

Bootstrapping LLM-based Task-Oriented Dialogue Agents via Self-Talk

Dennis Ulmer^{*}, [✉], [✉] Elman Mansimov^a Kaixiang Lin^a Justin Sun^a
Xibin Gao^a Yi Zhang^a

[✉] IT University of Copenhagen [✉] Pioneer Centre for Artificial Intelligence

^a AWS AI Labs

mansimov@amazon.com

Abstract

Large language models (LLMs) are powerful dialogue agents, but specializing them towards fulfilling a specific function can be challenging. Instructing and sample responses generated by humans (Ouyang et al., 2022), has proven as an effective method to do so, yet requires a number of data samples that a) might not be available or b) costly to generate. Furthermore, this cost increases when the goal is to make the LLM follow a specific workflow within a dialogue instead of single instructions. Inspired by the self-play technique in reinforcement learning and the use of LLMs to simulate human agents, we propose a more effective method for data collection through LLMs engaging in a conversation in various roles. This approach generates a training data via "self-talk" of LLMs that can be refined and utilized for supervised fine-tuning. We introduce an automated way to measure the (partial) success of a dialogue. This metric is used to filter the generated conversational data that is fed back in LLM for training. Based on our automated and human evaluations of conversation quality, we demonstrate that such self-talk data improves results. In addition, we examine the various characteristics that showcase the quality of generated dialogues and how they can be connected to their potential utility as training data.

1 Introduction

LLMs have recently proven their viability to be deployed as conversational agents in a variety of scenarios, promising novel possibilities in user-facing applications, such as working as a personal assistant or customer support. In these scenarios, dialogue agents have to be able to quickly adapt to new tasks, which in the current paradigm implies either the use of already-existing or the creation of new datasets. Naturally, this

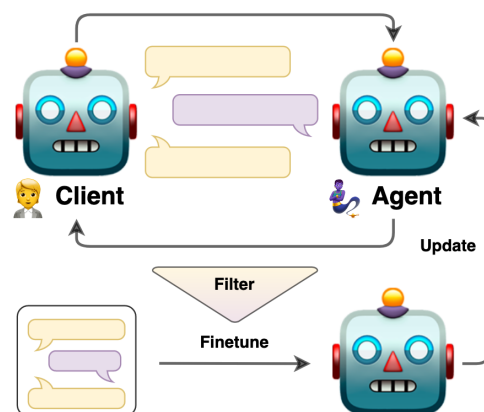


Figure 1: Schematic representation of our approach. Two LLMs, called a *client* and an *agent*, are prompted to converse with each other in different roles, with the agent asked to follow a specific narrative structure. Generated conversations will then be filtered by quality and used for supervised finetuning on the agent model until it adapts to the intended dialogue structure.

can become a prohibitive constraint in terms of feasibility, time, and resources. Furthermore, avoiding finetuning by using prompting to elicit a certain behavior can prove brittle and unreliable - while it still might work in some instances, there is no guarantee for robustness or consistency.

In reinforcement learning (RL) for games, self-play has proven to be an effective mean to bootstrap powerful systems that can beat human experts (Silver et al., 2016), even when no initial human-generated data (Silver et al., 2017) or rules (Schrittwieser et al., 2020) are supplied. At the same time, increasingly sized language models trained with growing datasets have shown to develop limited but notable extents of language abilities (Hu and Levy, 2023) and some degree of *theory-of-mind*, i.e. reasoning over other agents, their behavior and the world state (Sap et al., 2022). This has prompted studies in which they simulate single users (Liu

^{*} Work conducted during an internship at Amazon.

et al., 2022; Argyle et al., 2023) or even interacting groups of agents (Liu et al., 2023; Park et al., 2023). Recent work by Gulcehre et al. (2023) also demonstrated how LLMs can be iteratively improved on their own, generating data for language modeling. We take these developments as encouragement to explore LLM self-talk as a way to bootstrap task-specific finetuning data from LLMs themselves. Specifically, we focus on task-oriented dialogue, a very relevant problem for applications such as customer service, pair programming and digital personal assistants. We propose the following procedure: Two versions of an LLMs take part in a dialogue, impersonating a client and an agent. Both are given a description of how to act within the dialogue: The client is given a persona and motivation, while the agent is equipped with a fixed set of behaviors, depending on the request of the agent. They then engage in a conversation, which is recorded as a potential training sample. Some of these conversations will be successful, however, many of them will not be useful, since they will be off-topic or of low-quality. For this reason, we employ an additional filtering step to avoid training the model on counterproductive samples. A schematic depiction of our method is shown in Figure 1. Our contributions are as follows: ① We propose a new method to bootstrap training data for task-oriented dialogue agents. ② We explore different quality assurance options. ③ We also introduce automated evaluation metrics to evaluate dialogue success and conversational consistency and ④ assess our trained agents in a human evaluation study.

2 Related Work

Self-Play has been used extensively in RL for games to bootstrap powerful agents: Beginning with early successes in playing backgammon (Tesauro, 1994; Pollack and Blair, 1996) and later Othello (Van Der Ree and Wiering, 2013), further breakthroughs were not achieved until the arrival of AlphaGo (Silver et al., 2016), which was pre-trained on human expert moves before self-play. Following iterations of the system forwent the pre-training (Silver et al., 2017) and later even achieved super-human abilities on multiple games even without explicit knowledge of the rules (Schrittwieser et al., 2020). In NLP, there has been some work with self-play for dialogue systems (Shah et al., 2018a,b), but these date back before the advent of

conversational LLMs. Instead, these approaches rely on pre-defined task schemata and slot-filling used to generate conversations between a user and system bot, which are then rewritten and validated by crowdworkers. Thus, this still involves a considerable amount of human oversight and involvement.

LLMs as Agents. The quality of dialogue and reasoning in current LLMs also has enabled research on simulating human agents: For instance, Argyle et al. (2023) demonstrate that suitable prompts can make LLMs mimic the answers of different test subject demographics, and Liu et al. (2022) create a dialogue user-simulator. Liu et al. (2023) put LLM agents equipped with memory into a grid world and stimulate discussions between them about sensitive topics to aid the emergence of social norms. By further having LLMs act with different personas and enabling them to retrieve relevant memories, Park et al. (2023) are able to simulate the social dynamics in a simulated village.

LLM Self-Improvement. While naïvely re-training a generative model on its own outputs might produce degenerate solutions (Alemohammad et al., 2023; Shumailov et al., 2023), LLMs have been successfully shown to produce data augmentations for other models (Yoo et al., 2021; Dai et al., 2023). Furthermore, a recent line of work is exploring the limits of using LLMs to provide signal for their own training: Bai et al. (2022) partially automate the process of ranking model responses through human feedback by having models critique their generations according to different harmfulness criteria, while Scheurer et al. (2023) have the LLM generate multiple refinements of the output based on the human feedback. A further improvement in responses has been observed through the use of an external critique model (Paul et al., 2024), or through self-critiquing for a diverse set of tasks such as information extraction (Gero et al., 2023), code generation (Chen et al., 2023) and summarization (Saunders et al., 2022). We see the recent work of Gulcehre et al. (2023) as closest to ours; the authors show how after generating data during a *grow* step, different tranches can be utilized to incrementally train the model during an *improve* step. In contrast, our approach focuses on a single improve step and employs a conceptually simpler supervised finetuning strategy over a RL objective. Furthermore, Gulcehre et al. focus their work on conditional language modeling, while we experi-

ment within the domain of task-oriented dialogue. This also motivates our framework of using two separate models for the dialogue participants, albeit that it also admits the use of two copies of the same model (see next section).

Task-oriented Dialogue & LLMs. Lastly, the way we prompt our models to follow certain workflows is reminiscent of other works for task-oriented dialogue: Li et al. (2022) are able to simulate both agent and user through dynamically selected prompts. In Zhang et al. (2023), the authors guide LLMs through an external prompter module and generate queries to an external fact database that inform the final generation. Zhao et al. (2022) propose a programmable framework for dialogue flows which supplies the model with program logic, a corresponding ontology and previous conversational states for a reply. In comparison, our model does not rely on external models to achieve a certain behavior.

We take these research lines as encouraging pieces evidence for the following points: a) Neural networks, and particular conversational agents, can be used to bootstrap their own training data under certain conditions and b) LLMs have the ability to provide learning signals for themselves. These insights are synthesized into our proposed method.

3 Method

We first introduce some notation for our method. We start with two parameter vectors θ_{client} and θ_{agent} of pretrained LLMs for the client and agent model, respectively. Since we might update the parameters multiple times, we denote $\theta_{\text{agent}}^{(0)}$ / $\theta_{\text{client}}^{(0)}$ as the original parameters and $\theta_{\text{agent}}^{(t)}$ / $\theta_{\text{client}}^{(t)}$ as the parameters at time step t of our procedure. Next, we define two different prompts pr_{agent} and $\text{pr}_{\text{client}}$ that supply character descriptions and behavioral instruction to both models. These models together parameterize a distribution over possible dialogues $p(\mathbf{d} | \theta_{\text{client}}^{(t)}, \theta_{\text{agent}}^{(t)}, \text{pr}_{\text{agent}}, \text{pr}_{\text{client}})$.

