

オーダーメイド対話管理： 商用チャットボットに向けた対話管理学習データ自動作成

馬 春鵬

株式会社リクルート Megagon Labs, Tokyo, Japan

ma.chunpeng@megagon.ai

概要

ツール拡張言語モデルを用いた対話管理エージェントは流行しているが、既存ベンチマークデータは実際のビジネスタスクと乖離があり、商用チャットボットに使用できない。本研究で提案するデータ作成手法ではデータ作成者が与えた必要最小限の情報に基づいて、要望に沿った大量で高品質なデータを自動的に生成してオーダーメイド対話管理を実現した。作成されたデータは学習データとして対話管理エージェントの学習や改良に使える。

1 はじめに

自然言語処理分野において対話は重要なタスクの一つである [1]。ChatGPT¹⁾が登場してから、対話の重要性がますます認識され、大規模言語モデル (LLM) に基づいた対話の枠組みに様々なタスクが取り込まれている [2, 3, 4]。

対話の出力の正確性を高めるために、LLM に外部ツール²⁾を呼び出させて返事を生成させる取り組みは近年盛んに取り込まれている [6, 7]。このようにツール拡張言語モデルは対話管理エージェント³⁾として使われていることにより、ハルシネーションの課題が緩和され、実際のビジネス場面にも使える LLM に基づいたチャットボットが可能になった。

ツール拡張言語モデルを学習するために、多くの既存研究がベンチマークデータを作成したが、これらのデータはビジネス現場のユースケースとミスマッチであり、問題設定が単純で、人間の専門家に依存するため、商用チャットボットにおける対話管理に向いていない (2 節を参照)。

これらの課題を踏まえて、本研究は商用チャット

ボットにおける対話管理向けの学習データ自動作成手法を提案した (3 節を参照)。事前に指定されたドメインや事前定義された外部ツールの集合に依存せずに、データの作成者は自らタスクの説明文とツールの説明文を与えると、作成者の要望に沿った高品質なデータが自動的に大量生成され、オーダーメイド対話管理が実現できる (4 節を参照)。これらのデータを学習データとして使うと、対話管理エージェントの性能は向上させた (5 節を参照)。

2 背景：既存ベンチマークデータ

ツール拡張言語モデルの代表的なベンチマークデータの課題を説明する。

2.1 課題 1：ビジネス現場とのミスマッチ

既存ベンチマークデータと実際のビジネス現場に主に以下のミスマッチがある。

分野のミスマッチ。 既存データは特定の API 種類 [8, 9] や特定分野 [10, 11] に特化したものが多く、それ以外のタスクに適用できない。

タスクのミスマッチ。 実際に取り組んでいるタスクは、多分野のベンチマークデータ [12, 13, 14] に含まれたとしても、実際の商用サービスに特化されておらず、商用チャットボットの対話管理エージェントの改良に使いにくい。

現実性のミスマッチ。 実際のタスク情報を得られないまま、LLM を使って架空のデータを作成するベンチマークデータが多く存在する [15, 16]。実際のタスクを考慮したベンチマークデータもあるが [17, 18]、データ作成の段階で現場のタスク情報は得られないため、現場とのミスマッチが生じる。

2.2 課題 2：問題設定の単純化

既存ベンチマークデータは以下の点において問題設定の単純化が現れている。

1) <https://chatgpt.com/>

2) 既存研究 [5] に従い、本稿では、ツールを「LLM の外部環境に稼働しているプログラムだ」と定義する。

3) 対話管理に様々なタスクがあるが、本稿には対話管理におけるツール呼び出しに限定する。

に基づいてツールの振る舞いを模倣する。入力に対して、ツールの説明文の情報から出力を推測する。

ツールの出力を確認する LLM。 前記のツールを模倣する LLM は人間プログラマーが実装したツールではないため、出力の正確性は保証できない。結果をチェックする LLM では、出力を検査し、正常な場合のみ応答として使う。

