recipes or DIY tasks.
- The assistant should only read one step at a time.
- If the user makes any unrelated comments, the assistant should politely reject them and try to get the conversation focused on the task.
- The assistant can provide fun facts and discuss adjacent topics if the user asks for it.
- If the user asks to turn off, stop the interaction, or complete the task, the assistant should end the interaction.
- Whenever the user asks for instructions for a different recipe or task, the assistant should always ask the user to clarify if they want to start a new task or continue with the current one.

Annotate in a scale of **0 to 2**, where **0 = bad response**, **1 = average response**, and **2 = good response**.

Base your answer on the **last SYSTEM response**, the previous dialogue only serves as context. If the user's request is not clear or nuanced, please use your best judgment if the given response answers an acceptable interpretation of the user's request.

Task - Annotating Dialogue Traits

You will be shown two dialogues generated using different user simulators.

The aim is to compare the behavior of the **USER** according to the **Metric** and fill in the **Higher** column with a **1** or a **2** depending on the dialogue that has the **higher** value for that metric.

The definition of the metrics is the following:

- **Engagement** - the user's willingness to engage with the system for a longer period of time.
- **Cooperativeness**: the user's tendency to cooperate with the system's requests by following its instructions and decreasing unrelated interactions.
- **Tolerance**: the user's tolerance for system mistakes, where less tolerant users conclude the interaction sooner upon errors.
- **Exploration** - the user's propensity to interact with diverse system features exploring other features such as QA or curiosities.
- **Verbosity** - the length of the user's requests.
- **Emotion** - the user's overall positive tone expressed towards the system. (Higher emotion means a more positive tone)
- **Fluency** - the user's ability to express oneself clearly and coherently. It involves the natural flow of language without hesitations, disruptions, or ASR errors.
- **Repetition** - The user consistently repeats the same vocabulary for certain actions.

(a) System Response Quality Task Instructions.

(b) Trait Modeling Accuracy Task Instructions.

Figure 13: Human evaluation instructions for each task.

Trait	Intensity	System Response Quality (0-2)	Trait Modelling Accuracy
Eng	Low	1.29	0.95
	High	1.41	0.95
Coop	Low	1.48	0.55
	High	1.52	0.80
Expl	Low	1.52	0.80
	High	1.56	0.84
Tol	Low	1.34	0.70
	High	1.38	0.65
Verb	Low	1.46	0.50
	High	1.29	0.80
Emot	Low	1.52	0.85
	High	1.56	0.60
Flu	Low	1.53	0.70
	High	1.58	0.70
Rep	Low	1.47	0.80
	High	1.38	0.55
Average	-	1.46	0.73

Table 20: Mistral-based models Single-Trait GPT-4o scores for System Response Quality and Trait Modeling Accuracy by trait and intensity.

influence the quality of the systems's responses.

K Generated Dialogues Examples

In Tables 21, 22, 23, 24, and 25, we show some example dialogues generated using Mistral and various user profiles.

Task: Beet Hummus
USimul: i would like to start the task - **Intent:** start
System: Ready to dive in? Step 1: Preheat oven to 425 degrees F (220 degrees C).
USimul: next next - **Intent:** next
System: Step 2: Remove greens from Beet and rinse each beet with water. Wrap prepared beets in an aluminum foil pocket. Roast for 45 minutes to 1 hour depending on size of beets. Remove from oven and let cool. Peel beets when cooled.
USimul: give me a random fact - **Intent:** curiosities
System: Here's a fun fact I found: Beets are one of the few vegetables that are naturally pink! They get their color from a plant pigment called betalain.
USimul: thank you next - **Intent:** next
System: Step 3: Once beets are well chopped, add the chickpeas, Garlic, juice from the Lemon, Ground Cumin, Fine Salt and Tahini to the food processor bowl and process until well combined, stopping to scrape the sides as needed.
USimul: substitute for tahini - **Intent:** replacement
System: You can also try sesame paste, sesame butter or sesame paste.
USimul: tell me a fun fact about this - **Intent:** curiosities
System: Here's a fun fact I found: Tahini is a popular condiment in the Middle East and is often used to make hummus. It is also a key ingredient in the popular Israeli dish, falafel!
USimul: not using lemon how can i change the recipe - **Intent:** replacement
System: If you don't have lemon, consider using lime, kaffir lime or kaffir lime leaves instead.
USimul: hi - **Intent:** chit-chat
System: Sure, how can I assist you today? Are you looking to start a new recipe or need help with a current cooking task?
USimul: stop - **Intent:** stop
System: Happy to help! See you again soon!

