ProductAgent: Benchmarking Conversational Product Search Agent with Asking Clarification Questions

Anonymous ACL submission

Abstract

This paper introduces the task of *product demand clarification* within an e-commercial scenario, where the user commences the conversation with ambiguous queries and the task-oriented agent is designed to achieve more accurate and tailored product searching by asking clarification questions. To address this task, we propose **ProductAgent**, a conversational information seeking agent equipped with abilities of strategic clarification question generation and dynamic product retrieval. Specifically, we develop the agent with strategies for product feature summarization, query generation, and product retrieval. Furthermore, we propose the benchmark called **PROCLARE** to evaluate the agent's performance both automatically and qualitatively with the aid of a LLM-driven user simulator. Experiments show that ProductAgent interacts positively with the user and enhances retrieval performance with increasing dialogue turns, where user demands become gradually more explicit and detailed.[1]

1 Introduction

For decades, researchers have been attempting to develop agents with satisfying abilities of understanding human intentions and making decisions (Wooldridge and Jennings, 1995; Agha, 1986). However, these efforts have often been limited to constrained situations (Aliannejadi et al., 2021), where user-system interactions are insufficient (Vats et al., 2024). This limitation becomes more prominent and even critical especially in high-stakes domains or task-oriented systems (Yi et al., 2024), where users often struggle to express their ambiguous and personal information requirements in the first place (Rahmani et al., 2023).

A conversation example

A: I am a product sales assistant. Please just type in the targeted product category first, and I would search for you.

U: Sports shoes.

A: I have search several relevant items for you: <Item$_1$>, <Item$_2$>, $\cdots$. Also, you can answer these questions so that I could recommend more suitable items for you!
Q1: What is the suitable scenario for the sports shoes you want to purchase?
C1: ["Outdoor", "Brisk walking", "Basketball", "Dance", "Travel", "Other"]
Q2: What style of sports shoes do you like?
C2: ["Versatile", "Internet celebrity", "Foreign style", "Basketball", "Breathable style", "Other"]
Q3: Please choose your preferred color for sports shoes.
C3: ["Light gray", "White gray", "Moonlight", "Yellow", "Rose Red", "Other"]

U: **Answers for Q1**: "Outdoor", "Basketball".
Answers for Q2: "Basketball", "Breathable style".
Answers for Q3: I like green.

A: I have search several relevant items for you: <Item$_1$>, <Item$_2$>, $\cdots$. Also, you can answer these questions...

Table 1: A conversation example between ProductAgent (A) and the user (U). Since the clarification questions are formulated as multi-choice questions, the user can conveniently respond by selecting the preferred options or typing in the answers when needed.

The focus of this paper is e-commerce product search, where users usually spend excessive time in locating their desired products since they might not be familiar about which product aspects are suited to their individual use case. We claim that incorporating conversational information seeking could bridge the gaps between traditional product search and offline shopping, where expert salesmen can provide support in identifying the target products that are suitable to users (Papenmeier et al., 2022). Therefore, we propose the task of *product demand clarification* as illustrated in Table 1, which aims to investigate the role of *conversational information seeking* (CIS) in e-commerce product search (Zhang et al., 2018). Specifically, in each turn, the agent retrieves relevant product items and asks clarification questions (Lee et al., 2023) to

---

[1] All the source codes will be released after the review anonymity period.

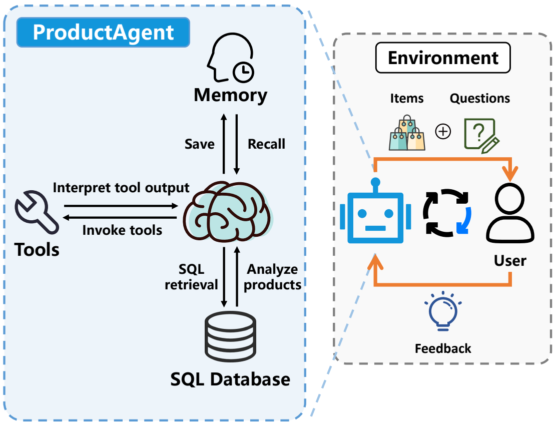

Figure 1: Simplified illustration of our ProductAgent.

refine user requirements. As the interaction progresses, it is expected that the user demands become clearer, leading to more accurate product searching. This task is challenging since it necessitates the agent to possess abilities of accessing extensive product knowledge (Peng et al., 2024; Li et al., 2023b), understanding subtle user intents, asking strategic clarification question, and retrieving relevant product items. Additionally, products often include different facets, such as brand, color, and style, resulting in different preference slot combinations or shopping schema (Yang et al., 2018).

To address this task, We propose **ProductAgent**, a conversational information seeking agent that can serve as a virtual shopping assistant. Technically, ProductAgent performs autonomous loops that integrate three core modules: product databases, a memory module, and a set of tools (Figure 1). Specifically, 1) the databases store product items in both structured and vectorized forms, enabling ProductAgent to retrieve relevant products and summarize them into statistics reflective of the current user demands. 2) The memory module caches a user's contextual information throughout a conversation session including structured clarification questions and unstructured dialogue records, enabling the agent to ask next questions dynamically. 3) The available tools support various actions such as product retrieval and clarification question generation. By leveraging carefully designed prompts, we enable effective interaction between language and machine learning models (LLMs) and these modules, automating the task.

In order to comprehensively evaluate ProductAgent, we introduce the **PROduct CLArifying REtrieval** (**PROCLARE**) Benchmark, which focuses on automatic conversational retrieval evaluation (Zhou et al., 2023; Semnani et al., 2023). This evaluation is implemented using the user simulation technique (Zhu et al., 2020; Yoon et al., 2024), in which a user simulator driven by a LLM (Terragni et al., 2023; Sekulić et al., 2024) makes it convenient and efficient to generate simulated user responses, thus eliminating the need for human utterances. Specifically, we synthesize a dataset consisting of 2,000 dialogues between ProductAgent and the user simulator, with each dialogue centered around a specific product item of the user's interest. Using this dataset, we build an automatic and reliable retrieval evaluation pipeline for the product demand clarification task. Our experiments in this paper showcase the significant effectiveness of ProductAgent, as the retrieval performance improves with the number of interaction turns. Our contributions are as follows:

(1) We formulate the task of product demand clarification and propose the solution called ProductAgent, which aims to pinpoint user demands by asking clarification questions in a conversational manner.

(2) We introduce an straightforward benchmark, called PROCLARE , to ensure a reliable and automatic evaluation pipeline for the product demand clarification task.

(3) Extensive experiments are conducted to confirm the effectiveness of our approach. Extra extensive analyses are also carried out to reveal promising improvements for the agents.