3.3 LLM による対話品質の監査

上記のように対話が進んで、ある時点で対話の状態を判断する LLM が「終わり」と判断すると、対話を終了する。この対話は良質な対話でないと、学習データとして使えない。そのため、対話の品質を判定する LLM で生成された対話の品質を判断し、高品質な対話のみ学習データとして採用する。

4 実験結果と議論

本研究の目的を踏まえて、作成されたデータは以下の 2 つの特徴を持つことが必須である。

- 汎用性：作成者が必要最小限の情報さえ提供できれば、要望に沿ったデータを作成できる。
- 高品質：データは対話管理エージェント改良用の学習データであるため、品質が高い。

我々は GPT-4o⁴⁾ を LLM として実験を行い、提案手法によって作成されたデータが確かにこの 2 つの特徴を持っていることを検証する。

4.1 議論 1：データ作成手法の汎用性

汎用性を検証するために、我々は以下の 3 つのタスクに対してデータを作成した：(1) 宿の予約、(2) 問い合わせの自動返答、(3) 電車の乗換案内。タスクの説明文およびそれぞれのタスクに使われたツールの一覧を表 3 に記した。これらの情報はデータ作成者から提供された全ての情報である。

作成されたデータの一部を表 4 に記した。作成されたデータを観察して以下のことがわかった。

提案手法は汎用性を持っている。 3 つのタスクそれぞれの特徴がかなり異なるが、提案手法でそれぞれのタスクに対して実際にありそうな対話を作成した。データ作成者に提供された情報はかなり少ないにもかかわらず、LLM の協同作業により、データ作成者の要望が理解された。さらに、ツールの説明文からそれぞれのツールがうまく模倣されて、与え

られた入力に対して、正しい出力が得られた。

対話の流れの計画が実現できた。 ツールの連続的呼び出しが必要な場合、対話の状態を判断する LLM は対話の主導権を何回も連続的にチャットボットに渡し、最後に応答を生成する LLM が今までのツールの出力をまとめて返事を生成する。ツールを呼び出す順番は正しく、計画が実現できた。

実世界との連結が実現できた。 例えば、電車の乗換案内内のタスクに対して、タスクの説明文やツールの説明文に東京にある駅の情報が含まれていないが、作成されたデータに東京に実在する駅や路線名が正しく生成された。

4.2 議論 2：データの高品質

データの品質を保証するために、我々は品質に大きく影響する LLM に対して、プロンプトエンジニアリングを行った。宿予約タスクを対象とする比較は表 5 に記した。

顧客の発話を生成する LLM に対する工夫。 LLM が顧客の発話を生成する際に、顧客を模倣することを忘れて AI のように話すことがある。我々はプロンプトに「あなたは顧客で、リクエストを出す側です」や「サービス側の仕事をやらないください」のような厳しい制約を入れて、人間らしい発話を生成させた。

対話の状態を判断する LLM に対する工夫。 短期の計画を実現するために提案手法に同じ話者の連続的発話が許されるが、計画に関係ないときにも同じ話者が連続的に類似している内容を話すことがある。我々は「ツールの呼び出しのときのみ同じ話者を選べます」や「自然文の返事が生成されたあと対話の主導権が変わる」などの制約を入れて、計画能力を保つまま、類似している内容の重複を避けた。

対話の品質を判定する LLM に対する工夫。 既存ベンチマークデータ [14, 19] を作成する際に最終検査の責任者に出された指示に参考して、我々是对話の品質を判定する LLM のプロンプトを設計した。具体的に、我々是对話全体の合理性や自然さを考慮しつつ、外部ツールの呼び出しの正確さに特に重視している。対話全体の品質が高いかつ外部ツールの呼び出しが正しい対話のみが採用される。

5 自動作成されたデータの応用

タスクに一致する高品質なデータが自動的に大量に作成でき、データの使い道は多様である。

4) <https://openai.com/index/hello-gpt-4o/>

表 1 軽量化対話管理エージェント学習の実験結果。認識と選択タスクの評価尺度は accuracy であり、適用タスクの評価尺度は完全一致の割合である。

| | 認識 | 選択 | 適用 |
|----------|------|------|------|
| 直近 1 ターン | 0.86 | 0.89 | 0.81 |
| 直近 3 ターン | 0.93 | 0.94 | 0.87 |