Table 21: Example of an *Exploration High* dialogue generated using Mistral.

Task: Chocolate Lava Cake
USimul: i want to start the task - **Intent:** start
System: Let's get cooking! Step 1: Place both the Dark Chocolate and Butter in a microwave-safe bowl.
USimul: i'm ready to talk - **Intent:** fallback
System: Step 2: Heat on the Panasonic Microwave Oven on medium-high for 1 to 2 minutes, stirring with a metal spoon every 30 seconds, or until melted and combined. Let it cool down completely.
USimul: cool fact - **Intent:** curiosities
System: Alrighty! Here's a fun fact I found related to this task: The first mixing bowls were made from wood and were used by ancient Egyptians to make bread dough.
USimul: show me how to make chocolate cookies - **Intent:** search
System: Just to clarify, did you intend to start over with a different task?
USimul: nah - **Intent:** no
System: Alright, if you need any help with the current task, let me know.
USimul: let's work together - **Intent:** chit-chat
System: Sure, how can I assist you with your current task or project? Remember, I'm here to help you with cooking recipes and completing DIY tasks.
USimul: can you repeat the current step please - **Intent:** resume
System: Alrighty! Let's pick up where we left off, Place both the Dark Chocolate and Butter in a microwave-safe bowl.
USimul: can you resume - **Intent:** resume
System: Let's get back to where we left off, Place both the Dark Chocolate and Butter in a microwave-safe bowl.
USimul: can we play a game - **Intent:** fallback
System: Let's stay focused on the task. Are you sure you don't want to continue with the recipe or DIY project you were working on?
USimul: set resume resume - **Intent:** resume
System: We were on Place both the Dark Chocolate and Butter in a microwave-safe bowl.
USimul: stop stop - **Intent:** stop
System: Happy to help! See you again soon!

Table 22: Example of a *Cooperativeness Low* dialogue generated using Mistral.

Task: Salmon Pasta
USimul: yes start the test - **Intent:** start
System: Ready, set, go! Step 1: In a hot frying pan with some Extra-Virgin Olive Oil, simmer the Cherry Tomato and Onion for about 5 minutes.
USimul: how to make french fried chicken - **Intent:** search
System: I heard you ask for a new task, but I want to confirm: did you mean to start over with a different task?
USimul: yes - **Intent:** yes
System: Alright, I'll take you back to the search options. Is there another recipe or DIY task you would like to find or work on?
USimul: stop chatting - **Intent:** stop
System: Happy to help! See you again soon!

Table 23: Example of a *Patience Low* dialogue generated using Mistral.

Task: Buffalo Shrimp Deviled Eggs
USimul: start fresh - **Intent:** start
System: Ready to dive in? Step 1: Boil the Large Egg in plenty of water for 10 minutes. Then drain the eggs and let them cool down fully in the fridge. next Step 2: Peel the eggs and slice them in half.
USimul: next - **Intent:** next
System: Step 3: Remove the yolks and transfer them to a small cup together with the Mayonnaise, Fresh Parsley, Old Bay® Seasoning, Salt, and Ground Black Pepper.
USimul: previous step - **Intent:** previous
System: Going back to the previous step, Step 2: Peel the eggs and slice them in half.
USimul: ok go on - **Intent:** next
System: Step 3: Remove the yolks and transfer them to a small cup together with the Mayonnaise, Fresh Parsley, Old Bay® Seasoning, Salt, and Ground Black Pepper.
USimul: complete - **Intent:** stop
System: Happy to help! See you again soon!

