ASK: Aspects and Retrieval based Hybrid Clarification in Task Oriented Dialogue Systems

Rishav Sahay, Lavanya Tekumalla, Purav Aggarwal, Arihant Jain, Anoop Saladi
Amazon
`rishavsahayiit@gmail.com, lavanya.tekumalla@gmail.com`
`{aggap, arihanta, saladias}@amazon.com`

Abstract

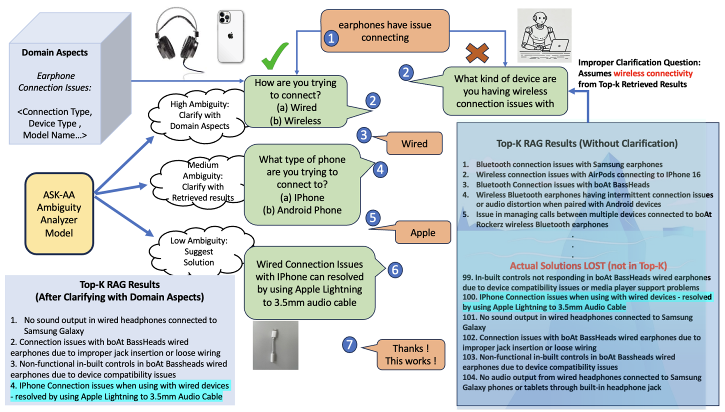
Ambiguous user queries pose a significant challenge in task-oriented dialogue systems relying on information retrieval. While Large Language Models (LLMs) have shown promise in generating clarification questions to tackle query ambiguity, they rely solely on the top-k retrieved documents for clarification which fails when ambiguity is too high to retrieve relevant documents in the first place. Traditional approaches lack principled mechanisms to determine when to use broad domain knowledge vs specific retrieved document context for clarification. We propose AsK, a novel hybrid approach that dynamically chooses between document-based or aspect-based clarification based on query ambiguity. Our approach requires no labeled ambiguity/clarification data and introduces: (1) Weakly-supervised Longformer-based ambiguity analysis, (2) Automated domain-specific aspect generation using clustering and LLMs and (3) LLM-powered clarification generation. AsK demonstrates significant improvements over baselines in both single-turn and multi-turn settings (recall@5 gain of ~20%) when evaluated on product troubleshooting and product search datasets.

1 Introduction

Ambiguity in user queries remains a fundamental challenge in task-oriented dialogue (ToD) systems relying on information retrieval (IR), where the goal is to assist users in completing specific tasks—such as retrieving product information or identifying precise troubleshooting solutions from an underlying knowledge base (KB). Users often provide incomplete information or indulge in multi-faceted queries that map to multiple distinct interpretations. For example, a query like *"earphones have issue connecting"* lacks crucial details—Is the connection wired or Bluetooth? What device is being used? What is the earphone model? Similarly, in product search, a vague query like *"camera for photography"* can map to multiple distinct needs (DSLRs, mirrorless cameras, action cameras). Without clarification, the system risks retrieving irrelevant results (Kuhn et al., 2023; Deng et al., 2023).

Recent advances in LLMs (OpenAI, 2024; Anthropic, 2025), have enhanced ToD systems, especially with the adoption of Retrieval-Augmented Generation (RAG) (Lewis et al., 2021). However, LLMs often fail to proactively seek clarification, defaulting to generic or incomplete responses, thereby shifting the burden onto users to refine their queries. Asking the right clarification questions in TOD systems remain a crucial challenge (Louvan and Magnini, 2020).