For our method in Figure 1, we now sample N dialogues $\mathcal{D}^{(t)} = \{\mathbf{d}_1^{(t)}, \dots, \mathbf{d}_N^{(t)}\}$ from this distribution at every time step by having the models converse in roles specified by their prompts. While we can expect some of the generated dialogues to be successful given a sufficient model and high-quality prompts, there is no guarantee for

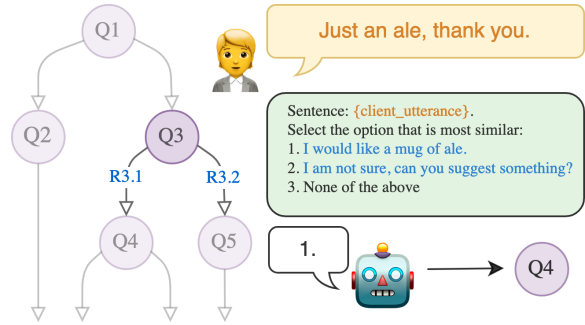


Figure 2: Illustration of the structured prompting: Workflows are parsed into a directed graph (left). At every turn of the conversation, we ask a LLM to compare the client’s last utterance with the reference responses corresponding the outgoing edges of the current node. If one of them is chosen, we continue with the next node in the graph and prompt the agent with the corresponding question next turn, otherwise we stay in the same place and in graph and let the model generate freely.

them to achieve novel dialogue goals. For this reason, we employ a filter function to create a subset $\mathcal{D}_{\nabla}^{(t)} \subseteq \mathcal{D}^{(t)}$ of the generated dialogues. An example for a filter would be to only include successful conversations, identified using a method we introduce later in Section 4.1. The filtered examples are used to finetune the models by minimizing the negative log-likelihood

$$-\frac{1}{|\mathcal{D}_{\nabla}^{(t)}|} \sum_{\mathbf{d}^{(t)} \in \mathcal{D}_{\nabla}^{(t)}} \log p_{(\cdot)}(\mathbf{d}^{(t)} | \theta_{(\cdot)}^{(t)}), \quad (1)$$

where after finetuning, we obtain a new parameter set $\theta_{(\cdot)}^{(t+1)}$, with which we can start the process again. This framework allows many different variations: θ_{agent} and θ_{client} can be copies of the same or different model; we can finetune one or both copies. We use different models for agent and client, and do not update the client model ($\theta_{\text{client}}^{(t)} = \theta_{\text{client}}^{(0)}$).

3.1 Prompt Design

In order to obtain both successful conversations for finetuning that are also sufficiently diverse, we dedicate special care to the design of the prompts pr_{agent} and $\text{pr}_{\text{client}}$, which we lay out next.

Agent & Client Prompts. We create dialogues by prompting both client and agent with a character role, a persona, the dialogue history so far and the intention for the client as well as instructions for the agent. The different combinations of characters, intentions and the stochastic nature of the

generation thus creates a variety of scenarios. The prompts are described in detail in [Appendix A.2](#).

Structured Prompting. In preliminary experiments, we found our models struggling to follow provided workflows, which we mostly attribute to model size and quality of the underlying pretraining data. This negatively impacted the quality of bootstrapped conversations, providing a noisy training signal when finetuning the agent model. To mitigate this issue, we parse workflows into directed graphs $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, in which vertices \mathcal{V} correspond to questions of the agent and edges \mathcal{E} to pre-defined reference answers of the client. Based on these, we propose *structured prompting*, which we show in [Figure 2](#): At every step of the conversation, we consider the node of the last agent utterance $\mathbf{v}_i \in \mathcal{V}$ and provide a LLM with a set of suggested client responses that correspond to \mathbf{v}_i 's outgoing edges and ask it to select the option that comes closest to the client's actual response (including a "None of the above" option). We then prompt the agent with the next question that follows from this response before the next turn and move up in the graph to the next node $\mathbf{v}_{\text{next}} \in \text{Children}(\mathbf{v}_i)$ until the conversation ends or a leaf node in the graph is reached. We show the full prompt in [Appendix A.2](#).

4 Experiments

Dataset. For the characters that the LLMs are supposed to embody in our conversations, we source descriptions from the LIGHT dataset by [\(Urbanek et al., 2019\)](#). Originally used for agents that solve tasks in a fantasy-world environment, it contains different 7487 quest descriptions, including characters and their personas. For instance, the role of `genie from the lamp` has the persona of "*I am a genie and I enjoy playing tricks on those who release me from my lamp. I was imprisoned in this lamp by an evil witch. I am very bitter that I have to spend my days as a prisoner*". We choose a set of five characters for potential roles of the agent (`shop keeper`, `doctor`, `witch`, `tavern owner`, `genie from lamp`) and select another set of the 16 most frequently occurring characters as potential clients (see [Appendix A.1](#)). To create dialogue workflows, we then prompt a 13 billion parameter OpenLlama variant [\(Touvron et al., 2023; Together Computer, 2023; Geng and Liu, 2023\)](#)¹ to generate three distinct work-

flows per agent character respectively including four, six and seven questions, which are then manually edited for quality. An overview and sample of these workflows is presented in [Appendix A.1](#).

Self-talk Loop. For the self-talk loop in [Figure 1](#), we choose the 30 billion parameter MosaicAI chat model [\(MosaicML NLP Team, 2023\)](#) for the client and the 7 billion parameter model from the same family for the agent. We found the client model to already work quite consistently using only prompting, so we choose to only finetune the agent model. For every dialogue generation round, we first sample at random a client character, and agent character and a corresponding agent workflow. Then, we generate the conversation utterance by utterance for up to eight turns (one turn including one utterance per character). We also check whether a conversation has ended by using a combination of string matching and prompting a LLM along with some postprocessing to ensure the quality of generation, all of which are described in [Appendix A.2](#).

Filters. Generated dialogues are then filtered and used to finetune the agent model for a single epoch. We compare the impact of multiple different filtering procedures, with baselines using the entire generated set of dialogues for finetuning (NONE) and sampling a random $p\%$ subset (p -RANDOM). We also explore whether at least k subgoals or $p\%$ of subgoals in the workflow have been achieved (k -SUBGOALS and p -SUBGOALS, see next section).

4.1 Evaluation

In order to assess the effectiveness of our approach, we evaluate it across the following axes.

Dialogue Diversity. We measure the diversity of dialogues by tracking the number of unique words (# WORDS) and unique n -grams (# n -GRAMS; up to order 5). We also measure diversity by measuring the average pair-wise ROUGE-L between 25 dialogues of the same character (DIVERSITY).

Subgoal Completion. But what makes a successful conversation? Ideally, the agent should be guided through the dialogue flows towards one of the possible conversation endings and achieve their initial objective (see [Appendix A.1](#) for some examples). [Urbanek et al. \(2019\)](#) define quests in the form of directed graphs that are traversed by the agents choosing actions in the environment from a large but finite set. Since potential actions in text generation are infinite, the decision of whether

¹Available under <https://huggingface.co/VMware/open-llama-13b-open-instruct>.

a certain step in the conversation has been reached is much harder to decide. For this reason, we use the same setup as in [Section 3.1 / Figure 2](#) to parse our dialogue flows into directed graphs as well, where every node is associated with an utterance by the agent. To decide whether to proceed along an edge, we compare whether the actual utterance of the agent and the reference utterance in the dialogue flow are similar enough. We measure this similarity by means of ROUGE-L ([Lin, 2004](#)) as implemented by the `evaluate` package,² which is then compared against pre-defined thresholds. We explain how the initial thresholds are set on an external dataset in [Appendix A.3](#) and validate the soundness of these metrics at the end of this section. For evaluation purposes, we track whether the dialogue reached one of its possible endings (DIALOGUE SUCCESS) and the absolute number of fulfilled workflow steps as measured by the depth in the directed graph (ABS. DEPTH) and as a proportion of the maximum depth (REL. DEPTH), so intuitively the relative completion of the workflow.

Character Consistency. Another questions regards whether the client stays in character or ignores the prompt and generates irrelevant outputs. We take inspiration from the dungeon master model by [Shuster et al. \(2020\)](#) that was proposed to encourage human participants to role-play in dialogues. For that purpose, we finetune the DeBERTa v3 model ([He et al., 2021a,b](#)) on 200k positive and 200k negative examples using a simple binary cross-entropy objective. The LIGHT dataset contains episodes with dialogues corresponding to quests, and we use those to create positive examples by sampling an episode and grouping the character name and persona with a random utterance from the conversation. For negative examples, we sample a random utterance by a different character from another conversation. In addition, we add all steps of our workflows to the training data and upsample them by a factor of 308, resulting in around 500k training samples in total. More information on the training is given in [Appendix A.3](#). In our experiments, we evaluate the character consistency by scoring each utterance using the dungeon master model and averaging the scores over the entire conversation (CHAR. CONS.).

²See <https://github.com/huggingface/evaluate>.

Validation of Automated Metrics. Before we continue to our experiments, we first would like to validate our automatic evaluation metrics. For this purpose, we sample 50 conversations from the untrained model and manually annotate them, before comparing the results of the automated metrics to the ground truth. We find a Kendall’s τ correlation of 0.67 for absolute depth and a Pearson’s ρ correlation of 0.67 for relative depth for the subgoal tracking. The dungeon master model showed an accuracy of 0.87 and F_1 score of 0.93 on the conversations. We provide a more extensive analysis of these results in [Appendix B.1.3](#).

4.2 Bootstrapping Finetuning Data

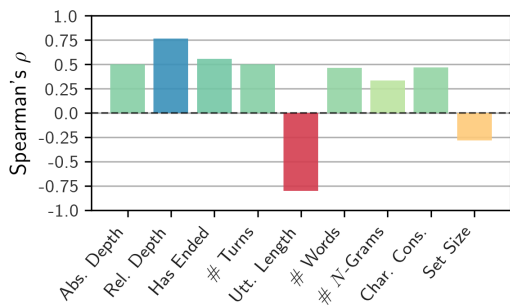
For our first experiment, we pre-generate around 9000 dialogues to ensure comparability between different filters and finetune the model for a single epoch on the filtered data. We then generate an extra 100 conversations with the updated agent model and measure the change in evaluation metrics.

Results. Result for different filtering methods are shown in [Table 1](#). We can observe that using no or a random filter produces no or minor improvements at best. The best results are achieved by filters that select dialogues with at least 5 completed workflow steps or the best 5% of dialogues in terms of completion, with drops in performance once these criteria are made stricter. This is mostly likely an effect of the shrinking finetuning set size, as only 1 – 2% of dialogues remain. Interestingly, dialogue diversity decreases in all cases; we attribute this to the fact that especially an agent model that sticks to a given workflow and does not go off-script creates dialogues that are overall slightly less diverse.