2 Related Work

2.1 LLM-based Agents

Building agents centered on LLMs has been widely studied recently (Park et al., 2023; Liu et al., 2023; Nakano et al., 2021), owing to their preeminent abilities of reasoning (Yao et al., 2023; Lu et al., 2024), planning (Song et al., 2023; Huang et al., 2023), memorizing, reflexion (Shinn et al., 2024; Huang et al., 2024) or self-correction (Li et al., 2024; Ye et al., 2023b,a), and tool utilization (Schick et al., 2024; Qin et al., 2023). Thanks to techniques like Chain-of-Thought (CoT) (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022) and problem decomposition, LLM-based agents are able to solve challenging problems across a wide rage of real-world scenarios (Li et al., 2023a).

However, many advanced agents focus on resolving real-world problems with minimal human intervention. This often results in bias (Ye et al., 2023c) and the lack of flexibility and inefficiency, particularly in task-oriented scenarios that emphasize interaction (Wang et al., 2023). The involvement of human feedback (Peng et al., 2023; Zhang et al., 2024a), therefore, becomes progressively critical in guiding and supervising the action trajectory of agents, ensuring their alignment with human requirements and objectives (Kenton et al., 2021; Ngo et al., 2022).

2.2 Asking Clarification Questions

Aiming to provide task-specific services, task-oriented agents often confront vague queries from users, who may lack the required domain knowledge to articulate their queries accurately. The ensuing reduced performance is mitigated in recent studies (Zamani et al., 2020; Kuhn et al., 2022), which advocate to ask clarification questions to pinpoint user demands in a conversational manner. For example, MAS2S (Feng et al., 2023)introduces the idea of asking questions based on user profiles and task knowledge to clarify user requests. Lee et al. (2023) presents the CAMBIGNQ dataset consisting of ambiguous questions. This dataset provides a foundation for a pipeline system enabling the asking of clarification questions to become more efficient. Furthermore, recent studies (Wu, 2024; Erbacher and Soulier, 2023; Zhang et al., 2024b) applies clarification questions onto LLMs, achieving significant improvements across various downstream applications. A concurrent study (Chang and Chen, 2024) develops SalesAgent that concentrates on injecting dialogue strategies into LLMs by fine-tuning, which is orthogonal to our work. Additionally, Vedula et al. (2024) also propose to generate product-related clarification questions through prompting and fine-tuning LLMs.

However, all these works rely heavily on the internal knowledge of LLMs to generate clarification questions, often compromising the quality of the generated clarification questions. On the contrary, our work leverages specific databases and external tools, thus unburdening LLMs from knowledge acquirement and promoting the generation of quality clarification questions.

3 Product Demand Clarification Task

The objective of the product demand clarification task is to enhance the e-commerce search process through the collaboration between humans and the agent. This task is based on a typical online shopping scenario that users (customers) have decided to purchase a product of a certain category but they have not yet determined detailed aspects of the product of their interests. So they often struggle to clearly articulate their purchasing requirements, especially for product categories accompanied by unfamiliar jargon. To tackle this challenge, the task aims to identify user demands by explicitly posing clarification questions that help clarify any uncertainties towards product searching.

The task takes the form of conversation, where an agent proactively interacts with a user. Specifically, given a product category as U_1, the conversation D between them is formulated as follows:

$$D = \{U_1, A_1, P_1, U_2, A_2, P_2, \cdots\} \qquad (1)$$

where U_t $(t = 1, 2, \cdots)$ is the user utterance, A_t is the agent utterance, and P_t is the retrieved product items. The agent utterance primarily consists of several multi-choice clarification questions $A_t = \{Q_{t1}, Q_{t2}, \cdots, Q_{tn}\}$[2]. Each question includes the content of the question itself and a set of answer candidates. It is important to note that, in each turn, the agent outputs both the questions and the retrieved items as timely feedback, facilitating a customer-friendly application interface.

The task presents several key challenges: 1) Generating non-trivial clarification questions requires a significant amount of product knowledge. 2) The agent should possess the ability to understand user intentions accurately in order to retrieve the most relevant product items and generate dynamic and valuable clarification questions. 3) The establishment of a quantitative and automated evaluation protocol is necessary to facilitate the development of effective agents for this task. This paper addresses the first two challenges in Section 4, and the third challenge in Section 5.

4 ProductAgent

4.1 Overview of ProductAgent

This section delineates the proposed solution ProductAgent for the introduced task. As shown in Figure 2, ProductAgent executes conversational loops, each of which can be decomposed by three main steps. Before initializing the loops, the user commences the conversation by inputting a product category as a naive query. We will provide an

---

[2] We set $n = 3$ in our implementation.

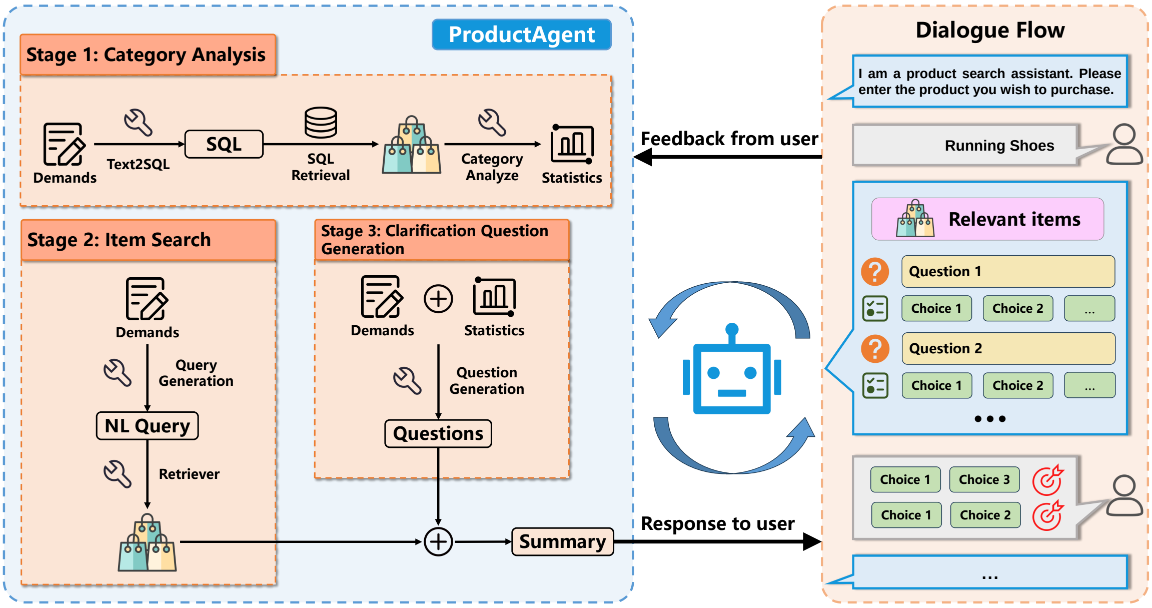

Figure 2: Overview of our proposed ProductAgent.

overview in this section and detailed implementation is discussed in Section 4.2.