Two key aspects of this challenge are **what to ask** and **when to ask** a clarification question. Earlier approaches to *what to ask* relied on rigid and domain specific rule based and slot filling approaches (Louvan and Magnini, 2020; Ye-Yi et al., 2005; Gokhan and Renato, 2011), or information gain maximization and confusion set reduction to decide on the discriminating aspects (Sajjad et al., 2012a; Arabzadeh et al., 2022). More recent methods incorporate weakly supervised sequence-to-sequence models (Zamani et al., 2020; Feng et al., 2023) that require labeled clarification data which is impractical to collect at scale across multiple domains. The state-of-the-art in this space is LLM based retrieval-augmented clarification (Chi et al., 2024). However, reliance on top retrieved results becomes problematic when the query ambiguity is high, where the retrieved set might not capture the diverse range of potential interpretations, leading to narrow or misaligned clarification questions. To address this limitation, some works have explored using pre-curated domain aspects (Wang et al., 2014; Sircar et al., 2022) to capture the broader facets of questions in the domain. However, this can be overly broad leading to redundant questions when


Domain Aspects
Earphone Connection Issues:
<Connection Type, Device Type , Model Name...>
1 earphones have issue connecting
How are you trying to connect? (a) Wired (b) Wireless
2
High Ambiguity: Clarify with Domain Aspects
3 Wired
ASK-AA Ambiguity Analyzer Model
Medium Ambiguity: Clarify with Retrieved results
What type of phone are you trying to connect to? (a) IPhone (b) Android Phone
4
5 Apple
Low Ambiguity: Suggest Solution
Wired Connection Issues with IPhone can resolved by using Apple Lightning to 3.5mm audio cable
6
7 Thanks ! This works !
2 What kind of device are you having wireless connection issues with
Improper Clarification Question: Assumes wireless connectivity from Top-k Retrieved Results

Top-K RAG Results (After Clarifying with Domain Aspects)

1. No sound output in wired headphones connected to Samsung Galaxy
2. Connection issues with boAt BassHeads wired earphones due to improper jack insertion or loose wiring
3. Non-functional in-built controls in boAt Bassheads wired earphones due to device compatibility issues
4. IPhone Connection issues when using with wired devices - resolved by using Apple Lightning to 3.5mm Audio Cable

Top-K RAG Results (Without Clarification)

1. Bluetooth connection issues with Samsung earphones
2. Wireless connection issues with AirPods connecting to IPhone 16
3. Bluetooth Connection issues with boAt BassHeads
4. Wireless Bluetooth earphones having intermittent connection issues or audio distortion when paired with Android devices
5. Issue in managing calls between multiple devices connected to boAt Rockerz wireless Bluetooth earphones

.
.
.

Actual Solutions LOST (not in Top-K)

99. In-built controls not responding in boAt BassHeads wired earphones due to device compatibility issues or media player support problems
100. IPhone Connection issues when using with wired devices - resolved by using Apple Lightning to 3.5mm Audio Cable
101. No sound output in wired headphones connected to Samsung Galaxy
102. Connection issues with boAt BassHeads wired earphones due to improper jack insertion or loose wiring
103. Non-functional in-built controls in boAt Bassheads wired earphones due to device compatibility issues
104. No audio output from wired headphones connected to Samsung Galaxy phones or tablets through built-in headphone jack

Figure 1: The user asks the chatbot *earphones have issue connecting*. On the right side, we see a sub-optimal LLM generated clarification based on the top-k retrieved results that discuss wireless connectivity issues. However, the user has wired earphones. Illustrated on the left is our framework that leverages AsK-Ambiguity-Analyzer to ask clarification questions based on (1) domain aspects when the ambiguity is high (2) top-k retrieved results when there is relatively lower ambiguity (3) No clarification is asked and the solution is presented when there is very low ambiguity successfully resolving the user query.

the query itself has insufficient information to identify relevant solutions from the retrieved context.

This presents a key challenge: deciding when to use broad domain aspects versus retrieved document context for clarification. When query ambiguity is high, domain aspects help explore the wider solution space. Conversely, when the query scope is narrower, leveraging top-k retrieved results can lead to more precise, contextually relevant clarifications. Existing methods lack a principled mechanism to switch between these approaches. This motivated us to develop an ambiguity analyzer that can dynamically choose between these approaches based on the query characteristics.