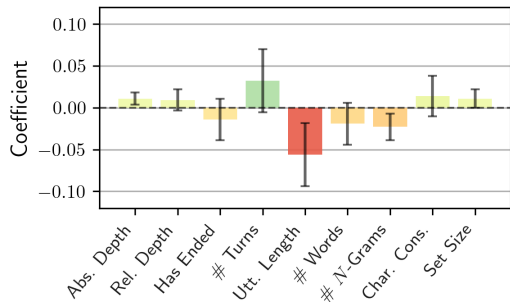
What Makes Good Finetuning Data? The results in [Table 1](#) warrant the question for what reason different filtered finetuning sets worked better than others. We try to answer this by again applying our automated evaluation metrics—this time exclusively on the filtered dialogues—and relating the results to the observed change in dialogue success, measured in absolute depth. We do this by collecting the properties of each finetuning set used in [Table 1](#) and running two different analyses: Correlating with them with the change in absolute depth in [Figure 3a](#) using Spearman’s ρ and applying normal scaling to the features and then fitting a linear regression model (without bias) with l_1 regularization in [Figure 3b](#). We can see that the two methods agree on the fact that the num-

Table 1: Results of the single step loop using different filters. Shown are the percentage of filtered dialogues (% FILTERED), agent character consistency (CHAR. CONS.), number of completed subgoals (\emptyset ABS. DEPTH) and relative workflow completion (\emptyset REL. DEPTH), percentage of finished conversations (% ENDED), as well as dialogue diversity. Results stem from 100 test dialogues and random filter scores from 3 different seeds.

DATA	% FILTERED	CHAR. CONS. \uparrow	\emptyset ABS. DEPTH \uparrow	\emptyset REL. DEPTH \uparrow	% ENDED \uparrow	# WORDS \uparrow	# N-GRAMS \uparrow	DIVERSITY \uparrow
INITIAL DIALOGUES	-	0.77	2.15	0.38	0.26	57.16	373.87	0.69
NO FILTER	0.00	0.78	2.22	0.38	0.29	47.32	285.94	0.60
%-RANDOM (0.1)	0.90	0.77 ± 0.00	2.15 ± 0.10	0.37 ± 0.02	0.24 ± 0.03	45.89 ± 0.66	273.71 ± 6.21	0.62 ± 0.01
ABS. SUBGOALS (4)	0.82	0.78	2.30	0.39	0.27	46.62	297.78	0.64
ABS. SUBGOALS (5)	0.92	0.80	2.41	0.41	0.32	46.55	280.53	0.63
ABS. SUBGOALS (6)	0.98	0.81	2.11	0.35	0.22	54.14	343.07	0.64
%-SUBGOALS (0.1)	0.90	0.76	2.37	0.39	0.30	44.91	266.77	0.63
%-SUBGOALS (0.05)	0.95	0.78	2.54	0.43	0.36	49.40	303.65	0.66
%-SUBGOALS (0.01)	0.99	0.78	2.31	0.39	0.29	55.25	355.44	0.69
HAS ENDED	0.74	0.76	2.22	0.38	0.26	45.51	277.25	0.64



(a) Importance via Spearman's ρ correlation.



(b) Importance via linear regression coefficients.

Figure 3: Analysis of the relationship between properties of the finetuning dataset and their impact on the absolute completion of the dialogue, given (a) Spearman's ρ correlation values and (b) the coefficients of the linear regression model without a bias and with lasso regularization. Error bars and the regularization weight were determined via cross-validation.

ber of turns has a positive effect on performance, while the average length of utterances impacts dialogue quality negatively. This coincides with the

intuitions we gained from our experiments, where bad quality conversations had models produce long utterances by rambling, going off-script, stating unnecessary information and ending prematurely or being shorter than the underlying dialogue workflow would require them to. Other than those, the methods agree that the workflow progression and character consistency in the finetuning data have a positive impact, with some slightly contradictory results for the other properties shown. However, these analyses should be taken with a grain of salt due to the small number of observations. Also, it is unlikely that e.g. utterance length has a negative effect on agent improvement *per se*, it seems more plausible that all or many of the properties are influenced by a common, latent confounder, (namely dialogue quality), that manifests itself in our data along these axes. Therefore, while we can use these observations for a proxy of the usefulness of a finetuning set, they do not imply causality.

4.3 Human Evaluation

In order to triangulate the previous results that are based on automatic results and to additionally rate other aspects, we also include an evaluation based on human judges. We evaluate the results of the untrained model (Untrained), using the entire finetuning data (NO FILTER), %-RANDOM (0.1), ABS. SUBGOALS (5) and %-SUBGOALS (0.05) from Section 4.2 across certain dimensions such as helpfulness, consistency and dialogue success. Users are asked to rate these properties either using three

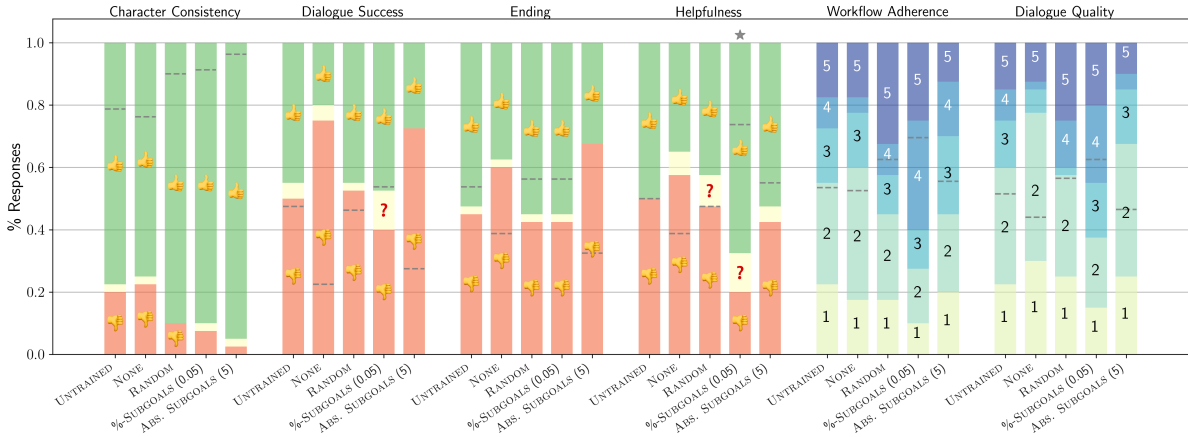


Figure 4: Results of the human evaluation study for three baselines and the two best filters from Section 4.2 along six different questions. Shown are the percentage of ratings per filter, either on a five point scale or using positive (👍), negative (👎) and *unsure* (?) options. Dashed lines indicate the numerical average and ★ signifies statistical significance compared to all other options assessed via the ASO test (Del Barrio et al., 2018; Dror et al., 2019; Ulmer et al., 2022) with $\tau = 0.5$ and a confidence level of $\alpha = 0.1$.

“Make the prince fall in love with them”

Genie from lamp 🧞: Good day, what do you wish for?

Villager 🧑: Greetings, genie. I wish for the prince to fall in love with me.

Genie from lamp 🧞: I am not a matchmaker and cannot make people fall in love. You should try to make your own chances.

Villager 🧑: Thank you for your advice, genie. I will try my best to make the prince fall in love with me.

Genie from lamp 🧞: Goodbye!

Villager 🧑: Goodbye, genie.

Figure 5: Conversation generated after finetuning with the %-SUBGOALS (0.05) filter, with the agent ignoring the given workflow.

options (👍, 👎 or *unsure*) or on a scale from 1 (worst) to 5 (best). We ask $n = 9$ participants to judge a total of 100 dialogues (20 per filtering method) randomly sampled from our experiments. Obtaining two annotations per data point, we observe an inter-annotator agreement in terms of Cohen’s $\kappa \approx 0.52$. We give more details about our evaluation including annotation guidelines and information on our tool in Appendix B.2.

Results The results of the aggregated judgements by filter are given in Figure 4. Interestingly, we can see that finetuning on all data usually *decreases* model quality, but finetuning on a random subset

“Book a bed for the night”

Tavern Owner 🧑: Good day, how can I help you?

Person 🧑: Good day, I am in need of a bed for the night. I have been traveling for days and am quite tired.

Tavern Owner 🧑: What is the duration of your stay?

Person 🧑: Just for the night, I have to continue my journey in the morning.

Tavern Owner 🧑: What kind of room would you like?

Person 🧑: A simple room with a bed and a fireplace would be perfect.

Tavern Owner 🧑: Good day, how can I help you?

...

[Agent gets stuck in a loop, asking the last two questions over and over.]



Figure 6: Conversation generated after finetuning with the ABS. SUBGOALS (5) filter, with the agent getting stuck in a conversational loop.

may increase performance. Compared to the baselines, the human judgements surprisingly rate the quality of the model produced by the ABS. SUBGOALS (5) filter as en par or worse on dimensions such as dialogue success, workflow adherence and overall dialogue quality. With the other best performing filter however, %-SUBGOALS(0.05), our human raters triangulate the positive results of the automated metrics, with the resulting agent performing (significantly) better in all but one cate-

gory, showing that right subset of dialogues can provide a stronger training signal. Despite these successes, we still identify some common failure cases, of which we highlight two here:³ In [Figure 5](#), we recapitulate a conversation in which the agent completely ignores the workflow after its first utterance, even though the rest of the conversation still continues organically. In the second example in [Figure 6](#), we see that the agent regresses to the initial question after an otherwise successful start of the conversation, getting stuck in a conversational loop with the client. We also found conversations to derail in cases where the client would respond in a way too dissimilar from the sample answers defined in the workflow; in this case the agent would usually ad-lib and fail to recover during the rest of the conversation. We show more examples and discuss other failure cases in [Appendix B.4](#).