5.1 軽量化対話管理エージェントの学習

LLM が流行っている今の時代でも、携帯電話など計算リソースが限られて LLM が使えない場面はまだ残っている。このような場面にチャットボットを作る場合、対話管理エージェントに従来の機械学習モデルが使われている。提案手法によって自動作成されたデータは従来の機械学習モデルの学習のために使えることを証明するために、我々は対話管理に必要な以下の 3 つのタスクに取り組んだ。

- **【認識】** 対話履歴に基づいて、次の返事を生成するために、外部ツールを呼び出すべきかどうかを予測する (2 値分類)。
- **【選択】** 外部ツールを呼び出すべく場合、どのツールを呼び出すかを予測する (k 値分類、 k は外部ツールの数である)。
- **【適用】** とある外部ツールを呼び出す場合、ツールの入力を予測する (生成タスク)。

我々は提案手法を使って宿予約タスクの対話を 750 個生成し、タスクごとにデータを整形して、無作為にそれぞれ 1,000 個のサンプルを抽出し、学習データとする。認識と選択タスクの場合、日本語 BERT[22]⁵⁾ の上に 768×2 (或いは k) のフィードフォワード層を加えて、ファインチューニングする。適用タスクの場合、日本語 BERT をファインチューニングした後、予測させたい部分を [MASK] にして予測させる。

さらに、我々は人間ワーカーに依頼し、評価用データをそれぞれ 100 個を作った。3 つのタスクの結果は表 1 に記した。

モデルは小規模言語モデルであり、作成されたデータに人間から提供した情報が必要最小限しか含まれていないにも関わらず、作成されたデータを使って、日本語 BERT は 3 つのタスクに全て高い精度を獲得した。これで提案手法で作成されたデータは学習データとして使えることがわかり、手法の有効性が証明できた。

5) <https://huggingface.co/tohoku-nlp/bert-base-japanese-v3>

表 2 既存対話管理エージェント改良の実験結果。

| | 認識 | 選択 | 適用 |
|-------------|------|------|------|
| GPT-4o | 0.40 | 0.34 | 0.55 |
| +1-shot 学習 | 0.42 | 0.38 | 0.63 |
| +5-shot 学習 | 0.45 | 0.41 | 0.70 |
| +10-shot 学習 | 0.48 | 0.49 | 0.76 |

5.2 既存対話管理エージェントの改良

軽量化モデルの学習だけではなく、普通の商用チャットボットにおける対話管理エージェントを改良するために、提案手法で自動作成されたデータを活用できる。一例として、我々は few-shot 学習の実験を行った。具体的に、GPT-4o に基づいた対話管理エージェントがあり、プロンプトに対話管理に関する指示はあるが、タスクに特化した例は含まれていないケースを想定する。そこで、我々は提案手法で自動作成されたデータから無作為に n 個のサンプルを選び、プロンプトに入れる。評価タスクと評価用データは 5.1 節と同じだ。実験結果は表 2 に記した。

既存対話管理エージェントは強力な大規模言語モデル GPT-4o を使ったが、プロンプトにタスクに特化した情報が含まれていない場合、精度は 5.1 に紹介した日本語 BERT に劣っている。プロンプトに作成された学習データを少量で加えただけで、3 つのタスクに対して性能の改善は確認できた。

5.3 データの有益性に関する議論

人間から必要最小限の情報しか提供されておらず、LLM を使って全自動で作成されたデータはなぜ有益になるのか？その理由はデータに含まれている知識に関わっている。軽量化対話管理エージェントには実世界に関する知識が不十分であるため、LLM を使って作成されたデータで実世界の知識が補完された。LLM に基づいた既存対話管理エージェントの場合、実世界の知識は含まれているが、人間が設計した LLM の協同作業に関する知識やデータの品質検査の知識などが含まれていないため、作成された学習データはやはり役に立った。

6 おわりに

本研究は商用チャットボットの対話管理エージェントの学習や改良するために、学習データの自動作成手法を提案した。作成されたデータは学習データとしての有効性が確認できて、オーダーメイド対話管理は実現できた。