There are some (Arabzadeh et al., 2022; Zhang and Choi, 2023; Kuhn et al., 2023; Deng et al., 2023) works on *when to ask* a clarification question or the termination criteria. Existing techniques suffer from the same constraints that they cannot look beyond the top-k results or poses too many questions based on overly broad domain aspects.

To this end, we propose AsK: a novel Clarification framework. We propose a weakly-supervised Longformer based classification model (AsK-Ambiguity-Analyzer) that addresses both *what to ask* through a hybrid approach to either use a broader set of diverse documents for clarification when ambiguity is high (*domain-clarify*) or the top-k documents when there is relatively lower ambiguity (*topk-clarify*) and *when to ask* to determine if the ambiguity is low enough that no clarification question is needed (*show-result*) (Figure 1). For actually generating the clarification questions, we rely on LLMs fed with the right context and instructions. We only assume the availability of a labelled IR dataset (mapping ambiguous queries to target documents) to evaluate our framework and train the weakly supervised AsK-Ambiguity-Analyzer. We do not assume any labelled ambiguity level or clarification data.

Summary of Contributions:

- We propose an LLM powered hybrid clarification framework, leveraging either the top-k documents or domain aspects based on query ambiguity
- We train a weakly-supervised Longformer model AsK-ambiguity-analyzer without access to labelled data, to analyze query ambiguity level.
- We propose an automated LLM based granular domain aspect generation from clusters of user queries through agglomerative clustering.
- We evaluate retrieval effectiveness and clarification quality in both single-turn and multi-turn settings for e-commerce product troubleshooting and product search datasets. Our approach results in improved retrieval accuracy (~20% recall@5 gain) and enhanced clarification quality (~2-3% questions and options relevance gain).

2 Related Work

Aspect Extraction: Prior work on product aspect extraction includes semi-supervised models such as FL-LDA and UFL-LDA (Wang et al., 2014) which extract seeding aspects from product descriptions to regroup reviews. In (Sircar et al., 2022), the authors introduce fully automated methods for clustering aspect phrases and generating human-readable names for clusters in e-commerce reviews.

Clarification Candidate Generation: Early work explored underspecified query refinement through question generation (Sajjad et al., 2012b). Studies on clarification in web and aspect-based search employ slot-filling models for weak supervision (Zamani et al., 2020) and retrieval-based aspect selection in multi-turn systems like MulClari-LLMs (Zhao and Dou, 2024). Fine-tuning approaches enhance LLMs through retrieval-aware conditioning (Chi et al., 2024) and ambiguity-driven prompting, as seen in CLAM (Kuhn et al., 2023) and ProCOT (Deng et al., 2023). A multi-attention sequence-to-sequence model has also been explored for generating user-specific clarification questions (Feng et al., 2023). Kim et al. (Kim et al., 2024) propose aligning LLMs to handle ambiguity via self-disambiguation using intrinsic knowledge. However, existing methods still lack a principled mechanism for dynamically assessing and resolving query ambiguity.

Termination Criterion: There is very little work on *when to ask* or the termination criterion for clarification. In (Arabzadeh et al., 2022) the authors analyze the coherency graph of the retrieved results, while state of the art baselines(Kuhn et al., 2023; Deng et al., 2023) have used a LLM to determine the termination criterion. However, their approach is limited either by the scope of top-k retrieved results or by reliance on inflexible, predefined aspect taxonomies, making it sub-optimal for highly ambiguous queries and leaving an opportunity for more adaptive clarification strategies.