Category Analysis. To generate appropriate clarification questions that can determine the user demands efficiently, it is necessary to collect and summarize relevant and well-elaborated product knowledge based on known user demands of each turn. To achieve this, the agent first synthesizes a query reflective of all the known user demands, and the query is used to retrieve relevant product items from databases. The retrieved items, which probably cover the desirable items of interest, are then summarized as statistics. These statistics encompass various product aspects and serve as a dynamic knowledge base that is constructed in real time based on the ongoing user demands, laying a foundation for the subsequent steps.

Item Search. We dynamically retrieve different relevant product items based on all the known demands. Specifically, ProductAgent first generates a natural language query based on the current demands by harnessing the Query Generation tool. The generated query is then used to retrieve the most suitable items that meet the user demands.

Clarification Question Generation. In this step, the agent poses several new clarification questions along with their answer candidates which are carefully curated by incorporating the knowledgeable statistics and known user demands. The user is then prompted to efficiently clarify their requirements by either selecting options or providing additional personal information that reflects their interests in the interaction interface. This user feedback is valuable for the task as it helps improve the accuracy of the product search process.

The above steps are repeated in each turn of the dialogue, with the aim of making the user demands in subsequent turns more precise compared to previous turns. Consequently, the accuracy of the product search steadily improves over time.

4.2 Key Implementation

Our proposed ProductAgent consists of three key components which work collectively to solve the product demand clarification task.

Databases. We store all product items in both SQL and dense vector datasets, allowing for flexible retrieval strategies. As shown in Figure 2, we employ SQL retrieval in the stage 1 since it enables us to efficiently retrieve items through exact matching. On the other hand, we leverage non-SQL retrieval in the stage 2 because non-SQL retrievers can return product items ordered by their relevance to the user demands, which cannot be typically accomplished by SQL retrieval.

Tools. We define 5 tools for ProductAgent, which are listed in Table 2. To enable SQL retrieval, ProductAgent uses the Text2SQL tool. This tool prompts LLMs with the history demands to generate practical SQL queries. The synthesized SQL query is then used to retrieve product items rel-

Tool	Description	Input	Output
Text2SQL	Generate SQL query	Demands	SQL Query
Category Analyze	Summarize a certain category	Product items	Category statistics
Query Generation	Generate NL query	Demands	NL query
Retriever	Retrieve items relevant to demands	NL query	Product items
Question Generation	Generate clarification questions	Demands + Category statistics	Clarification questions

Table 2: Available tools of ProductAgent.

evant to the current known demands. Then, the Category Analyzing tool is performed to generate summarized statistics. For non-SQL retrieval, we create the Query Generation tool, which generates concise natural language queries. Additionally, in stage 3, we utilize the Question Generation tool to fabricate reasonable clarification questions based on the summarized statistics and the user demands. To ensure the desired output format, we specify the json description for ProductAgent.

It should be kindly noted that for the purpose of autonomous evaluation, the demand clarification task has been slightly simplified and hence does not require a tool router, which is typically used to select the tool for the next action.

Memory. All the user demands, including clarification questions and corresponding answers, are stored as structured objects in the memory module. When the agent receives responses from users, it extracts the answers of clarification questions posed in the previous turn. The Q&A pairs are then saved as structured demand objects. Additionally, the memory module stores the dialogue records and searching history. When invoking certain tools, we integrate only the required information selected from the memory into the designated slots within the prompt. The memory are continuously updated as the dialogue progresses, ensuring that the intelligent agent can effectively handle dynamically user requirements. The design is crucial as it enables the agent to recall user preferences, thereby providing personalized assistance.

5 The PROCLARE Benchmark

Most existing information retrieval benchmarks focus on static search processes (Qiu et al., 2022; Long et al., 2022) that do not involve human-AI interactions, which may limit their effectiveness in evaluating practical and interactive performance of search systems. Therefore, with the aim of comprehensively investigating the performance of conversational e-commercial search agents in an automatic manner, we introduce PROCLARE , a benchmarking framework that evaluates retrieval performance in two distinct settings, namely the *traditional* setting (Section 5.2) and the *conversational* (Section 5.4) setting. Both ProductAgent and PROCLARE are implemented using the open-sourced framework *LlamaIndex*.

Setting	Documents	Query	Len. of Query
Trad.	1,000,000	2,000	27.02
Conv.	1,000,000	10,000	8.59 / 16.45 / 27.40 / 37.33 / 45.03

Table 3: Statistics of the PROCLARE benchmark. We report the average length of queries generated by GPT-4.

5.1 Document Set

We begin by creating a extensive collection of documents (product items), which serves as the knowledge base for our benchmark. This collection consists of 1,000,000 documents across 20 categories. To construct this dataset, we utilize AliMe KG (Li et al., 2020), a domain knowledge graph in the field of e-commerce that supports the top-50 main categories on the Alibaba e-commerce platform. From each category, we randomly sample 50,000 documents, resulting in the dataset used for our benchmark. To ensure efficient data management, we perform Named Entity Recognition (NER) preprocessing (Wang et al., 2021) on all the data. This results in a unified feature definition across various product categories, enabling us to store all the documents in a relational SQL dataset. For detailed statistics and description of the dataset, please refer to Table 3 and Appendix A.1.

5.2 Traditional Information Retrieval

In the traditional setting of PROCLARE, we evaluate the retrieval performance of the conventional product search task using the methodology employed in previous work (Long et al., 2022; Qiu et al., 2022). It is important to note that this experimental setting, which serves as a counterpart to the conversation setting introduced in Section 3, *does* *not* involve the use of the agent. To construct queries used for evaluation, we randomly sample 100 documents per category from the entire document set, resulting in a total of 2000 documents. We then leverage the *Doc2Query* technique (Gospodinov et al., 2023; Alaofi et al., 2023) to transform these selected documents into synthesized queries by prompting GPT-4. Subsequently, we measure the retrieval performance of different retrieval methods based on the synthesized queries. The prompt for Doc2Query and examples of synthesized queries can be found in Appendix B.

5.3 Retrieval Strategies

Our experiments employ three widely-adopted retrievers: BM25 (Robertson et al., 2009), General Text Embedding (GTE)[3], and CoROM (Long et al., 2022). Besides, we also incorporate the techniques of reranking and fusion into the retrievers. The detailed descriptions are provided in Appendix A.2.

5.4 Conversational Information Retrieval

The performance of ProductAgent can be comprehensively explored in the conversational setting, where its effectiveness is manifested directly via the retrieval performance across dialogue turns.

User simulation. It is well-acknowledged that the nature of dialogues, including the high cost of dialogue annotation (Schatzmann et al., 2005; Keizer et al., 2010), poses a challenge for the efficient evaluation of dialogue systems, particularly in an end-to-end manner. To handle this issue, we harness LLMs as intelligent user simulators, thereby accomplishing the automatic evaluation of ProductAgent. Specifically, a LLM-driven user simulator is presented with several clarification questions along with their corresponding candidates in each turn, and then is prompted to answer the questions with access to the ground truth item. It is worth noticing that to avoid potential information leakage, which may create shortcuts for the agent, we restrict the user simulator to only answer the given questions with provided candidates, thus simulating a more realistic and reliable situation.