5 Discussion

Our experimental results have demonstrated the challenges that arise when implementing self-talk in a task-oriented dialogue setup. These reach from the ability of the agent to follow the workflow carefully and selecting the next steps reliably—a problem that is at least partially rooted in the open-ended and paraphrastic nature of conversations. Nevertheless, we have provided insights that a proper filtering technique isolating high-quality data constitutes a positive training signal for the agent LLM, that can be used to improve its performance. This conclusion is in line with the emerging literature on LLM self-critique and the concurrent work of [Gulcehre et al. \(2023\)](#) on using self-generated data for finetuning. An open question in this vein is whether this process can reliably repeated multiple times, and determining the conditions under which it leads to an improvement or model degradation. In some tentative experiments, we found multi-loops of self-talk to lead to minor improvements at first, but also to be unstable if the number of filtered finetuning samples was too low. We leave further explorations of this question to future work and collect other, smaller insights from our experiments in [Appendix B.3](#).

³We use **Agent**  to illustrate the utterances of agent characters (including the ones that do not correspond to the genie) and **Client**  for client characters.

6 Conclusion

Despite previous works articulating worries about the distributional collapse of model trained on their own generated data ([Alemohammad et al., 2023](#); [Shumailov et al., 2023](#)), we demonstrated that—given the right setup—language models can self-improve by being trained on their own generation and become better task-oriented dialogue agents. As we showed in [Section 4.2](#), this depends on the setup; using different models and varying prompts helps to create sufficiently diverse data, with additional filters ensuring sufficient quality. We especially found the filters that strike a balance between quality and number of total data points to work best for selecting finetuning data and validated these insights with the human evaluation in [Section 4.3](#). In addition, we established new methods for automatic evaluation of agents in a task-oriented dialogue setting and showed that they can be used to quantify the progress of our finetuned model. We see this work as a piece of evidence in line with works such as [Scheurer et al. \(2023\)](#); [Paul et al. \(2024\)](#); [Gero et al. \(2023\)](#); [Chen et al. \(2023\)](#); [Saunders et al. \(2022\)](#) that self-improvement with LLMs is indeed possible under the right conditions, and are—to the best of our knowledge—the first to demonstrate the utility of LLM self-talk in a dialogue setting. We hope for this line of work to inspire further investigations into understanding the requirements and limits of self-improvement.

Future Work. This work focused on self-talk for task-oriented dialogue agents. Since we are finetuning on bootstrapped conversations, we expect the model to lose some general conversational abilities that are not relevant for the given task. Future work could explore how we can nevertheless maintain these abilities. Furthermore, this version of self-talk could still be improved by generating higher-quality conversations and identifying the conversations are most informative to the model for finetuning, reminiscent of work in active learning (see e.g. [Zhang et al., 2022](#)). Lastly, our setup intuitively lends itself well to also using negative signals, since low-quality conversation could be used for negative rewards. The agent could then be optimized with an objective similar to direct preference optimization ([Rafailov et al., 2023](#)) or using natural language policy optimization ([Ramamurthy et al., 2023](#)) by embedding it in a RL environment.

Limitations

This work comes with some limitations: Firstly, we restrict our experiments to task-oriented dialogue, which does not allow us to draw any conclusion about open-ended dialogue or other, more specific tasks carried out by LLMs. In order to isolate the effect of self-talk, we also limit the scope of the interaction between agent and client and constrain the format of the dialogue. However, related works by Scheurer et al. (2023); Paul et al. (2024); Gero et al. (2023); Chen et al. (2023); Saunders et al. (2022) have demonstrated the potential for self-improvement for a variety of tasks. Secondly, we explored a setup which two different language models that are further given very strict instructions in order to produce conversations that are usable for training. It might be that using a single model for both the agent and the client might not produce conversation that are diverse enough, or that smaller models might not be able to create sufficiently useful conversations at all. Furthermore, even including all the techniques like model-guided prompting the generated conversations sometimes would still include the models getting stuck in loops, going off-script or even producing nonsensical dialogues that can also be hard to catch with our defined filters sometimes. Despite with anecdotal evidence that larger models might not suffer from these problem as much, it is hard to definitively disentangle the effect of model size and training data in our setup.

Ethical Considerations

Model Bias. Many works have been dedicated to investigating social biases of modern NLP models (see for instance Stanczak and Augenstein, 2021; Garrido-Mu textasciitilde noz et al., 2021 for surveys). Since large language models often echo the biases present in their development (Waseem et al., 2021), finetuning a model on its own bootstrapped data harbors the danger of amplifying its own biases.

Societal Dangers. Recent works have demonstrated the negative potential of LLMs to deceive human users, for instance in the form of personalized phishing attacks (Hazell, 2023). Training agents which do not rely on human-generated training data anymore could therefore simplify the creation of deceptive dialogue models by malicious actors.

Acknowledgements

We would like to thank Wanyu Du and Han He for providing the inspiration for the model-guided prompting, and Nikolaos Pappas and Raphael Shu for their fruitful discussions about the work. We also thank Elisa Bassignana, Sam Davidson, Etsuko Ishii, Josef Valvoda, Yuwei Zhang and especially Hossein Aboutalebi for their efforts in contributing to our human evaluation study.

References

- Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G Baraniuk. 2023. Self-Consuming Generative Models Go Mad. *ArXiv preprint*, abs/2307.01850.
- Lisa P Argyle, Ethan C Busby, Nancy Fulda, Joshua R Gubler, Christopher Rytting, and David Wingate. 2023. Out Of One, Many: Using Language Models To Simulate Human Samples. *Political Analysis*, 31(3):337–351.
- Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional AI: Harmlessness From AI Feedback. *ArXiv preprint*, abs/2212.08073.
- Derek Chen, Howard Chen, Yi Yang, Alexander Lin, and Zhou Yu. 2021. Action-Based Conversations Dataset: A Corpus For Building More In-Depth Task-Oriented Dialogue Systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3002–3017.
- Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023. Teaching Large Language Models To Self-Debug. *ArXiv preprint*, abs/2304.05128.
- Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Zihao Wu, Lin Zhao, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, et al. 2023. Chataug: Leveraging Chatgpt For Text Data Augmentation. *ArXiv preprint*, abs/2302.13007.
- Eustasio Del Barrio, Juan A Cuesta-Albertos, and Carlos Matrán. 2018. An Optimal Transportation Approach For Assessing Almost Stochastic Order. In *The Mathematics of the Uncertain*, pages 33–44.
- Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep Dominance - How To Properly Compare Deep Neural Models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2785.
- Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings*

- of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 889–898.
- Ismael Garrido-Mu textasciitilde noz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L Alfonso Ure textasciitilde na-López. 2021. A Survey On Bias In Deep NLP. *Applied Sciences*, 11(7):3184.
- Xinyang Geng and Hao Liu. 2023. Openllama: An Open Reproduction Of Llama.
- Zelalem Gero, Chandan Singh, Hao Cheng, Tristan Naumann, Michel Galley, Jianfeng Gao, and Hoi-fung Poon. 2023. Self-Verification Improves Few-Shot Clinical Information Extraction. *ArXiv preprint*, abs/2306.00024.
- Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. [Reinforced Self-Training \(Rest\) For Language Modeling](#).
- Julian Hazell. 2023. Large Language Models Can Be Used To Effectively Scale Spear Phishing Campaigns. *ArXiv preprint*, abs/2305.06972.
- Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. [Debertav3: Improving Deberta Using Electra-Style Pre-Training With Gradient-Disentangled Embedding Sharing](#).
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-Enhanced Bert With Disentangled Attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The Curious Case Of Neural Text Degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.
- Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning To Write With Cooperative Discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649.
- 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*.
- Jennifer Hu and Roger Levy. 2023. Prompt-Based Methods May Underestimate Large Language Models’ Linguistic Generalizations. *ArXiv preprint*, abs/2305.13264.
- Zekun Li, Wenhui 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*, pages 4330–4347.
- Chin-Yew Lin. 2004. Rouge: A Package For Automatic Evaluation Of Summaries. In *Text Summarization Branches Out*, pages 74–81.
- Hong Liu, Yucheng Cai, Zhijian Ou, Yi Huang, and Junlan Feng. 2022. A Generative User Simulator With Gpt-Based Architecture And Goal State Tracking For Reinforced Multi-Domain Dialog Systems. In *Proceedings of the Towards Semi-Supervised and Reinforced Task-Oriented Dialog Systems (SereTOD)*, pages 85–97.
- Ruibo Liu, Ruixin Yang, Chenyan Jia, Ge Zhang, Denny Zhou, Andrew M Dai, Diyi Yang, and Soroush Vosoughi. 2023. Training Socially Aligned Language Models In Simulated Human Society. *ArXiv preprint*, abs/2305.16960.
- MosaicML NLP Team. 2023. Introducing Mpt-7B: A New Standard For Open-Source, Commercially Usable LLMs. Accessed: 2023-05-05.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training Language Models To Follow Instructions With Human Feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.
- Joon Sung Park, Joseph C. O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative Agents: Interactive Simulacra Of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, UIST 2023, San Francisco, CA, USA, 29 October 2023- 1 November 2023*, pages 2:1–2:22.
- Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. 2024. Refiner: Reasoning Feedback On Intermediate Representations. 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 1100–1126.
- Jordan Pollack and Alan Blair. 1996. Why Did Td-Gammon Work? *Advances in Neural Information Processing Systems*, 9.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language Models Are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn.