3 Problem Definition

Given a user query q, a document set D, and a retrieval system R, let the top-k retrieved documents be denoted by: $D^{topk} = \{d_1, \ldots, d_k\}$. To determine the query's ambiguity level a, we propose *AsK-ambiguity-analyzer model* (A), that takes the query q and top-k documents D^{topk}:

$$a = A(q, D^{topk})$$

$$a \in \{show\_response, topk\_clarify, domain\_clarify\}.$$

If ambiguity is low, the system presents solutions from D^{topk}. Otherwise, it generates a *clarification question* c, using model C where $\{o_1, \ldots, o_m\}$ are the possible options to the clarification question. :

$$c, \{o_1, \ldots, o_m\} = C(q, D^{topk}),$$

To train the ambiguity analyzer A, we assume access to a groundtruth IR dataset $\mathcal{D}^{\text{Target}}$, containing query-document pairs where each query is mapped to its most relevant document in the KB post clarification:$\mathcal{D}^{\text{Target}} = \{(q_i, d_i^*)\}$ where d_i^* is ground truth document for query q_i.

4 AsK Framework

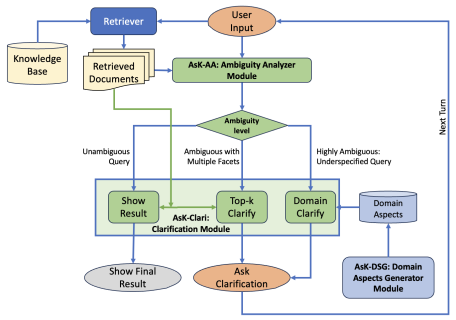


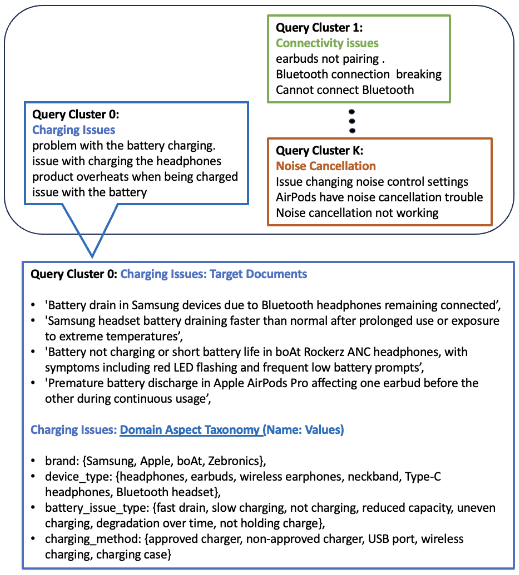
Figure 2: AsK Framework

The AsK framework is designed to retrieve the most relevant response in a ToD by analyzing query ambiguity to ask the right clarification questions. The AsK framework comprises of three main modules as shown in figure 2. (1) *Domain Aspects Generation (**AsK-DSG**)* module that clusters and categorizes query types and pre-curates domain aspects with a LLM for each query type. (2) The Longformer based weakly supervised **AsK-Ambiguity-Analyzer (AsK-AA)** model that determines if a clarification question is required (*when to ask*) and clarification strategy (*what to ask*) based on either the missing domain aspects when the query is highly ambiguous and under-specified or the top-k retrieved results for multi-faceted queries to narrow down the facet of interest. (3) The **AsK-Clarify** module that poses the actual clarification question when required. Each of these modules is described in more detail in the following subsections while the overall training, inference workflows are described in Appendix E.

4.1 Domain Aspects Generator (AsK-DSG)

In this section, we discuss generating domain aspects to ask clarification questions for highly ambiguous queries. To generate domain aspects, we leverage the training dataset $\mathcal{D}^{\text{train}}$ containing user queries and the best match target documents.