参考文献

- [1] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. **Communications of the ACM**, Vol. 9, No. 1, pp. 36–45, 1966.
- [2] Xueliang Zhao, Xinting Huang, Wei Bi, and Lingpeng Kong. SEGO: Sequential subgoal optimization for mathematical problem-solving. In **Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 7544–7565, 2024.
- [3] Shanshan Wang, Derek Wong, Jingming Yao, and Lidia Chao. What is the best way for ChatGPT to translate poetry? In **Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 14025–14043, 2024.
- [4] Yibo Hu, Erick Skorupa Parolin, Latifur Khan, Patrick Brandt, Javier Osorio, and Vito D’Orazio. Leveraging codebook knowledge with NLI and ChatGPT for zero-shot political relation classification. In **Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 583–603, 2024.
- [5] Zhiruo Wang, Zhoujun Cheng, Hao Zhu, Daniel Fried, and Graham Neubig. What are tools anyway? a survey from the language model perspective. In **First Conference on Language Modeling**, 2024.
- [6] Aaron Parisi, Yao Zhao, and Noah Fiedel. Talm: Tool augmented language models. **arXiv preprint arXiv:2205.12255**, 2022.
- [7] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. Augmented language models: a survey. **arXiv preprint arXiv:2302.07842**, 2023.
- [8] Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. Toolalpaca: Generalized tool learning for language models with 3000 simulated cases. **arXiv preprint arXiv:2306.05301**, 2023.
- [9] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. **arXiv preprint arXiv:2305.15334**, 2023.
- [10] Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. GPT4tools: Teaching large language model to use tools via self-instruction. In **Thirty-seventh Conference on Neural Information Processing Systems**, 2023.
- [11] Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. Toolqa: A dataset for llm question answering with external tools. In **Advances in Neural Information Processing Systems**, Vol. 36, pp. 50117–50143, 2023.
- [12] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-bank: A comprehensive benchmark for tool-augmented LLMs. In **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing**, pp. 3102–3116, 2023.
- [13] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In **The Twelfth International Conference on Learning Representations**, 2024.
- [14] Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. Metatool benchmark for large language models: Deciding whether to use tools and which to use. In **The Twelfth International Conference on Learning Representations**, 2024.
- [15] Bing Liu, Zhou Jianxiang, Dan Meng, and Haonan Lu. An evaluation mechanism of LLM-based agents on manipulating APIs. In **Findings of the Association for Computational Linguistics: EMNLP 2024**, pp. 4649–4662, 2024.
- [16] Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong Sun, and Yang Liu. StableToolBench: Towards stable large-scale benchmarking on tool learning of large language models. In **Findings of the Association for Computational Linguistics: ACL 2024**, pp. 11143–11156, 2024.
- [17] Yifan Song, Weimin Xiong, Dawei Zhu, Wenhao Wu, Han Qian, Mingbo Song, Hailiang Huang, Cheng Li, Ke Wang, Rong Yao, et al. Restgpt: Connecting large language models with real-world restful apis. **arXiv preprint arXiv:2306.06624**, 2023.
- [18] Junjie Ye, Guanyu Li, Songyang Gao, Caishuang Huang, Yilong Wu, Sixian Li, Xiaoran Fan, Shihan Dou, Qi Zhang, Tao Gui, et al. Tooleyes: Fine-grained evaluation for tool learning capabilities of large language models in real-world scenarios. **arXiv preprint arXiv:2401.00741**, 2024.
- [19] Shijue Huang, Wanjun Zhong, Jianqiao Lu, Qi Zhu, Jiahui Gao, Weiwen Liu, Yutai Hou, Xingshan Zeng, Yasheng Wang, Lifeng Shang, Xin Jiang, Ruifeng Xu, and Qun Liu. Planning, creation, usage: Benchmarking LLMs for comprehensive tool utilization in real-world complex scenarios. In **Findings of the Association for Computational Linguistics: ACL 2024**, pp. 4363–4400, 2024.
- [20] Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, and Kai Chen. T-eval: Evaluating the tool utilization capability of large language models step by step. In **Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 9510–9529, 2024.
- [21] Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. **arXiv preprint arXiv:2309.10691**, 2023.
- [22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**, pp. 4171–4186, 2019.