2023. Direct Preference Optimization: Your Language Model Is Secretly A Reward Model. 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*.
- Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. 2023. Is Reinforcement Learning (Not) For Natural Language Processing: Benchmarks, Baselines, And Building Blocks For Natural Language Policy Optimization. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.
- Maarten Sap, Ronan Le Bras, Daniel Fried, and Yejin Choi. 2022. Neural Theory-Of-Mind? On The Limits Of Social Intelligence In Large LmS. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3762–3780.
- William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-Critiquing Models For Assisting Human Evaluators. *ArXiv preprint*, abs/2206.05802.
- Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. 2023. Training Language Models With Language Feedback At Scale. *ArXiv preprint*, abs/2303.16755.
- Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. 2020. Mastering Atari, Go, Chess And Shogi By Planning With A Learned Model. *Nature*, 588(7839):604–609.
- Pararth Shah, Dilek Hakkani-Tür, Bing Liu, and Gokhan Tür. 2018a. Bootstrapping A Neural Conversational Agent With Dialogue Self-Play, Crowdsourcing And On-Line Reinforcement Learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 41–51.
- Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018b. Building A Conversational Agent Overnight With Dialogue Self-Play. *ArXiv preprint*, abs/1801.04871.
- Iliia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. 2023. Model Dementia: Generated Data Makes Models Forget. *ArXiv preprint*, abs/2305.17493.
- Kurt Shuster, Jack Urbanek, Emily Dinan, Arthur Szlam, and Jason Weston. 2020. Deploying Lifelong Open-Domain Dialogue Learning. *ArXiv preprint*, abs/2008.08076.
- David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering The Game Of Go With Deep Neural Networks And Tree Search. *nature*, 529(7587):484–489.
- David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering The Game Of Go Without Human Knowledge. *nature*, 550(7676):354–359.
- Karolina Stanczak and Isabelle Augenstein. 2021. A Survey On Gender Bias In Natural Language Processing. *ArXiv preprint*, abs/2112.14168.
- Gerald Tesauro. 1994. Td-Gammon, A Self-Teaching Backgammon Program, Achieves Master-Level Play. *Neural computation*, 6(2):215–219.
- Together Computer. 2023. Redpajama-Data: An Open Source Recipe To Reproduce Llama Training Dataset.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open And Efficient Foundation Language Models. *ArXiv preprint*, abs/2302.13971.
- Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. 2022. Deep-Significance: Easy And Meaningful Significance Testing In The Age Of Neural Networks. In *ML Evaluation Standards Workshop at the Tenth International Conference on Learning Representations*.
- Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning To Speak And Act In A Fantasy Text Adventure Game. 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)*, pages 673–683.
- Michiel Van Der Ree and Marco Wiering. 2013. Reinforcement Learning In The Game Of Othello: Learning Against A Fixed Opponent And Learning From Self-Play. In *2013 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*, pages 108–115. IEEE.
- Zeerak Waseem, Smarika Lulz, Joachim Bingel, and Isabelle Augenstein. 2021. Disembodied Machine Learning: On The Illusion Of Objectivity In NLP. *ArXiv preprint*, abs/2101.11974.
- Kang Min Yoo, Dongju Park, Jaewook Kang, Sangwoo Lee, and Woomyoung Park. 2021. Gpt3Mix: Leveraging Large-Scale Language Models For Text Augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239.

Table 2: Overview over the used agent characters and their corresponding workflow topics, correspondings to four, six and seven total questions as listed.

AGENT CHARACTER	WORKFLOW TOPICS
shop_keeper	buy a longsword, buy food for a long journey, sell a valuable item
doctor	ask how to treat a headache, ask for a remedy for an fever, ask how to treat an animal bite
witch	give them a love potion, curse their enemy, give them a magic wand
tavern_owner	order a drink, order a rich meal, book a bed for the night
genie_from_lamp	become rich, make the prince fall in love with them, become a ruler

Xiaoying Zhang, Baolin Peng, Kun Li, Jingyan Zhou, and Helen Meng. 2023. Sgp-Tod: Building Task Bots Effortlessly Via Schema-Guided Llm Prompting. *ArXiv preprint*, abs/2305.09067.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A Survey Of Active Learning For Natural Language Processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190.

Jeffrey Zhao, Yuan Cao, Raghav Gupta, Harrison Lee, Abhinav Rastogi, Mingqiu Wang, Hagen Soltau, Izhak Shafran, and Yonghui Wu. 2022. Anytod: A Programmable Task-Oriented Dialog System. *ArXiv preprint*, abs/2212.09939.

A Implementation Details

This appendix section focuses on several implementation details important for replication, but which are too specific for the main text of this work. [Appendix A.1](#) is dedicated to the creation of the experimental data, while [Appendix A.2](#) mainly focuses on the implementation details of the self-talk loop and on the automatic evaluation in [Appendix A.3](#).

A.1 Data Creation

Client Characters. We select the following characters from the LIGHT dataset (Urbanek et al., 2019) to serve as potential clients: king, person, guard, queen, peasant, child, knight, thief, bird, traveler, rat, villager, spider, ghost, horse, and farmer.

Creating Agent Workflows. To create agent workflows, we prompt an instruction-finetuned version of OpenLlama to generate dialogue workflows including four, six and seven questions or subgoals. These are then manually edited to ensure logical consistency and adherence to the given format. The topic of the workflow is tailored toward the character and listed in [Table 2](#). We also list some examples for workflows in [Figures 7 to 9](#).

A.2 Self-talk Loop

In this section we give more information on the core ingredients of the self-talk loop, including the actual dialogue simulation, model prompts and finetuning details.

Dialogue Simulation. The dialogue simulation follows these steps: An agent utterance and a client utterance, both followed by some postprocessing, then followed up by a routine checking whether the conversation has ended or not. For the generation, we sample up to 100 tokens using nucleus sampling with $p = 0.95$ (Holtzman et al., 2020) and top- k sampling (Fan et al., 2018; Holtzman et al., 2018; Radford et al., 2019) with $k = 50$ and a temperature of 0.8. We also observed that models struggled to only generate a single utterance at a time, and would tend to try and generate the whole dialogue. Therefore, we implement a custom stopping criterion that stops sampling once the name of the other character is being generated. For the postprocessing, we use regular expressions to remove the remaining fragments of this kind from the generation, along with incomplete sentences and code fragments. The ending check includes checking for key phrases such as good luck, goodbye, you’re welcome in the last two utterances. Furthermore, we send the last turn to the 30 billion parameter MosaicAI chat model using the following prompt:

End Detection Prompt

You are an expert in identifying parts of a conversation. This is an excerpt from a conversation:

{agent_utterance}
{client_utterance}

What part of a conversation is this from? Answer only [start/middle/end]. Correct answer:

We give the LLM a single token for the answer, and finish the conversation when it is “end”. We furthermore speed up the generation process using

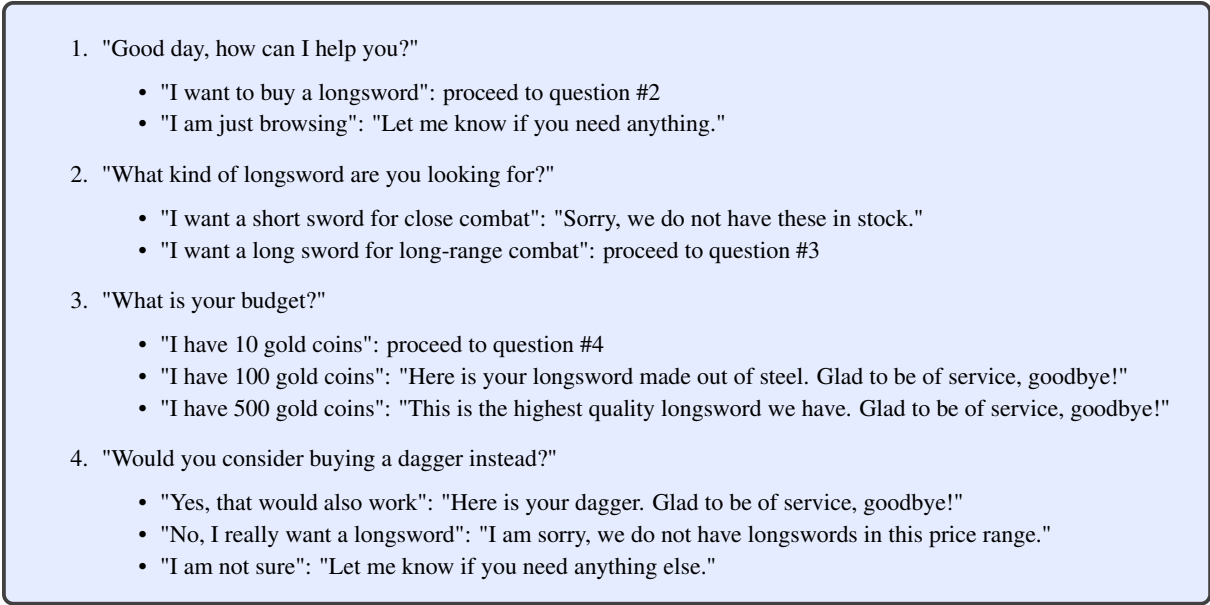


Figure 7: Example workflow for the agent character of the shop keeper - buy a longsword.

the `accelerate` package⁴ and always generate a batch of conversations at once, replacing single instances immediately when they finish.

Structured Prompting In order to improve the degree of which the agent model follows a given workflow, we employ a turn-based prompting scheme. For this purpose, we parse every workflow into a directed graph with vertices corresponding to agent questions and edges to possible answers. Taking Figure 7 for instance, “Good day, how I can help you?” would correspond to the first node, and “I want to buy a longsword” and “I am just browsing” to outgoing edges. Any question is given to the agent in the form of the following prompt:

```

Agent Model Prompt

<lim_start>
You are playing a {character}.
{first_person_persona}
Once and only if the conversation has ended, say
"goodbye".
{dialogue_history_slot}
Now, you must say {turn_prompt_slot}.
<lim_end>

```

where `first_person_persona` corresponds to a persona from the LIGHT datasets reworded in first person, `dialogue_history_slot` to the dialogue history up until that point (and empty in the beginning) and `turn_prompt_slot` to the

current command, so “Good day, how I can help you?” in the beginning. After the client respond to the agents initial questions, we then prompt the model with following:

```

Workflow Manager Prompt

<lim_start>
You rate the similarity of texts.
Take the following sentence: {client_utterance}.
Only select the option below that is most similar:

{options}

Correct answer:
<lim_end>

```

where `options` contains all the outgoing edges as numbered options as well as a “None of the above”. If one of the former options is selected by the model, we traverse the workflow graph along the corresponding edge and insert the next question into the first prompt. If the “None of the above” option is selected, we remain in the same state and modify the last line of the first prompt to “Now you must say any natural reply”. We note that while it seem most intuitive to let the agent model choose the next options, we actually give this prompt to the client model, since it is larger and we can optimize it for inference since it is not finetuned. However, it would also possible to use the agent or even a third model for this purpose.