Domain aspects vary across query types. For example, in earphone-related queries, *charging type* (USB, wireless, or charging case) is crucial for battery issues, whereas *connectivity type* (wired or Bluetooth) is key for pairing issues. A single domain-level aspect set would be too broad to capture these nuances. In order to generate granular domain aspects, we first cluster user queries using


Query Cluster 1:
Connectivity issues
earbuds not pairing .
Bluetooth connection breaking
Cannot connect Bluetooth

Query Cluster 0:
Charging Issues
problem with the battery charging.
issue with charging the headphones
product overheats when being charged
issue with the battery

Query Cluster K:
Noise Cancellation
Issue changing noise control settings
AirPods have noise cancellation trouble
Noise cancellation not working

Query Cluster 0: Charging Issues: Target Documents

- 'Battery drain in Samsung devices due to Bluetooth headphones remaining connected',
- 'Samsung headset battery draining faster than normal after prolonged use or exposure to extreme temperatures',
- 'Battery not charging or short battery life in boAt Rockerz ANC headphones, with symptoms including red LED flashing and frequent low battery prompts',
- 'Premature battery discharge in Apple AirPods Pro affecting one earbud before the other during continuous usage',

Charging Issues: Domain Aspect Taxonomy (Name: Values)

- brand: {Samsung, Apple, boAt, Zebronics},
- device_type: {headphones, earbuds, wireless earphones, neckband, Type-C headphones, Bluetooth headset},
- battery_issue_type: {fast drain, slow charging, not charging, reduced capacity, uneven charging, degradation over time, not holding charge},
- charging_method: {approved charger, non-approved charger, USB port, wireless charging, charging case}

Figure 3: Example: Domain Aspects Generator

agglomerative clustering (see Appendix A). For each query-type cluster, we leverage a LLM to derive relevant domain aspects from the *target documents* corresponding to the queries in the cluster that contain more detailed information using the Prompt G.7. In Figure 3 we show examples of different query clusters and the more detailed target documents corresponding to one of the clusters along with the domain aspects derived. Note that we also collate a possible set of values of each domain aspect curated to provide answer options to the user when asking a clarification question based on the aspect.

4.2 Ambiguity Analyzer (AsK-AA)

The AsK-AA is a Longformer based weakly supervised model designed to detect the ambiguity level of user queries. We define three classes of query ambiguity:

- ***show-result:*** Queries that are clear with minimal ambiguity, not requiring clarification
- ***topk-clarify:*** Queries that are slightly ambiguous, with multiple interpretations present in the top-k retrieved results D^{topk}.
- ***domain-clarify:*** Queries that are highly underspecified and ambiguous, where broader domain aspects are needed for clarification.

Model Architecture: The AsK-AA model is a classifier model based on the Longformer (4096 context length). It's input is the user query q and its corresponding top-k retrieved documents D^{topk} and it's output is one of the three ambiguity classes <*show-result, topk-clarify, domain-clarify*>. The Longformer efficiently handles longer sequences through its attention mechanism, ensuring that the combined length of the query and top-k documents is not limited by the 512-token limit of BERT.

Deriving Weak labels: For each query q in the dataset $\mathcal{D}^{\text{train}}$, we derive signals *num-aspects(q)*: the number of domain aspects in the query using the LLM Prompt G.9 and *retrieval-rank(q)*: the rank of ground truth target document in the retrieved results with query q. Weak labels for AsK-AA are defined based on thresholds for signals *num-aspects(q)* and *retrieval-rank(q)*. *show-result* is characterized by a high *num-aspects(q)* and a low *retrieval-rank(q)*. *domain-clarify* is characterized by a low *num-aspects(q)* and a high *retrieval-rank(q)*. *topk-clarify* falls between these extremes. The actual thresholds are decided automatically as described in the Appendix B. Finally, the dataset $\mathcal{D}^{\text{train}}$ is used to train the model that takes the query q and its retrieved D^{topk} documents as inputs.

4.3 Clarification Generation (AsK-Clarify)

Our clarification generation module follows two strategies: (1) Domain Aspects-Based Clarification (*domain-clarify*), used for highly ambiguous queries, leveraging a pre-curated set of domain aspects and answer options. (2) Top-k Documents-Based Clarification (*topk-clarify*), used for lower ambiguity, where multiple facets can be disambiguated from the retrieved top-k documents.