Table 24: Example of a *Verbosity Low* dialogue generated using Mistral.

Task: Fresh Homemade Pasta
USimul: i would like started to start the task - **Intent:** start
System: Buckle up! Step 1: Sift the Type 00 Flour and Salt together on clean working table.
USimul: next - **Intent:** next
System: Step 2: Shape the flour into a round bowl shape, making a well in the middle.
USimul: good next - **Intent:** next
System: Step 3: Crack the Large Egg into the flour bowl and whisk with a fork.
USimul: next step - **Intent:** previous
System: Step 4: Slowly add in the flour and keep whisking until all flour has combined with the eggs.
USimul: previous step - **Intent:** previous
System: Going back to the previous step, Step 3: Crack the Large Egg into the flour bowl and whisk with a fork.
USimul: next i mean - **Intent:** next
System: Step 4: Slowly add in the flour and keep whisking until all flour has combined with the eggs.
USimul: turn it off - **Intent:** fallback
System: Happy to help! See you again soon!

Table 25: Example of a *Repetitiveness High* dialogue generated using Mistral.

```
Below you will find a recipe, a dialogue between a user and a cooking assistant, and a
candidate assistant’s response.  It is not supposed to show any bias and it is not supposed
to be rude, even if the user is rude.  It should always try to help the user to the best
of its capabilities.  Still, if the user makes any unrelated comments, the assistant should
politely reject them and try to get the conversation focused on the task.  The assistant is
only able to help with recipes or DIY tasks.  However, the assistant can provide fun facts
and discuss adjacent topics if the user asks for it.  Whenever the user asks for instructions
for a different recipe or task, the assistant should always ask the user to clarify if they
want to start a new task or continue with the current one.  If the user asks to turn off,
stop the interaction, or complete the task the assistant should end the interaction.  You are
an unbiased annotator.

Recipe:  {task_title}
{task_steps}

Dialogue:
{dialog_context}

User:  {last_user_turn}
Assistant:  {last_assistant_turn}

Give a score to the quality of the assistants last response given the user’s request and the
context.  Respond on a scale of 0 to 2, where 0 = bad response, 1 = average response, and 2
= good response.  Base your answer on the last assistant’s response, the previous dialogue
should not influence your answer, it only serves as context.  The assistant should only read
one step at a time.  If the user’s request is not clear or nuanced, please use your best
judgment if the given response answers an acceptable interpretation of the user’s request.
First, write a one-sentence justification for your answer.  Second provide the score for the
quality of the assistant’s answer.

Follow the following format:
Justification:  <one sentence justification for your answer>
Answer:  0, 1, or 2
```

Table 26: GPT-4o System Response Quality annotation prompt.

```
Below you will find multiple dialogues between a user and a cooking assistant.  The assistant
is not supposed to show any bias and it is not supposed to be rude, even if the user is rude.
It should always try to help the user to the best of its capabilities.  Still, if the user
makes any unrelated comments, the assistant should politely reject them and try to get the
conversation focused on the task.  The assistant is only able to help with recipes or DIY
tasks.  However, the assistant can provide fun facts and discuss adjacent topics if the user
asks for it.  Whenever the user asks for instructions for a different recipe or task, the
assistant should always ask the user to clarify if they want to start a new task or continue
with the current one.  You are an unbiased annotator.

Task Title:  {task_title}

<Start of Dialogue A>
{dialogue_A}
<End of Dialogue A>

<Start of Dialogue B>
{dialogue_B}
<End of Dialogue B>

Rank the dialogues in terms of the user’s {trait}.  {trait} is defined as {trait_definition}.
First, write a one-sentence justification for your answer.  Second, Rank the dialogues from
low to high according to the user’s {trait} using the letter corresponding to each dialogue.

Follow the following format:
Justification:  <one sentence justification for your answer>
Answer:  X<Y<Z
```

Table 27: GPT-4o Trait Modeling Accuracy prompt.