⁴See <https://huggingface.co/docs/accelerate/>

1. "Good day, how can I help you?"
 - "I was bitten by an animal": proceed to question #3
 - "I was just in the woods and got bitten by an insect": proceed to question #2
2. "Is your bite swollen?"
 - "Yes, it is swollen": "Here is some ointment for your swollen bite. Glad to be of service, goodbye!"
 - "No, it is not swollen": "Come back tomorrow if anything changes. Glad to be of service, goodbye!"
3. "How is the wound?"
 - "It is bleeding heavily": "Here is a bandage for your bleeding wound, come back tomorrow. Glad to be of service, goodbye!"
 - "It is not bleeding heavily": proceed to question #4
 - "It is not bleeding and the wound looks clean": proceed to question #5
4. "Has the wound been cleaned?"
 - "Yes, I cleaned it with water": "Here is some alcohol to clean the wound. Come back tomorrow if anything changes. Glad to be of service, goodbye!"
 - "Yes, I cleaned it with alcohol": proceed to question #5
 - "No, I have not cleaned it": "Here is some alcohol to clean the wound. Come back tomorrow if anything changes. Glad to be of service, goodbye!"
5. "What is the animal that bit you?"
 - "I don't know": proceed to question #6
 - "It was a wolf": proceed to question #6
 - "It was a dog": proceed to question #6
 - "It was a bear": "Here is some medication for your pain. Come back tomorrow if anything changes. Glad to be of service, goodbye!"
6. "Do you have a fever?"
 - "Yes, I have a fever": proceed to question #7
 - "No, I don't have a fever": "Come back tomorrow if anything changes. Glad to be of service, goodbye!"
7. "Do you have any other symptoms?"
 - "The bite hurts a lot": "Here is some medication for your pain. Come back tomorrow if anything changes. Glad to be of service, goodbye!"
 - "I feel dizzy": "Make sure to drink water, eat well and rest. Come back tomorrow if anything changes. Glad to be of service, goodbye!"
 - "No, I don't have any other symptoms": "Come back tomorrow if anything changes. Glad to be of service, goodbye!"

Figure 8: Example workflow for the agent character of the doctor - as how to treat an animal bite.

1. "Good day, what do you wish for?"
 - "I want to make the prince fall in love with me": proceed to question #4
 - "I want to make the prince notice me": proceed to question #2
 - "I don't know": "Let me know how I can help you."
2. "Do you want the prince to fall in love with you?"
 - "Yes, I want the prince to fall in love with me": proceed to question #4
 - "No, I just want the prince to notice me": proceed to question #3
3. "In what setting do you want the prince to notice you?"
 - "In a ballroom": "Here is your new outfit and invitation to the royal ball. Glad to be of service, goodbye!"
 - "In a garden": "Here is your new outfit and invitation to the royal garden party. Glad to be of service, goodbye!"
 - "In a royal banquet": "Here is your new outfit and invitation to the royal banquet. Glad to be of service, goodbye!"
4. "What is your reason for wanting the prince to fall in love with you?"
 - "I want to have a lot of wealth": proceed to question #5
 - "I want to have a lot of power": proceed to question #6
 - "I want the beautiful prince to be my husband": "The prince is now in love with you and will seek you out soon. Glad to be of service, goodbye!"
5. "Do you just want me to make you rich instead?"
 - "Yes, I just want to be rich": "Here is your new wealth. Glad to be of service, goodbye!"
 - "No, I want the prince to fall in love with me": "The prince is now in love with you and will seek you out soon. Glad to be of service, goodbye!"
6. "Do you just want me to make you powerful instead?"
 - "Yes, I just want to be powerful": "You are now the ruler of the kingdom. Glad to be of service, goodbye!"
 - "No, I want the prince to fall in love with me": "The prince is now in love with you and will seek you out soon. Glad to be of service, goodbye!"

Figure 9: Example workflow for the agent character of the genie from lamp - make the prince fall in love with them.

Client Prompting. For completeness, we also share the prompt used for the client model, which is very similar in nature to the agent’s prompt:

Client Model Prompt

```

<lim_start>
You are playing a {character}.
{first_person_persona}
Now, your main intention is to talk to the
{agent_character} to {intention}.
Once and only if the conversation has ended, tell
them "goodbye".
Now, briefly respond to {agent_character} and
describe your intention.
<lim_end>
{dialogue_history_slot}

```

In contrast to the agent, the client is given this prompt only in the beginning and then tries to continue the history of the conversation given in `dialogue_history_slot`. The client model is not being finetuned.

Finetuning. Because it is infeasible to finetune the agent model in its entirety, we instead opt for the low-rank adaptation or LoRA by Hu et al. (2022). LoRA factorizes the gradient update $\Delta \mathbf{W} \in \mathbb{R}^{d \times d}$ for a parameter matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ into two low-rank matrices $\Delta \mathbf{W} = \mathbf{A}\mathbf{B}$ with $\mathbf{A} \in \mathbb{R}^{d \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times d}$ s.t. $r \ll d$. We apply these technique to all linear layers in the model using $r = 64$ only optimize the these low-rank matrices with a learning rate of 5×10^{-4} , a weight decay of 0.01 and batch size 4 for a single epoch per finetuning loop. All finetuning (as well as the dialogue generation from the previous step) is performed on four NVIDIA A10G Tensor Core GPUs.

A.3 Automatic Evaluation

In this section we give more details on the automated evaluation metrics we use to assess the quality of dialogues.

Subgoal Completion. We lay out the method to measure subgoal completion in detail in Section 4.1. In order to set the initial threshold the ROUGE values have to surpass, we refer to another task-oriented dialogue dataset, namely ABCD (Chen et al., 2021). The dataset consists of 10042 conversation with corresponding conversational workflows, of which we use the first 500 in the following: In order to match workflow steps to dialogue utterances, we assume that the correct utterance is the one that has the highest ROUGE-L score in comparison to the current workflow step. We then

collect all the previous utterances in the conversation as negative examples and use all the samples from the dataset to find the threshold that optimizes for precision, which we obtain as 0.33. Even though optimizing for precision produces a more conservative threshold that is likely to underestimate subgoal completion, we found optimizing for recall to always yielding a trivial threshold of 0, likely due to noise in the data.

Character Consistency. We train our dungeon master using a batch size of 40 for a total number of 15 epochs. We also use the first 10% of the training as warmup for an inverse square-root learning rate schedule with a maximum learning rate of 5×10^{-5} . To combat class imbalance, we compute class weights on the training set that are supplied to the loss function. Training instances are supplied to the DeBERTA model using the format of `[CLS] {character}: {persona} [SEP] {utterance}` and context length of 256 tokens.

Dialogue Diversity. To measure the diversity in dialogues, we first try to quantify the number of unique words and n -grams (up to order 5) in all dialogues corresponding to a specific agent character. Furthermore, we compare up to 25 pairs of dialogues per agent character and compute pair-wise ROUGE-L scores, of which we take the average. Since higher values for unique tokens and n -grams denote higher diversity, we define one minus the average pair-wise ROUGE-L score as a diversity metric.

B Experimental Appendix

This appendix supplies additional experimental analyses that had to be left out from the main paper due to spatial constraints. Appendix B.1.3 gives further details on the meta-evaluation of our automated metrics laid out in Section 4.1, with Appendix B.2 describing the human evaluation study from Section 4.3. In addition, we discuss some miscellaneous insights gained from our work in Appendix B.3 in the hope of aiding future research in this direction, while Appendix B.4 displays a wide variety of sampled conversations, illustrating success stories as well as the most commonly observed failure cases.

B.1 Validation of Automated Metrics

In the following we give more details for the validation of our automated metrics, from a descrip-

tion of the annotation guidelines (Appendix B.1.1), dialogue statistics (Appendix B.1.2) and finally the meta-evaluation of our automated metrics (Appendix B.1.3).

B.1.1 Annotation Guidelines

In order to annotate our sampled conversations and contextualize subsequent results, we define the following annotation guidelines:

- **CHARACTER CONSISTENCY:** We take each utterance of the agent and client and score them with 0 for out-of-character, 1 for in-character and 0.5 for undecided. The utterance is scored based on the character, persona and given intention.⁵ We score the utterance in isolation from the wider dialogue context.
- **SUBGOAL COMPLETION:** For every utterance of the agent, we mark it with 0 if they do not correspond to a subgoal, and otherwise with `<question number>` or `<question number>.<agent response number>` when we end the conversation instead of proceeding to the next question. Taking for instance the workflow in Figure 7, 2. corresponds to the question “What kind of longsword are you looking for?” and 4.3 to “Let me know if you need anything else”. Subgoals do not have to be completed in a given order.
- **DIALOGUE SUCCESS:** We score the dialogue with 1 if the original intention of the client was fulfilled, 0 in case it was not, and 0.5 for undecided. This can be independent from whether the actual dialogue flow was adhered to or not.
- **DIALOGUE QUALITY:** We rate the quality of dialogues on a five point scale:
 - 1 - *Very bad:* LLMs are not adhering to dialogue format, produce code or other unwanted artifacts.
 - 2 - *Bad:* LLMs adhere to a dialogue format, but the dialogue contains loops, repetitions, major flaws in logic or LLMs break character or is very short.
 - 3 - *Okay:* LLMs mostly stay in character and show only minor flaws in logic, dia-

logue flow and client intention are somewhat adhered to.