Based on this intuition, we propose three variants for generating clarification questions:

- AsK-Clarify-Soft-Routing (AsK-SR), where a single prompt includes the top-k documents, domain aspects, and ambiguity level, allowing the LLM to decide what to ask (G.5).
- AsK-Clarify-Combined (AsK-CM), where the LLM receives both sources but without explicit ambiguity classification (G.6).
- AsK-Clarify-Hard-Routing (AsK-HR), that explicitly selects either domain aspects or top-k documents from ambiguity level (G.1, G.3).

5 Experiments, Data and Results

In this section, we describe the evaluation process of the AsK framework. We describe our datasets and experimental setup in 5.1. To showcase the effectiveness of AsK, we first evaluate the AsK-AA (section 5.2) and then evaluate the AsK-Clarify in a single turn (section 5.3) and multi-turn (section 5.4) settings.

Method	PT				PS			
	SR-F1	TC-F1	DC-F1	W-F1	SR-F1	TC-F1	DC-F1	W-F1
llm-zs	-	-	-	-	60.77	51.16	13.95	48.05
llm-zs_cot	-1.19	-1.95	-6.34	-3.0	65.71	48.27	9.75	48.35
llm-icl_cot	+2.26	+17.84	+33.0	+16.83	75.13	55.14	24.48	57.9
AsK-AA	**+9.87**	**+35.05**	**+78.47**	**+39.1**	84.04	70.50	**78.26**	**77.98**
wo C_{ac}	+4.56	+26.49	+68.9	+31.39	**84.49**	**71.53**	75.00	77.91
wo R_{og}	-5.28	+15.84	+60.13	+21.59	70.65	52.69	8.89	51.96

Table 1: Results for Ambiguity Detection

Domain	#Trn.Q	#Test.CQ	#Test.AA	#Docs
PT	19433	1555	500	2858
PS	11432	2100	500	3454

Table 2: Dataset Details - Trn.Q: Training Queries, Test.CQ: AsK-Clarify test Queries, Test.AA: AsK-AA test Queries, Docs: Unique docs for retrieval

5.1 Datasets And Experimental Setup

We evaluate our approach on two large scale e-commerce datasets: (1) A proprietary **Product Troubleshooting (PT)** dataset: Historical chat transcripts between customers and troubleshooting agents are used and $\mathcal{D}^{\text{Target}}$ is derived as pairs of initial customer queries and the specific final solution from the KB identified through the course of the conversation. Note that due to confidentiality in the PT domain, we present relative improvements rather than absolute numbers. (2) Publicly available **Product Search (PS)** dataset: We leverage the ESCI dataset for headphones, cellular phones, and speakers. $\mathcal{D}^{\text{Target}}$ is derived as pairs of noisy user search queries to relevant product details on the Amazon page that the user finally interacted with.

The obtained $\mathcal{D}^{\text{Target}}$ for each dataset is divided into a $\mathcal{D}^{\text{train}}$ and a $\mathcal{D}^{\text{test}}$ set. $\mathcal{D}^{\text{train}}$ is used to generate domain aspect taxonomy and training the ambiguity analyzer. $\mathcal{D}^{\text{test}}$ is used to evaluate the quality of the clarification and the accuracy of the ambiguity analyzer. (Dataset size details in Table 2).

Experimental Setup: We leverage *claude-3.5-sonnet* LLM for tasks such as domain aspects generation and clarification question generation, and use *cohere.embed-multilingual-v3* (Cohere, 2023) as the text encoder with cosine-similarity based matching. For training AsK-AA, we used a 4096 context length Longformer model trained for 15 epochs with a batch size of 8 and a dropout rate of 0.3 (to avoid overfitting to noisy weak labels).

5.2 Evaluating AsK-Ambiguity-Analyzer

We measure classification accuracy for AsK-AA using two metrics: the class-level F1 scores (SR-F1 for *show-result*, TC-F1 for *topk-clarify*, and DC-F1 for *domain-clarify*) and the weighted F1 (W-F1) across all three classes.