表3 タスクの説明文とツールの説明文

| | | |
|---|--|---|
| <p>宿の予約 顧客が宿を予約するために、AIと対話している</p> | <p>extract.information: 顧客の発話から宿予約するために必要な情報を抽出する 入力: utterance.from.customer (顧客の発話) 出力: number.of.people (宿泊する人数)、name.of.representative (宿泊代表者の名前)、checkin.date (チェックインの日付)、checkout.date (チェックアウトの日付) check.information: 情報が揃っているかどうかをチェックする 入力: number.of.people (宿泊する人数)、name.of.representative (宿泊代表者の名前)、checkin.date (チェックインの日付)、checkout.date (チェックアウトの日付) 出力: unknown.information.list (未知な情報の一覧)</p> | <p>search.hotels: 顧客の要望に満たす宿を検索する 入力: number.of.people (宿泊する人数)、name.of.representative (宿泊代表者の名前)、checkin.date (チェックインの日付)、checkout.date (チェックアウトの日付) 出力: hotel.list (宿の一覧) hotel.reservation: 指定された宿を予約する 入力: name.of.hotel (宿の名前) 出力: status.of.reservation (宿を正常に予約できたかどうか)</p> |
| <p>問い合わせの自動返信 顧客の問い合わせに対して、自動的に返信を生成する</p> | <p>dissatisfaction.detector: 顧客の発話に苦情や不満があるかどうかを判定する。 入力: utterance.from.customer (顧客の発話) 出力: is.dissatisfaction (不満があるかどうか) dissatisfaction.handling: 不満がある場合、それを処理するための対策を決める。 入力: utterance.from.customer (顧客の発話) 出力: dissatisfaction.solution (不満を処理するための対策) intent.inference: 顧客の意図を推定する 入力: utterance.from.customer (顧客の発話) 出力: intent.label (顧客の意図のレベル) is.need.to.access.user.database: ユーザーの問い合わせを解決するために、データベースにアクセスする必要がありますか? 入力: utterance.from.customer (顧客の発話) 出力: is.need.to.access (データベースにアクセスする必要があるかどうか)</p> | <p>sql.query.generation: 問い合わせを解決するための情報を抽出する SQL を生成。 入力: utterance.from.customer (顧客の発話) 出力: sql.query (データベースから情報を抽出するための SQL query) retrieve.database.using.sql: SQL を用いて、データベースから情報を抽出。 入力: sql.query (sql query) 出力: information.retrieved (データベースから抽出した情報) is.ambiguous: 顧客の発話が曖昧であるかどうかを判定する 入力: utterance.from.customer (顧客の発話) 出力: is.ambiguous (顧客の発話は曖昧であるかどうか)</p> |
| <p>電車の乗換案内 電車で現在地から目的地まで最適なルート案内してください</p> | <p>find.nearest.station: 現在地や目的地の最寄駅を探す 入力: location (地名) 出力: station (最寄駅) extract.information: 顧客の発話から現在地と目的地情報を抽出する 入力: utterance.from.customer (顧客の発話) 出力: origin.location (現在地の地名)、destination.location (目的地の地名) check.information: 情報が揃っているかどうかをチェックする。揃っていない場合、未知な情報の一覧を返す。揃っている場合、空のリストを返す 入力: origin.location (現在地の地名)、destination.location (目的地の地名) 出力: unknown.information.list (未知な情報の一覧)</p> | <p>search.route: 電車で出発駅から到着駅まで最適なルートにある駅の一覧を返す。 入力: origin.station (出発駅)、destination.station (到着駅) 出力: route (途中経過した駅の一覧) search.transfer.stations: 駅からどの駅が乗り換え駅かを判定する 入力: list.of.stations (駅の一覧) 出力: list.of.transfer.stations (乗り換え駅の一覧)</p> |