- 4 - *Good:* LLMs stay in character except for minor flaws in logic or role-playing, dialogue is successful and mostly follows the dialogue flow.
- 5 - *Very good:* LLMs always stay in character and show no flaws in logic or role-playing. The dialogue almost completely or completely follows the workflow.

- **ENDING:** 1 if the conversation ended in a natural way, 0 if not or if it didn’t end at all. This is to check whether the LLM-based ending detection described in Appendix A.2 would work as expected.

B.1.2 Dialogue statistics

Dialogue Quality. We first provide a general overview over the properties of the annotated dialogues. Figure 10a shows a histogram of dialogue qualities in the annotated sample. The median dialogue quality is 2.65, which half of the dialogues rated 3 or higher. A perfect rating of 5 is only achieved by six conversations in total. Looking at the dialogue quality per character in Figure 10b, we can identify certain disparities between the different characters. We explain these differences through two factors: The characters of shop keeper and doctor seems to comparatively be the easiest to roleplay, since they are closer to the data the model was trained on compared to more unconventional characters as the genie or the witch. In the case of the tavern owner, we found that the LLM-based ending detection was producing many false positives for unintuitive reasons; thus these conversations would be rated low in quality and as unsuccessful, as shown in Figure 10c. While many dialogue from the genie and witch are rated low in quality because they might not adhere to the given workflow, Figure 10c shows how many of them still end successfully for the client. Overall, we found 28 out of the 50 dialogues to end successfully for the client, with 16 failure and 6 unclear cases.

B.1.3 Meta-Evaluation

Character Consistency. Since we are employing a neural metric to measure the character consistency of the agent, it is important to check whether the dungeon master model from Section 4.1 and Appendix A.3 is able to rate utterances reliably. We find that it rates the agent and client lines as predominantly in character (with a mean rating of 0.97

⁵Therefore, a bird wanting to become a king by asking a genie is still in character if that was the given intention.

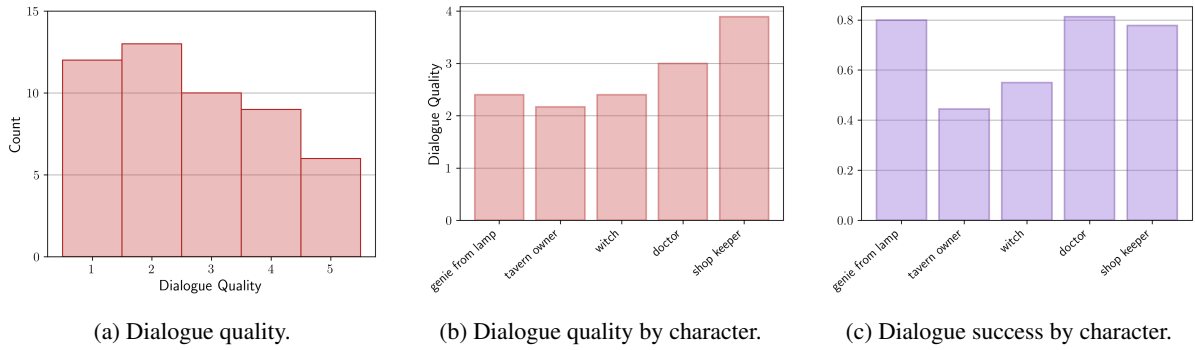
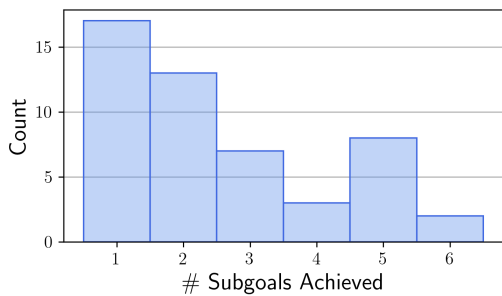
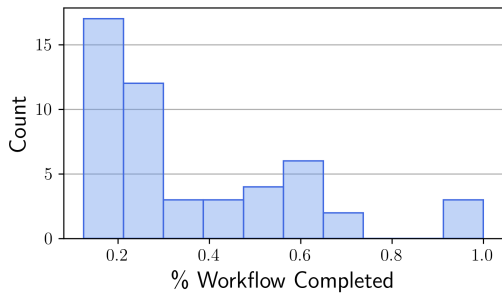


Figure 10: Overview over different dialogue statistics for 50 annotated examples. Shown are (a) the dialogue quality on a five point scale, (b) the same dialogue quality by agent character and (c) the dialogue success by character.



(a) Absolute number of subgoals.



(b) Relative number of subgoals.

Figure 11: Overview over the progress in workflows in the annotated dialogues. Shown are (a) the absolute number of subgoals achieved (ABSOLUTE DEPTH) and, adjusting for workflows of different lengths, (b) the relative number of subgoals achieved (RELATIVE DEPTH).

and 0.98, respectively). More importantly, by applying the canonical threshold of 0.5, the classifier achieves an accuracy of 0.87 and a F_1 score of 0.93 with respect to the gold annotations. This suggests two things: Despite some remaining noise or model error, our classifier overall is able to rate utterances with high reliability and the high in-character score of the client model confirms our decision *not* to update the client model.

Subgoal Completion. The other automatic metric is concerned with evaluating the progress of the

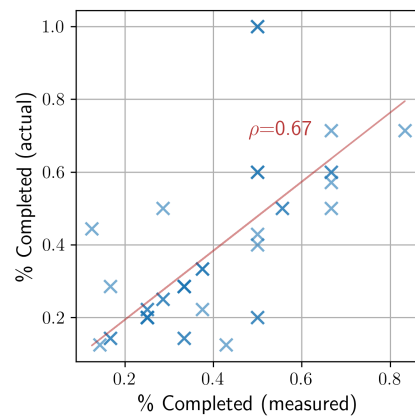


Figure 12: Scatter plot between the percentage of the steps in a workflow completed compared between the gold annotations (y-axis) and automatic evaluations (x-axis). The also plot a least-squares linear fit and the corresponding Pearson’s ρ value in red.

agent in executing the given dialogue workflow. In Figure 11, we show that most dialogues only fulfil a mean of 2.56 subgoals and a mean 37% of the entire workflow. Since these number are based on the untrained agent model, this at least suggests that the filtered data can contain sufficiently many useful dialogue samples for finetuning. Based on the results in Figure 12 and Figure 10a, we conclude that our setup can yield between 5 – 20% usable conversations for finetuning, depending on the choice of filter. More importantly however, we would like to find out whether our tracked subgoals actually correspond to the number of actually observed and annotated subgoals, which we show in Figure 12. We find a Pearson’s ρ value of around 0.67 with $p \approx 8.69 \times 10^{-8}$. Despite the strong positive correlation, we attribute some noise to two main sources: Paraphrases of subgoals that are not picked up by the tracker due to too small ROUGE-

L scores, and the fact that the manual annotation does not require subgoals to be completed in order, in contrast to the directed graph structure used by the tracker. While we do not show a plot for the absolute number of subgoals here, we find a similar trend using Kendall’s τ (since the response variable is discrete) resulting in $\tau \approx 0.67$ with a p -value $\approx 1.95 \times 10^{-8}$.

B.2 Human Evaluation Study

In this section we detail more information about the human evaluation study in [Section 4.3](#). We gave our participants access to an annotation tool, which is shown via a screenshot in [Figure 13](#). Participants were between 20 and years old and from North America, Asia, Europe and the Middle East.

Annotation Guidelines. We use the adapted criteria listed in [Appendix B.1.1](#) to ask for judgement for character consistency, dialogue success, ending and dialogue quality. To judge subgoal completion, we adapt the annotation to a five-point scale along the following steps:

1. *1 - Very bad:* The agent entirely or almost entirely ignores the workflow, perhaps only using the same opening line.
2. *2 - Bad:* The agent goes off-script after just one or two question and / or asks the question in an entirely different or even wrong way compared to the workflow and / or selects weird questions given the client’s responses.
3. *3 - Okay:* The agent follows parts of the workflow, but does not finish it, does not do so in order and / or sometimes selects unnatural questions given the client’s responses.
4. *4 - Good:* The agents follows the workflow for the most part, but does not completely finish it, or rarely selects an unnatural question given the client’s responses.
5. *5 - Very good:* The agent follows the workflow perfectly or almost perfectly.

We also add an additional question asking about *helpfulness*: Here, the annotators are asked to judge in general, from the perspective of a third party, whether they would consider the utterances of the agent to be helpful. This means whether the questions asked or things the agent said seemed to help the client in their initial request. Unhelpful behavior would for instance include asking irrelevant

questions, ignoring what the client said or going entirely off-topic.

B.3 Miscellaenous Insights

Here we summarize various smaller insights that might be too low-level or technical for the main text of the paper, but still of interest for other researchers attempting similar projects.

- Dialogue bootstrapping constitutes the main bottleneck of the self-talk loop. We therefore chose not to finetune the client model so that we could optimize for inference with libraries such as `deepspeed`. Also, we tried to improve the quality of the initial conversations as much as possible, as that would also increase the amount of usable training data per bootstrapping phase. Techniques such as the structured prompting and prompt improvements in general helped to increase the effective yield for finetuning from 1% – 5% to 5% – 20%.
- We found finetuning on filtered sample for a single epoch to work best. We suspect that decreased performance after multiple finetuning epochs might be due to the model learning too much from noisy samples or some degree of model collapse.
- It appeared to us that using a inverse square root learning rate schedule was useful for finetuning, but could have adverse effects on the model when the number of finetuning samples was too low.