For each queried document, we simulate 10 turns (5 user turns and 5 agent turns) with the user simulator starting the conversation by inputting the desired product category. Examples of simulated dialogues and the prompt for user simulation can be found in Appendix B.

Retriever	Retrieve		Rerank
	HIT@10	MRR@10	MRR@10
BM25	34.80	26.18	33.35
GTE	**69.00**	**52.29**	**63.82**
CoROM	61.65	44.95	57.41
BM25 + GTE	37.70	26.69	35.81
BM25 + CoROM	37.35	26.77	35.65
GTE + CoROM	59.45	26.91	55.90

Table 4: Retrieval performance of the traditional setting.

5.5 Automatic evaluation metrics

Drawn from the evaluation methodology adopted in earlier information retrieval benchmarks (Long et al., 2022), we evaluate the retrieval performance in both settings using Mean Reciprocal Rank at 10 documents (MRR@10) and hit rate at 10 documents (HIT@10). For the reranking results, we only report the result of the MRR@10 metric since the hit rates are identical to their counterparts.

6 Experiments

6.1 Results of Traditional Setting

The overall experimental results of the traditional setting are reported in Table 4, from which we can make the following conclusions.

Dense retrievers outperform BM25. In order to generate diverse literal expressions in the process of synthesizing queries, we deploy LLMs. This, however, presents a challenge to sparse retrievers such as BM25. Dense retrievers such as GTE and CoROM, which leverage semantic matching, enhance retrieval performance on the MRR@10 scores by notable 26.11% and 18.11%, respectively. It should be noted that the efficiency of the retrievers might be affected by the Doc2Query prompt in our attempt to simulate realistic user queries. Furthermore, we observe that CoROM, despite being trained on the e-commerce dataset, falls behind GTE – trained on generic datasets – in performance. We suspect that this is because GTE, due to its pre-training and contrastive fine-tuning stages, is better equipped to capture complex semantics.

Reranking can significantly improve the MRR scores. The results indicate that reranking can greatly boost the MRR@10 scores for all retrievers and retriever combinations by 7.17%, 11.53%, and

---

[3] `https://help.aliyun.com/zh/dashscope/developer-reference/text-embedding-quick-start`

LLM	Retriever	Retrieve		Rerank
		HIT@10	MRR@10	MRR@10
GPT-3.5	**BM25**	**35.04**	**27.26**	**25.69**
	GTE	8.49	4.95	7.18
	CoROM	12.48	7.96	10.79
	B + G	32.35	16.12	25.21
	B + C	32.54	17.66	25.27
	G + C	7.61	4.21	6.67
GPT-4	**BM25**	**39.48**	**32.00**	**30.20**
	GTE	8.27	4.92	7.17
	CoROM	13.86	9.11	12.54
	B + G	36.93	18.36	29.91
	B + C	37.02	20.37	30.13
	G + C	7.57	4.15	6.80
Qwen	**BM25**	**31.58**	**25.24**	**24.85**
	GTE	16.45	10.56	13.82
	CoROM	20.71	13.80	17.77
	B + G	30.65	16.78	24.61
	B + C	30.79	17.98	24.79
	G + C	14.26	7.74	12.36

Table 5: Retrieval performance of the conversational setting. We report results of fusion retrievers represented as X + Y, such as B + G (BM25 + GTE).

12.46% for BM25, GTE, and CoROM, respectively. It also shows similar trends for retriever fusion.

Retriever fusion decreases both HIT@10 and MRR@10 scores. We have also tested additional fusion strategies assisted by LlamaIndex in our initial experiments. The reciprocal rerank fusion algorithm outperforms others in most instances. This suggests that naive fusion strategies are unable to enhance retrieval performance in this case.

6.2 Results of Conversational Setting

In the conversational setting of ProClare, we employ OpenAI GPT-3.5-turbo-0125 to power the user simulator for all experiments, ensuring a fair comparison for ProductAgent built on various LLM backbones. We benchmark a series of agents with Qwen-max-0107[4], GPT-3.5-turbo-0125, and GPT-4-turbo-0409 as the backbones in this paper.

The results of conversational retrieval are detailed in Table 5, with several distinctions compared to the traditional setting.

BM25 surpasses dense retrievers. Surprisingly, dense retrievers, which excel in the traditional setting, yield significantly lower performance than BM25 across all LLMs. This can be attributed to the fact that synthesized queries are based on user responses, most of which are essentially provided choices of clarification questions without much modification. Consequently, the challenge faced by BM25 - computing relevance scores through exact term matching - is eliminated. This underlines the importance of generating queries with a specific pattern that is preferred by the utilized retriever.

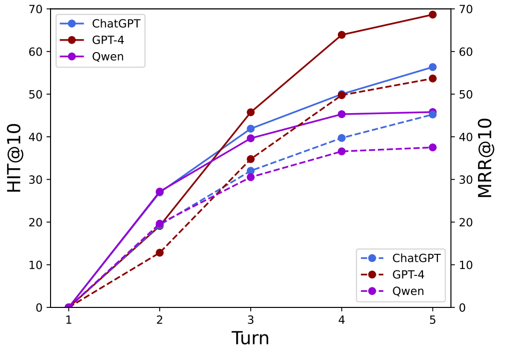

Figure 3: Retrieval performance of ProductAgent driven by different LLMs with increasing turns using BM25.

Reranking boosts the MRR@10 scores, except those of BM25. We hypothesize that the level of noise in queries is so minimal that BM25 does not require the further application of an additional reranker. Other retrievers, however, struggle in capturing subtle semantics, leaving room for the reranker to augment their performance. In the same vein, the retriever fusion technique does *not* enhance retrieval performance.

7 Analysis

7.1 Effect of Interaction Turns

Figure 3 presents the trends in retrieval performance as interaction turns increase. The first turn can not retrieve any ground truth product items due to the access to only product categories. We observe that as the number of turns increase, all retrieval scores improve regardless of the types of retrievers employed, signifying the effectiveness of posing clarification questions. Notably, GPT-4 scores lower at the second turn but outperforms all the other LLMs after the third turn, which means GPT-4 tend to ask clarification questions from a global perspective.