表4 3つのタスクに対して作成されたデータをそれぞれ一例を示した。

| タスク | 対話 |
|------------|--|
| 宿の予約 | <p>User: こんにちは！宿の予約をしたいのですが、手続きをお手伝いいただけますか？ AI: もちろんです！宿の予約をお手伝いしますので、まずは以下の情報を教えてください：1. 宿泊する人数 2. 宿泊代表者の名前 3. チェックインの日付 4. チェックアウトの日付。お答えいただければ、予約手続きを進めることができます。 User: 1. 宿泊する人数は2人です。 2. 宿泊代表者は佐藤花子です。 3. チェックインは11月15日を予定しています。 4. チェックアウトは11月17日です。 AI: extract.information(utterance.from.user="1. 宿泊する人数は2人です。 2. 宿泊代表者は佐藤花子です。 3. チェックインは11月15日を予定しています。 4. チェックアウトは11月17日です。") Tool: {number.of.people:2,name.of.representative:"佐藤花子",checkin.date:"11/15",checkout.date:"11/17"} AI: check.information(number.of.people=2,name.of.representative="佐藤花子",checkin.date="11/15",checkout.date="11/17") Tool: unknown.information.list:[] AI: ありがとうございます。情報はすべて揃っています。それでは、お客様のご要望に合った宿を検索させていただきます。少々お待ちください。 AI: search.hotels(number.of.people=2,name.of.representative:"佐藤花子",checkin.date="11/15",checkout.date="11/17") Tool: {hotel.list:["Hotel Sakura","Hotel Fuji","Hotel Kiyomizu"]} AI: 以下のホテルが見つかりました。どの宿を予約されたいですか？ 1. Hotel Sakura 2. Hotel Fuji 3. Hotel Kiyomizu 選んでください。 User: Hotel Fuji を予約したいです。お願いいたします。 AI: hotel.reservation(name.of.hotel="Hotel Fuji") Tool: {status.of.reservation:true}</p> |
| 問い合わせの自動返信 | <p>User: こんにちは！実は商品について少し質問があるんですが、よろしいでしょうか？ AI: もちろんです！どの商品についての質問でしょうか？お手伝いできることがあれば教えてください。 User: ご丁寧ありがとうございます！実は、先日購入した商品が少し予想と違って、返品について知りたいんです。返品の手順を教えてくださいいただけますか？ AI: intent.inference(utterance.from.customer="先日購入した商品が少し予想と違って、返品について知りたいんです。返品の手順を教えてくださいいただけますか？") Tool: {intent.label:"返品に関する情報取得"} AI: 返品についての手順をお知らせいたします。1. 購入した商品の状態を確認してください。未使用で、購入時と同じ状態であることが基本的な条件です。2. 領収書や購入証明書を手に用意してください。返品手続きにはこれらの書類が必要です。3. 返品申し込みを行います。通常は、購入先のオンラインプラットフォームや店舗のカスタマーサポートに連絡してください。4. 商品を元の梱包材に入れ、返品先の住所に発送してください。なお、返品のルールやポリシーは購入先によって異なる場合がありますので、詳しくは購入先の方針をご確認ください。何か他に疑問やご不明点がありましたら、お気軽にお知らせください！ User: 詳しく教えていただきありがとうございます。返品の手続きについて、何かさらにわかることがありましたら是非教えてください。返品がスムーズにいくといいんですが、手続きがどうなっているかが私もまだ少し不安なんです。 AI: dissatisfaction.detector(utterance.from.customer="返品がスムーズにいくといいんですが、手続きがどうなっているかが私もまだ少し不安なんです。") Tool: {is.dissatisfaction:false} AI: ご心配なことがある場合は、まず購入先のカスタマーサポートへ直接お問い合わせされると良いでしょう。多くの場合、すぐに詳細な手順を案内してもらえますし、不明点や心配事についても丁寧に対応してくれるはずです。また、返品期限や手続きに必要な情報を事前に確認しておく、スムーズに進められることが多いです。それでも何かお困りのことがあれば、どうぞぜひご相談くださいな。</p> |
| 電車の乗換案内 | <p>User: こんにちは！実は電車で移動について教えてほしいのですが、いくつか質問があります。お力を貸していただけますか？ AI: もちろんです！どのような質問でもお聞きください。電車の移動についてお手伝いしますよ。まず、今の場所と行きたい目的地を教えてくださいませんか？どの区間を移動される予定ですか？ User: 現在いるのは東京駅で、目的地は新宿駅です。この区間の移動について知りたいです。どのルートが良いのでしょうか？ AI: search.route(origin.station="東京駅",destination.station="新宿駅") Tool: {route:["東京駅","神田駅","御茶ノ水駅","水道橋駅","飯田橋駅","市ヶ谷駅","四ツ谷駅","信濃町駅","千駄ヶ谷駅","代々木駅","新宿駅"]} AI: 東京駅から新宿駅までのルートは、中央・総武線(各駅停車)を利用するのが便利です。具体的には以下の順で各駅に停車します。1. 東京駅 2. 神田駅 3. 御茶ノ水駅 4. 水道橋駅 5. 飯田橋駅 6. 市ヶ谷駅 7. 四ツ谷駅 8. 信濃町駅 9. 千駄ヶ谷駅 10. 代々木駅 11. 新宿駅 中央・総武線を利用すると、東京駅から新宿駅までスムーズに移動できますよ。何か他にもご質問があれば教えてくださいな。 User: ご案内ありがとうございます。中央・総武線各駅停車での移動ですね。指定されたルートで確認します！</p> |