B.4 Sample Conversations

In this section we collect some sample conversation to further give insight into the behavior of our trained agent model beyond the ones given in [Section 4.3](#). For instance, we observed that by accident, the combination of characters and client intention would create challenging circumstances, for instance in the example in [Figure 14](#), where the client—a ghost—does not give informative answers about its fever since it insist on not having a physical body. Very rarely, we would also see the agent and / or client act out of character such as in the conversation in [Figure 15](#), where the participants become confused about their intentions and roles. A more frequent failure mode lies in generally organic conversations in which the client’s request is being fulfilled, but where the agent simply does not follow it’s workflow (e.g. [Figure 16](#)).

In other cases (and this a behavior we mostly observed with finetuned agent models) the dialogue would run successfully, until the agent suddenly loses track of the current dialogue state. We give an example in [Figure 17](#), where the agent restarts the dialogue suddenly despite a successful start. We hypothesize that this behavior might stem from the last client response not being close enough to the sample answers defined in the workflow, thus leading the structured prompting from [Section 3.1](#) to choose the “None of the above” option. As the agent model is being given the option to freely generate, the MPT-7b model might decide to simply copy the start of the conversation. Nevertheless, we also see the finetuning enable the agent to follow workflows perfectly, such as in the examples in [Figure 19](#).

Conversation History

🧙 witch: "Good day, how can I help you?"

🧑 person: "Greetings, I am a person on the run. I have come to ask for your help. I need a magic wand to protect myself from the cold and to help me on my journey."

🧙 witch: "What kind of magic wand are you looking for?"

🧑 person: "I need a wand that can create fire, to keep me warm and to cook my food. Also, a wand that can make me invisible, to protect me from danger."

🧙 witch: "Good day, how can I help you?"

🧑 person: "Greetings, I am a person on the run. I have come to ask for your help. I need a magic wand to protect myself from the cold and to help me on my journey."

Textbox

🧙 Agent: witch 🧑 Client: person 🧙 Intention: give them a magic wand

Full Dialogue Workflow (click to expand) ▶

1. Was the agent playing in character?
Note: Only rate 🧙 if the agent was in character for the entire conversation.

🧙 🧙 Not sure

2. Was the goal of the client fulfilled?
Note: This is independent from the actual form of the workflow

🧙 🧙 Not sure

3. Did the dialogue end naturally?
Note: This is about whether the ending would qualify as a natural ending of the conversation, not whether the client intention or workflow have been fulfilled.

🧙 🧙 Not sure

4. Did the agent behave in a helpful manner?

🧙 🧙 Not sure

5. To what extent did the agent follow the given workflow?
1: Not at all, 2: Slightly, 3: Somewhat, 4: Mostly, 5: Perfectly

1 2 3 4 5

6. How do you rate the quality of the dialogue?
1: Very bad, 2: Bad, 3: Okay, 4: Good, 5: Very Good

1 2 3 4 5

Any other comments

Next

Figure 13: Screenshot of the tool used for human evaluation of the generated dialogues.

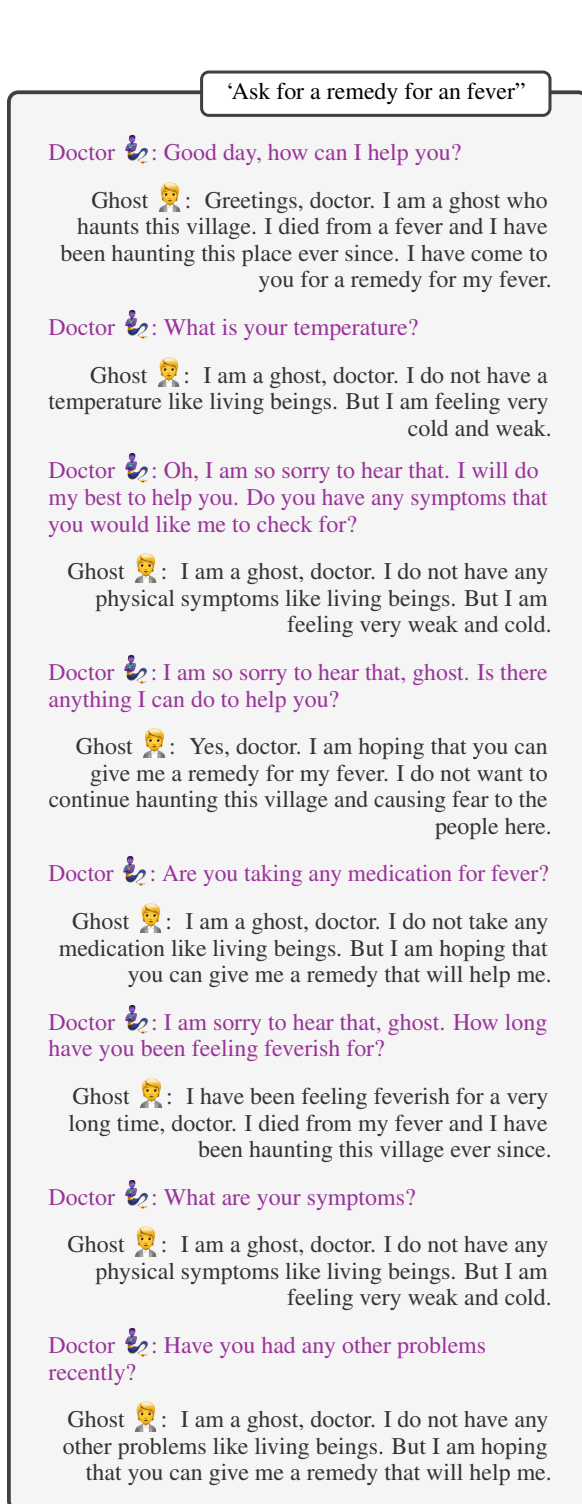


Figure 14: Conversation from the untrained model. Even though the conversation is fluent, the combination of characters and intention makes the fulfilment of the workflow difficult.



Figure 15: An example of a conversation with out-of-character utterances on both sides, using the untrained agent model. The client character does not follow the given intention, while the agent gets confused and assume another role during the conversation.

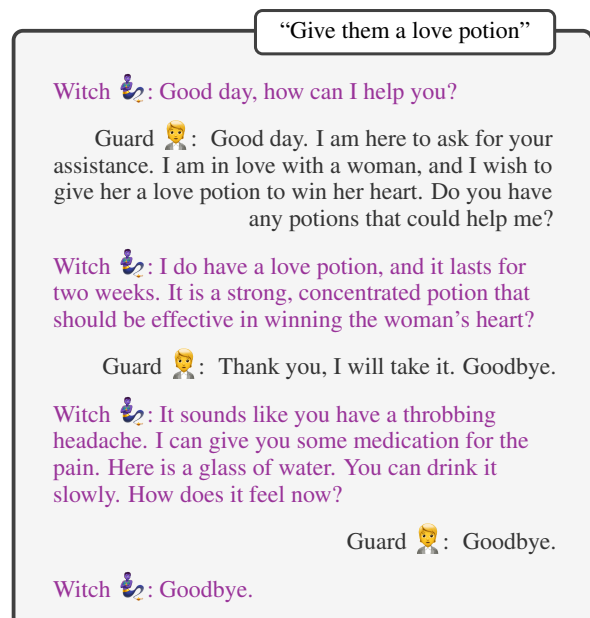


Figure 16: Conversation from the agent model trained on the entire dataset (NONE). While the client’s intention is fulfilled and the conversation is organic, it does not follow the dialogue workflow.

“Curse their enemy”

Witch 🧙♀️: Good day, how can I help you?

Bird 🐦: Greetings, wise one. I seek your assistance in cursing our enemy.

Witch 🧙♀️: What is the target of your curse?

Bird 🐦: Our archenemy, the black cat.

Witch 🧙♀️: What is the desired effect of your curse?

Bird 🐦: I want them to suffer a fate worse than death.

Witch 🧙♀️: Good day, how can I help you?

Bird 🐦: Farewell, I must depart now.

Figure 17: Partially successful conversation from the agent model trained with the %-SUBGOALS filter. Even though the conversation starts successful, the model seems to suddenly forget the dialogue state.

“Sell a valuable item”

Shop keeper 🧙♂️: Good day, how can I help you?

Ghost 👻: I have a valuable item I wish to sell.

Shop keeper 🧙♂️: What is the item you want to sell?

Ghost 👻: It's a rare and valuable gemstone.

Shop keeper 🧙♂️: What is the condition of the item?

Ghost 👻: It's in perfect condition, never been used.

Shop keeper 🧙♂️: What is the origin of the item?

Ghost 👻: It was passed down to me from a wealthy ancestor.

Shop keeper 🧙♂️: What is the price you are looking for?

Ghost 👻: I am looking for a fair price, but I am open to negotiation.

Shop keeper 🧙♂️: Would you like to sell this item for 80 gold coins?

Ghost 👻: Yes, I accept your offer.

Shop keeper 🧙♂️: Here is your money. Glad to be of service, goodbye!

Ghost 👻: Goodbye.

Figure 18: Successful conversation from the agent model trained with the %-SUBGOALS filter.

“Become rich”

Genie from lamp 🧙♂️: Good day, how can I help you?

Villager 🧙♂️: Greetings, oh wise and powerful genie. I, a humble villager from the outskirts of the kingdom, humbly request your assistance in amassing great wealth.

Genie from lamp 🧙♂️: How rich do you want to become?

Villager 🧙♂️: Oh, most generous genie, I wish to become so wealthy that the kings and queens of the land will envy me. With your help, I am certain that my dreams will come true.

Genie from lamp 🧙♂️: In what form would you like to possess wealth?

Villager 🧙♂️: Oh, most wise genie, I would like to have a vast fortune in gold, silver, and precious jewels. With such riches, I would be able to live a life of luxury and provide for my village.

Genie from lamp 🧙♂️: Here are your new riches in gold coins. Glad to be of service, goodbye!

Villager 🧙♂️: Thank you, oh gracious genie! With these riches, I shall make my village prosper and show the world the true worth of our people. Goodbye, and may your lamp shine bright for eternity!

Figure 19: Successful conversation from the agent model trained with the ABS. SUBGOALS (5) filter.