7.2 Performance Increase of Clarification

To further demonstrate the effectiveness of clarification, we replace the initial user queries, i.e., product categories, with the synthesized queries introduced in Section 5.2. The results illustrated in Figure 4

---

[4] `https://help.aliyun.com/zh/dashscope/developer-reference/quick-start`

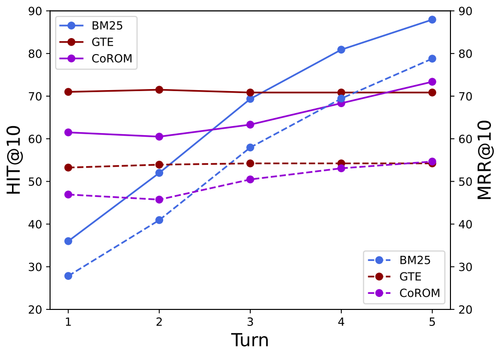

Figure 4: Retrieval performance of the variant of the conversational setting. All scores are from ProductAgent driven by GPT-4.

Retriever	HIT@10	MRR@10
w/o Statistics	15.60	10.69
Random	39.50	19.54
BM25	**47.00**	**38.51**
CoROM	45.00	38.09
SQL (Default)	39.90	32.40

Table 6: Ablation results of different strategies of acquiring statistics.

	Invalid SQL %	Trivial SQL %
GPT-3.5	1.21	54.59
GPT-4	3.52	55.36
Qwen-max	3.06	44.92

Table 7: Failure rates of Text2SQL.

indicate that retrievers benefit further from clarification, except for GTE. In detail, the performance of BM25 improves dramatically, similar to the results of conversational setting in Section 6.2. However, the performance of GTE falls into a plateau. We speculate the reason is that general-purpose retrievers fail to capture subtle product features that are rare in their training data.

7.3 Ablation Study

Considering that summarized product statistics constitute a crucial part of the context for generating clarification questions, we perform a comprehensive ablation study focusing on strategies of acquiring statistics. This is conducted with the aim of probing the influence of dynamic statistics on ProductAgent. Specifically, we experiment with three different strategies to derive the statistics: 1) for the "w/o Statistics" variant, we do not supply statistics to the process of clarification question generation. 2) For the "Random" variant, we randomly sample product items to summarize the statistics, ignoring dynamic user demands. 3) For the "BM25" variant, we substitute the SQL retriever with BM25, and 4) for the "CoROM" variant, similarly, we replace the SQL retriever with CoROM.

The ablation study results, which are shown in Table 6, reveal that BM25 and CoROM retrievers are better suited for statistic summarization. Additionally, both removing statistics or ignoring real-time user demands can impair the performance of product search. We analyze further this phenomenon in Section 7.4.

7.4 Failures of ProductAgent

We identify several failure types for ProductAgent throughout our experiments, all of which are due to the generation of either invalid or trivial SQL queries when using the Text2SQL tool. Specifically, invalid SQL queries mean Text2SQL generates queries that cannot be executed, while trivial SQL queries refer to those that retrieve nothing. The occurrence rates of both kinds of failures are presented in Table 7, which reveals that all the involved LLMs have a tendency to produce trivial SQL queries. Delving deeper into the generated SQL queries, we discovered that trivial ones are typically produced in later turns, whereby LLMs form complex queries by merely amalgamating all known user demands. We addressed these issues by creating statistics derived from previously retrieved items, leading to ProductAgent sometimes failing to propose valuable questions directly related to the user's real-time demands. In the future, the exploration of more sophisticated SQL generation is worth investigating to enhance both ProductAgent's persona and effectiveness.

Additionally, we observe ProductAgent tends to generate clarification questions when conversations progress further. So we analyze the similarity of synthesized clarification questions, and the results are reported in Appendix A.3.

8 Conclusion

This paper formulates the task of product demand clarification and presents the solution called ProductAgent. In order to implement the evaluation in an automatic and quantitative manner , we introduce the ProClare benchmark with the aid of a user simulator. Experiments indicate that ProductAgent significantly enhances the retrieval performance with increasing dialogue turns. We also identify certain limitations in the agent.

Limitations

Absence of human users. In this paper, we employ the user simulation technique to facilitate a more consistent and standardized evaluation process. Utilizing this approach allows for all the experiments conducted within our research to be completely void of any human involvement, contributing to cost-efficiency and saving time. We also acknowledge that potentially unidentified biases may arise from the hallucination effect of LLM-based user simulators within our evaluation pipeline.

Limitations of datasets. Our dataset was derived from the AliMe KG, encompassing multiple dimensions of products. Nonetheless, it does not provide comprehensive information that could be crucial to customers' purchase decisions, such as product pricing and customer reviews. Future research could delve into more diverse and higher-quality datasets to overcome these deficiencies.

Controllability of clarification questions. We encourage ProductAgent to ask clarification questions following in-context learning augmented with summarized statistics, without explicit and strong constraints. This strategy is not always optimal for all situations. In the future work, we will explore more effective planning strategies that can handle the task from a global perspective, providing a pathway for potential enhancements to create more intelligent conversational agents.

Lack of analysis of prompt sensitivity. The impact of prompt design on the performance of LLMs, often demonstrated as prompt sensitivity, is widely recognized. In our research, however, we did not conduct a systematic examination concerning the sensitivity of prompts used to develop the agent and user simulator. Detailed engineering of prompts could serve as a valuable measure for enhancing the performance of the ProductAgent in future studies.

Ethics Statement

We are aware that our proposed ProductAgent may be potentially misused for improper purposes, such as privacy data collection and excessive propaganda. However, this vulnerability is not unique to our approach but a common threat to many LLM-based applications. It also highlights the significance of conducting appropriate regulations and enhancing the safety of LLMs in the future.

References

Gul Agha. 1986. *Actors: a model of concurrent computation in distributed systems*. MIT press.

Marwah Alaofi, Luke Gallagher, Mark Sanderson, Falk Scholer, and Paul Thomas. 2023. Can generative llms create query variants for test collections? an exploratory study. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1869–1873.

Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2021. Building and evaluating open-domain dialogue corpora with clarifying questions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4473–4484, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wen-Yu Chang and Yun-Nung Chen. 2024. Injecting salesperson's dialogue strategies in large language models with chain-of-thought reasoning. *arXiv preprint arXiv:2404.18564*.

Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Pierre Erbacher and Laure Soulier. 2023. Circle: Multi-turn query clarifications with reinforcement learning. *arXiv preprint arXiv:2311.02737*.

Yue Feng, Hossein A Rahmani, Aldo Lipani, and Emine Yilmaz. 2023. Towards asking clarification questions for information seeking on task-oriented dialogues. *arXiv preprint arXiv:2305.13690*.

Mitko Gospodinov, Sean MacAvaney, and Craig Macdonald. 2023. Doc2query–: When less is more. In *European Conference on Information Retrieval*, pages 414–422. Springer.

Haojing Huang, Jingheng Ye, Qingyu Zhou, Yinghui Li, Yangning Li, Feng Zhou, and Hai-Tao Zheng. 2023. A frustratingly easy plug-and-play detection-and-reasoning module for Chinese spelling check. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore.