表5 プロンプトエンジニアリング前後 LLM の出力。

| LLM | LLM の出力 (修正前) | LLM の出力 (修正後) |
|----------------|--|---|
| 人間の返事を生成する LLM | <p>AI: こんにちは！宿の予約をお手伝いいたします。まずは、いくつかの情報が必要です。以下の項目を教えてくださいいただけますか？ 1. 宿泊する人数 2. 宿泊代表者の名前 3. チェックインの日付 4. チェックアウトの日付 これらの情報を教えていただければ、次のステップに進めます。 User: すみません、料金の範囲やご希望のエリアなども教えていただけたらありがたいです。他に特に何か希望条件があればそれとも一緒にお知らせくださいな。</p> | <p>AI: こんにちは！宿の予約をお手伝いします。ご希望の宿泊について、以下の情報を教えてくださいいただけますか？ 1. 宿泊する人数 2. 宿泊代表者の名前 3. チェックインの日付 4. チェックアウトの日付 これらの情報を元に、最適なプランを提案いたします。 User: じゃあ、まずは宿泊する人数は大人2人です。あと、チェックインは来月の15日、チェックアウトは17日を考えています。宿泊代表者の名前は田中太郎です。</p> |
| 対話の状態を判断する LLM | <p>AI: こんにちは！宿の予約をお手伝いしますので、何かご希望や質問がありましたらお気軽にお知らせください。 AI: こんにちは！宿の予約をお手伝いします。 何かご希望や質問がありましたらお気軽にお知らせください。どのような宿をご希望ですか？また、日付や人数なども教えていただけますと、より適切な情報を提供できます。</p> | <p>AI: どのような条件で宿を予約したいのか教えていただけますか？ 宿泊する人数やチェックイン・チェックアウトの日付など、具体的な情報をいただければと助かります。 User: 3名で宿を予約したいのですが、チェックイン日は12月15日、チェックアウト日は12月20日です。名前は田中太郎です。</p> |
| 対話の品質を判定する LLM | <p>【採用】 User: 1. 宿泊する人数は大人2人と子供1人です。2. 宿泊代表者の名前は佐藤太郎です。3. チェックインの日付は10月15日です。4. チェックアウトの日付は10月17日です。どんな選択肢があるか教えてくださいな。 AI: search.hotels(number.of.people=3,name.of.representative="佐藤太郎",checkin.date="10/15",checkout.date="10/17")</p> | <p>【採用】 User: 宿泊する人数は2人で、代表者は田中です。チェックインは12月24日、チェックアウトは12月26日です。 AI: check.information(number.of.people=2,name.of.representative="田中",checkin.date="12/24",checkout.date="12/26")</p> |