Baselines: We compared the AsK-AA with several LLM-based ambiguity classification baselines: *llm-zs* (zero-shot prompting), *llm-zs_cot* (zero-shot chain-of-thought prompting), and *llm-icl_cot* (in-context examples with chain-of-thought prompting). We conducted ablation studies on our proposed weakly supervised approach (*AsK-AA*) to assess the impact of key signals used during weak labeling. Specifically, we examined the effect of removing the aspect count ($wo\ C_{ac}$) and the rank of the original document ($wo\ R_{og}$). For the $wo\ C_{ac}$ setting, we relied solely on thresholds for *retrieval-rank(q)* to weakly label the training data, omitting the aspect count signal. Conversely, in the $wo\ R_{og}$ setting, we used thresholds on *num-aspects(q)* while ignoring the document rank signal. The results of these ablations are presented in Table 1, providing insights into the contribution of these signals to the labeling process.

5.3 AsK Framework: Single-Turn Evaluation

In the single-turn setting, an ambiguous test query is fed to the system. When the system generates a clarification question, an LLM user simulator (Appendix C) provides an answer. Evaluation metrics include Recall@5 (R@5), Mean Reciprocal Rank (MRR), Mean Rank Gain (RG) of the target document retrieved after clarification, and relevance scores of the clarification question (QR) and options (OR) measured using an LLM (Details in Appendix D, Alg 3).

Baselines: We compare single-turn AsK with Query-Ref (Sajjad et al., 2012b), a max-entropy classifier using top-k documents; CLAM (Kuhn et al., 2023), which learns when to ask and generates questions via few-shot prompting; MulClari-LLMs (Zhao and Dou, 2024), an LLM-based multi-turn clarification model; and ProCOT (Chi et al., 2024), which detects ambiguity from top-k documents and generates questions using few-shot COT

Method	PT					PS				
	$R@5$	MRR	RG	QR	OR	$R@5$	MRR	RG	QR	OR
Query-Ref	-	-	-	-	-	28.03	0.220	10.51	94.00	85.32
CLAM	+2.47	+0.01	+1.81	+1.59	+1.48	30.21	0.235	11.95	95.93	87.23
MulClari-LLMs	+6.9	+0.05	+7.96	-1.4	+0.0	37.20	0.289	19.67	94.43	86.36
ProCOT	+10.75	+0.08	+11.09	+1.85	-0.85	38.65	0.294	18.94	97.03	88.06
AsK-SR	+11.01	+0.08	+31.51	+2.85	+2.15	37.99	0.291	29.80	98.0	88.2
AsK-CM	+11.01	+0.08	+37.94	+3.2	+2.55	40.02	0.308	30.51	98.36	88.5
AsK-HR	**+12.88**	**+0.09**	**+40.78**	**+3.54**	+3.05	**40.24**	0.308	**37.86**	**99.93**	**89.23**
wo top-k	+9.08	+0.06	+34.17	+2.1	**+4.6**	40.11	**0.31**	37.76	96.0	89.01
wo aspects	+9.98	+0.07	+33.32	+1.34	-0.77	38.13	0.292	20.44	97.11	87.12

Table 3: Single-turn evaluation of various clarification methods.

prompting. See results in table 3

5.4 AsK Framework: Multi-Turn Evaluation

In the multi-turn evaluation, we use the AsK-AA to determine *when to ask*, in addition to the AsK-Clarify, over a conversation lasting up to 4 turns. We report the change in the Recall@K ($\Delta R@5$) with respect to the initial retrieval numbers at the end of the conversation and the mean number of conversational turns (MT). (See Alg. 4 for details)

Baselines: We compare performance in multi-turn versions of the best performing variant of single-turn ASK (AsK-HR) with the best performing single-turn baseline ProCOT in Figure 4.