Xiang Huang, Sitao Cheng, Shanshan Huang, Jiayu Shen, Yong Xu, Chaoyun Zhang, and Yuzhong Qu. 2024. Queryagent: A reliable and efficient reasoning framework with environmental feedback based self-correction.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Simon Keizer, Milica Gasic, Filip Jurcicek, François Mairesse, Blaise Thomson, Kai Yu, and Steve Young. 2010. Parameter estimation for agenda-based user simulation. In *Proceedings of the SIGDIAL 2010 Conference*, pages 116–123.

Zachary Kenton, Tom Everitt, Laura Weidinger, Iason Gabriel, Vladimir Mikulik, and Geoffrey Irving. 2021. Alignment of language agents. *arXiv preprint arXiv:2103.14659*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2022. Clam: Selective clarification for ambiguous questions with generative language models. *arXiv preprint arXiv:2212.07769*.

Dongryeol Lee, Segwang Kim, Minwoo Lee, Hwanhee Lee, Joonsuk Park, Sang-Woo Lee, and Kyomin Jung. 2023. Asking clarification questions to handle ambiguity in open-domain QA. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11526–11544, Singapore. Association for Computational Linguistics.

Feng-Lin Li, Hehong Chen, Guohai Xu, Tian Qiu, Feng Ji, Ji Zhang, and Haiqing Chen. 2020. Alimekg: Domain knowledge graph construction and application in e-commerce. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2581–2588.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023a. Camel: Communicative agents for" mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*.

Yangning Li, Shirong Ma, Xiaobin Wang, Shen Huang, Chengyue Jiang, Hai-Tao Zheng, Pengjun Xie, Fei Huang, and Yong Jiang. 2023b. Ecomgpt: Instruction-tuning large language model with chain-of-task tasks for e-commerce. *arXiv preprint arXiv:2308.06966*.

Yinghui Li, Shang Qin, Jingheng Ye, Shirong Ma, Yangning Li, Libo Qin, Xuming Hu, Wenhao Jiang, Hai-Tao Zheng, and Philip S Yu. 2024. Rethinking the roles of large language models in chinese grammatical error correction. *arXiv preprint arXiv:2402.11420*.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023c. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

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 arXiv:2305.16960*.

Dingkun Long, Qiong Gao, Kuan Zou, Guangwei Xu, Pengjun Xie, Ruijie Guo, Jian Xu, Guanjun Jiang, Luxi Xing, and Ping Yang. 2022. Multi-cpr: A multi domain chinese dataset for passage retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3046–3056.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2024. Chameleon: Plug-and-play compositional reasoning with large language models. *Advances in Neural Information Processing Systems*, 36.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Richard Ngo, Lawrence Chan, and Sören Mindermann. 2022. The alignment problem from a deep learning perspective. *arXiv preprint arXiv:2209.00626*.

Andrea Papenmeier, Alexander Frummet, and Dagmar Kern. 2022. “mhm...”–conversational strategies for product search assistants. In *Proceedings of the 2022 Conference on Human Information Interaction and Retrieval*, pages 36–46.

Joon Sung Park, Joseph 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*, pages 1–22.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*.

Bo Peng, Xinyi Ling, Ziru Chen, Huan Sun, and Xia Ning. 2024. ecellm: Generalizing large language models for e-commerce from large-scale, high-quality instruction data.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, et al. 2023. Tool learning with foundation models. *arXiv preprint arXiv:2304.08354*.

Yifu Qiu, Hongyu Li, Yingqi Qu, Ying Chen, QiaoQiao She, Jing Liu, Hua Wu, and Haifeng Wang. 2022. DuReader-retrieval: A large-scale Chinese benchmark for passage retrieval from web search engine. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5326–5338, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hossein A. Rahmani, Xi Wang, Yue Feng, Qiang Zhang, Emine Yilmaz, and Aldo Lipani. 2023. A survey on asking clarification questions datasets in conversational systems. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2698–2716, Toronto, Canada. Association for Computational Linguistics.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Jost Schatzmann, Kallirroi Georgila, and Steve Young. 2005. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 45–54.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2024. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36.

Ivan Sekulić, Silvia Terragni, Victor Guimarães, Nghia Khau, Bruna Guedes, Modestas Filipavicius, André Ferreira Manso, and Roland Mathis. 2024. Reliable llm-based user simulator for task-oriented dialogue systems. *arXiv preprint arXiv:2402.13374*.

Sina Semnani, Violet Yao, Heidi Chenyu Zhang, and Monica Lam. 2023. Wikichat: Stopping the hallucination of large language model chatbots by few-shot grounding on wikipedia. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009.

Silvia Terragni, Modestas Filipavicius, Nghia Khau, Bruna Guedes, André Manso, and Roland Mathis. 2023. In-context learning user simulators for task-oriented dialog systems. *arXiv preprint arXiv:2306.00774*.

Vanshika Vats, Marzia Binta Nizam, Minghao Liu, Ziyuan Wang, Richard Ho, Mohnish Sai Prasad, Vincent Titterton, Sai Venkat Malreddy, Riya Aggarwal, Yanwen Xu, et al. 2024. A survey on human-ai teaming with large pre-trained models. *arXiv preprint arXiv:2403.04931*.

Nikhita Vedula, Oleg Rokhlenko, and Shervin Malmasi. 2024. Question suggestion for conversational shopping assistants using product metadata. *arXiv preprint arXiv:2405.01738*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1800–1812, Online. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Michael Wooldridge and Nicholas R Jennings. 1995. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(2):115–152.

Jie JW Wu. 2024. Large language models should ask clarifying questions to increase confidence in generated code.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding.

Yunlun Yang, Yu Gong, and Xi Chen. 2018. Query tracking for e-commerce conversational search: A machine comprehension perspective. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1755–1758.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Jingheng Ye, Yinghui Li, Yangning Li, and Hai-Tao Zheng. 2023a. MixEdit: Revisiting data augmentation and beyond for grammatical error correction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10161–10175, Singapore. Association for Computational Linguistics.

Jingheng Ye, Yinghui Li, and Haitao Zheng. 2023b. System report for CCL23-eval task 7: THU KELab (sz) - exploring data augmentation and denoising for Chinese grammatical error correction. In *Proceedings of the 22nd Chinese National Conference on Computational Linguistics (Volume 3: Evaluations)*, pages 262–270, Harbin, China. Chinese Information Processing Society of China.

Jingheng Ye, Yinghui Li, Qingyu Zhou, Yangning Li, Shirong Ma, Hai-Tao Zheng, and Ying Shen. 2023c. CLEME: Debiasing multi-reference evaluation for grammatical error correction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6174–6189, Singapore. Association for Computational Linguistics.

Zihao Yi, Jiarui Ouyang, Yuwen Liu, Tianhao Liao, Zhe Xu, and Ying Shen. 2024. A survey on recent advances in llm-based multi-turn dialogue systems. *arXiv preprint arXiv:2402.18013*.

Se-eun Yoon, Zhankui He, Jessica Maria Echterhoff, and Julian McAuley. 2024. Evaluating large language models as generative user simulators for conversational recommendation. *arXiv preprint arXiv:2403.09738*.

Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *Proceedings of the web conference 2020*, pages 418–428.

Shaokun Zhang, Jieyu Zhang, Jiale Liu, Linxin Song, Chi Wang, Ranjay Krishna, and Qingyun Wu. 2024a. Training language model agents without modifying language models. *arXiv preprint arXiv:2402.11359*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. 2024b. Clamber: A benchmark of identifying and clarifying ambiguous information needs in large language models. *arXiv preprint arXiv:2405.12063*.

Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th acm international conference on information and knowledge management*, pages 177–186.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. ConvLab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 142–149, Online. Association for Computational Linguistics.

A Experimental Details

A.1 NER Pre-processing

To accomplish the retrieval process in a consistent manner for different product categories, we initially apply named entity recognition (NER) to the documents obtained from AliMe KG. For this paper, we chose to use an e-commerce specific NER model (Wang et al., 2021) trained with a Cooperative Learning objective. This coaching strategy encourages two input views to generate similar contextual representations or output label distributions. The NER processing transform all the data from various categories into structured documents with a consistent named entity schema consisting of 54 hierarchical entity labels. We then compress this label space into a new one composed of only 10 entity labels, which allows the agent to retrieve documents more efficiently and accurately from the SQL database. The details of the product item description are provided in Table 8.

A.2 Details about Retrievers

BM25. As the representative of sparse retrievers, BM25 (Robertson et al., 2009) consistently exhibits impressive performance across diverse information retrieval benchmarks (Long et al., 2022).

GTE. The General Text Embedding (GTE) model is a general-purpose text embedding model

Name	Description	Type
Category	Category of the item.	str
Brand	Brand of the item.	List[str]
Series	Series of the item.	List[str]
Target Customer	Target customers of the item.	List[str]
Applicable Scenario	Applicable scenarios of the item.	List[str]
Decorative Attribute	Decorative attributes of the item.	List[str]
Material	Material of the item.	List[str]
Style	Styles of the item.	List[str]
Specification	Available specification of the item.	List[str]
Color	Available colors of the item.	List[str]
Function	Function of the item.	List[str]

Table 8: Product item description.

trained with the multi-stage contrastive learning objective (Li et al., 2023c). It has shown exceptional results in the Massive Text Embedding Benchmark (Muennighoff et al., 2023).

CoROM. The CoROM model (Long et al., 2022) uses BERT-base (Devlin et al., 2019) as its backbone. It is a dual Encoder-based embedding model (Karpukhin et al., 2020) trained on annotated query-passage e-commerce datasets.

Reranker. We also integrate the lightweight bge-reranker-base model (Xiao et al., 2023)[5] to rerank the top-k documents retrieved.

Fusion. Furthermore, we also seek to investigate whether the fusion of multiple heterogeneous retrievers can boost performance on our benchmark. Consequently, we test hybrid fusion retrievers that utilize the reciprocal rerank fusion algorithm (Cormack et al., 2009) without requiring any additional models or significant computation.

A.3 Similarity of Synthesized Clarification Questions

We observe ProductAgent tends to generate clarification questions that overlap with those previously presented, which can negatively impact the task effectiveness and overall user satisfaction. Hence, we take a qualitative approach to measuring the similarity among clarification questions proposed in different dialogue turns by using BERTScore (Zhang* et al., 2020). BERTScore computes a similarity score for each token in the candidate sentence by comparing it to each token in the reference sentence. In the experiment, we concatenate the question content and its answer choices into a sentence. By treating all other questions and choices as references, We compute the similarity score for each synthesized clarification questions.

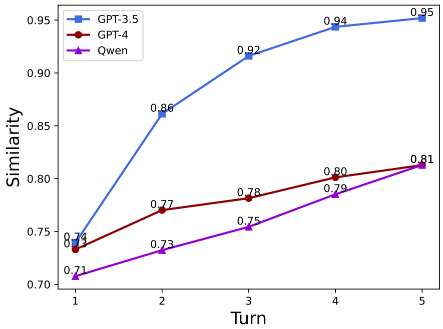

Figure 5: Similarity of synthesized clarification questions computed by BERTScore.

As Figure 5 illustrates, we discern an upward trend in the similarity of synthesized clarification questions as dialogues progress. This suggests that although retrieval performance continues to enhance over ten dialogue turns, we may be still able to improve further the performance by boosting the diversity of clarification questions. Additionally, we observe that the agent powered by GPT-3.5 is more inclined to fabricate similar questions than those powered by GPT-4 and Qwen-max, despite the fact that the former achieves higher retrieval scores than the agent powered by Qwen-max when both utilize BM25 for item search.

A.4 Product Aspects of Clarification Questions

We observe most clarification questions focus on one specific product aspects indicated in Table 8, such as category, brand, and series. Therefore, we analyze the proportion of product facets to which clarification questions are related. The results, illustrated in Figure 6, show that all three LLMs tend to generate clarification questions with even distributions. To detail, three aspects, namely applicable scenarios, style, function, respectively account for more than 10% across LLMs. On the other hand, LLMs are less likely to clarify on series and specification, which may be too specialized for users. This reveals that LLMs naturally have a preference for asking specific types clarification questions in this task, since we do not incorporate any explicit constrains on clarification question generation.

B Prompts and Examples

B.1 Prompts for ProductAgent

We report all the prompts used in our proposed ProductAgent, including Text2SQL in Table 9, Query

---

[5] `https://huggingface.co/BAAI/bge-reranker-base`

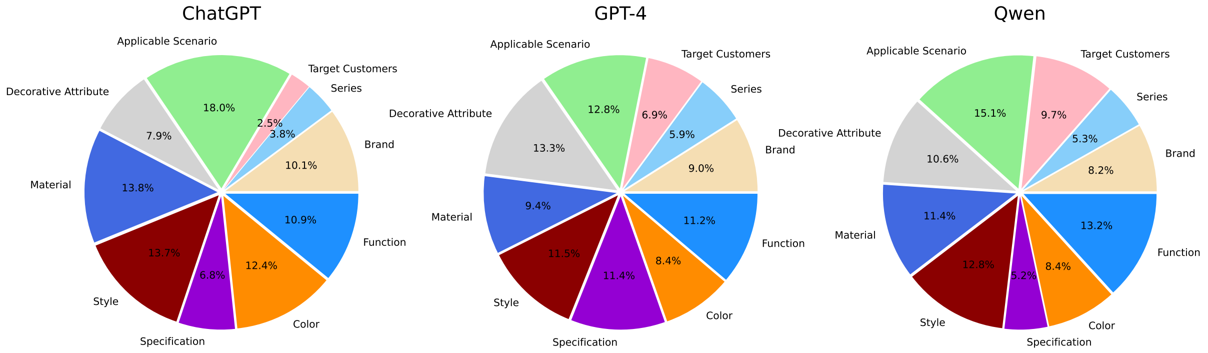

Figure 6: Proportion of product aspects of clarification questions generated by three LLMs.

Generation in Table 10 and Clarification Question Generation in Table 11.

B.2 Prompt for User Simulator

We list the prompt for user simulation in Table 12.

B.3 Examples of Conversation

Here we provide a detailed conversation example regarding “Canvas shoes" in Table 13. We can notice that ProductAgent first generates some critical clarification questions, such as the color, applicable scenarios and functions, which are helpful to precisely identify the basic user demands. Subsequently, the agent probes for additional specifics like style, decorative attributes and material. On the other hand, the user simulator, which is driven by GPT-3.5, occasionally offers responses that go beyond the provided options.

Text2SQL Prompt Template

You are a SQL generation assistant. Given several constraints, you need to generate a syntactically correct {dialect} SQL query statement to retrieve target records. In order to generate reasonable queries, you must follow the following rules:

1. Directly generate only SQL query statements, without outputting any explanation or inference information.
2. Directly use “*” to retrieve all columns.
3. Use the keyword “LIMIT” to limit the maximum number of retrieved records to {max_number}.
4. Carefully construct the where conditions for the query and use the keyword ’LIKE’ as much as possible.

Given the following table structure description, only retrieval of that table is allowed:
Table schema: {schema}

Example input:
Product category: Casual pants
Question: What material is the main material for the casual pants you would like to purchase?
Answer: Polyester fiber

Example output:
SQL Query: SELECT * From item WHERE category=’Casual pants’ AND material LIKE ’% polyester fiber%’ LIMIT {max_number};

Input:
Product category: {category}
Question: {question_1}
Answer: {answer_1}
...

SQL Query:

Table 9: The one-shot prompt template of the Text2SQL tool in ProductAgent.

Query Generation Prompt Template

You are a query generation assistant. Given the user’s purchasing demands, you need to generate a short natural language query statement (Query) to retrieve the target product. In order to generate a reasonable query, you must follow the following rules:

1. The generated query should be concise, composed of keywords, and separated by spaces.
2. The generated query should cover all of the user’s purchasing requirements.
3. Do not output any explanations or inference information, and do not use unnecessary punctuation such as quotation marks.

Product category: {category}
Question: {question_1}
Answer: {answer_1}
...

Query:

Table 10: The zero-shot prompt template of the Query Generation tool in ProductAgent.

Clarification Question Generation Prompt Template

You are a product shopping assistant that can accurately identify user demands, and you are capable of generating three multiple-choice questions for demand clarification. To help you ask valuable questions, here is a summary of statistics about {category}.
Statistics: {statistics}

1. The generated content must focus on the product category ({category}) and contribute to accurately identifying user demands.
2. It is prohibited to generate new questions that are duplicates of previous ones.
3. When constructing options, try to directly select them from the statistical data.

You need to generate new multiple-choice questions for demand clarification based on the historical Q&A. Here is the historical Q&A:

Question: {question_1}
Answer: {answer_1}
...

The generated result should strictly follow the JSON format, and no additional information should be included.
JSON format description: json_description

```json

Table 11: The zero-shot prompt template of the Clarification Question Generation tool in ProductAgent. We require LLM to synthesize clarification questions in structured json form.

**User Simulator Prompt Template**

You are in a conversation with a shopping assistant, hoping that they can help you search for suitable products. To provide reasonable answers, you must follow these rules:
1. You must answer the question accurately based on the target product.
2. Try to use the options provided for your answer, if your answer is not covered by the options, simply answer "Other".
3. Directly output the answer, do not provide any explanation or reasoning information and avoid unnecessary punctuation like quotation marks.
4. Each question's answer should be on a separate line.

Here is the target product information: {item}

Here is the official question from the assistant: {questions}

Answer the above question directly, do not repeat the question, **each answer should be on a separate line**.

Table 12: The zero-shot prompt template of the user simulator.
```

Conversation example by GPT-4 driven ProductAgent and the user simulator

A: I am a product sales assistant. Please just type in the targeted product category first, and I would search for you.

U: Canvas shoes.

A: I have search several relevant items for you: <Item$_1$>, <Item$_2$>, $\cdots$. Also, you can answer these questions so that I could recommend more suitable items for you!
Q1: When you need canvas shoes, which color do you mainly focus on?
C1: ["Light gray", "Sky blue", "Yellow", "Military Green", "Rose Red", "Other"]
Q2: Which applicable scenarios do you prefer when wearing canvas shoes?
C2: ["Travel", "Dance", "Fitness", "Construction Site", "Official Flagship", "Other"]
Q3: When choosing canvas shoes, what functions do you value?
C3: ["Anti slip", "Increase by 10CM", "Waterproof bottom", "Anti puncture", "Antibacterial", "Other"]

U: **Answers for Q1**: "Military Green".
Answers for Q2: "Travel".
Answers for Q3: "Other".

A: I have search several relevant items for you: <Item$_1$>, <Item$_2$>, $\cdots$. Also, you can answer these questions so that I could recommend more suitable items for you!
Q1: When choosing canvas shoes, which style do you prefer?
C1: ["Leisure", "Internal elevation", "Ties", "Flat Heel", "Low help", "Other"]
Q2: What upper decoration attributes do you usually pay attention to?
C2: ["Adhesive", "Breathable", "Inside", "Mesh surface", "Sole", "Other"]
Q3: When purchasing canvas shoes, which material do you prefer to make them from?
C3: ["Mesh", "Rubber", "Canvas", "Other"]

U: **Answers for Q1**: "Leisure".
Answers for Q2: Coffee leopard print, leopard print, round head.
Answers for Q3: "Canvas".

A: I have search several relevant items for you: <Item$_1$>, <Item$_2$>, $\cdots$. Also, you can answer these questions so that I could recommend more suitable items for you!
...

Ground truth product item

Title: Feiyue thick-soled canvas shoes for women, spring 2023, leopard print loafers, casual and versatile mantou shoes 8925
Category: Canvas shoes, **Brand**: ["Feiyue"], **Series**: [], **Target Customer**: ["Female", "18-40 years old", "Youth"], **Applicable Scenario**: ["Spring"], **Decorative Attribute**: ["Shoe bottom", "Thick-soled", "Cross straps", "Coffee leopard print", "Shallow mouth", "Leopard print", "Pattern", "Round head", "Khaki leopard print"], **Material**: ["Rubber", "Canvas"], **Style**: ["Lace-up", "Low-cut", "Leopard print", "Casual", "Bread", "Steamed bun", "All-match", "Vulcanization process"] **Specification**: ["EUR34", "EUR35", "EUR36", "EUR37", "EUR38", ...] **Color**: ["Purple", "Lake blue and orange", "Army green and white", "Pink", "Mixed colors"] **Function**: []

Table 13: A conversation example between ProductAgent (A) and the user (U). We also list the ground truth product item that the user is searching for.