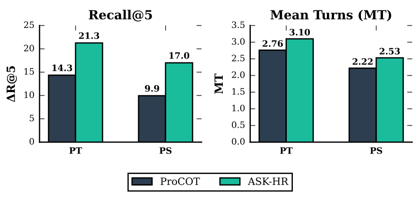


Figure 4: Multi-turn performance evaluation of ProCOT and AsK-HR for the PT and PS domains.

5.5 Discussion of Results

Ambiguity Analyzer Results: Table 1 summarizes the ambiguity detection performance across various methods. The results indicate that out-of-the-box LLMs with CoT/ICL prompting are less accurate, while fine-tuning models on *weakly labelled domain-specific data* yields better F1-scores for both the PT and PS domains. Ablation studies further highlight the importance of including both the aspects count (C_{ac}) and the rank (R_{og}) when creating the weak labels.

Clarification Quality: Table 3 summarizes the clarification quality across different methods in a single-turn setting. The results show that the AsK variants, led by AsK-HR, achieve higher rank gain(RG) and recall@5, demonstrating the advantage of *using domain aspects for clarification in high ambiguity scenarios*. AsK-HR also outperforms other baselines in terms of R@5 and MRR, indicating that *dynamically routing to either top-k clarification or domain aspects clarification boosts retrieval accuracy*. An ablation study using only the aspects (*wo top-k*) and only using the top-k (*wo aspects*) led to lower retrieval scores. The quality of clarification question (QR and SR) is also better in case of AsK and its variants.

In a multi-turn setting, we integrate ProCOT and AsK-HR with ToDs in the PT and PS domains. As shown in Figure 4, AsK-HR achieves a greater improvement in $\Delta R@5$ while maintaining a comparable number of MT, highlighting its effectiveness within ToDs. We observe ProCOT often terminates prematurely due to inaccurate termination criteria, resulting in insufficient clarification of user queries.

6 Industrial Impact

The AsK framework was integrated into a large-scale e-commerce troubleshooting chatbot, improving ambiguity resolution with a curated knowledge base. It increased self-troubleshooting adoption by 35%, reduced manual CS contacts by 12.7%, and lowered return rates in a 4-week A/B test across six product categories.

7 Conclusion

We introduced **AsK**, an LLM-powered clarification framework that dynamically selects between domain aspects and top-k documents for clarification. Our approach employs a Longformer-based ambiguity analyzer to determine when and what to ask, without labeled clarification data. Evaluations on product search and troubleshooting datasets show significant improvements in ambiguity resolution, retrieval accuracy, and clarification quality over baselines. We envision our framework serving as a foundation for future explorations into hybrid reasoning and clarification strategies.

Ethical Considerations

This research introduces **AsK**, a novel hybrid framework aimed at improving ambiguity resolution in task-oriented dialogue systems. Our goal is to enhance the efficacy and user experience of these systems. We've used anonymized datasets for this work, ensuring no personally identifiable information was involved, and importantly, no human subject data was collected or used. We carefully control the Large Language Models (LLMs) employed, evaluating generated clarifications for factual consistency to minimize hallucinations or user confusion.

We acknowledge the inherent biases that LLMs and retrieval systems may carry from their training data. While AsK's use of weak supervision techniques and automated aspect generation reduces reliance on manual annotations, making it more scalable, we still encourage future research to thoroughly explore fairness, transparency, and user safety in clarification question generation. AsK is intended as a research contribution to advance human-AI interaction in information-seeking tasks and is not designed for immediate deployment in high-stakes or safety-critical domains without further safeguards.

References

Anthropic. 2025. Claude - anthropic. `https://www.anthropic.com/claude/sonnet`. Feb 2025.

Negar Arabzadeh, Mahsa Seifikar, and Charles L.A. Clarke. 2022. Unsupervised question clarity prediction through retrieved item coherency. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 3811–3816, New York, NY, USA. Association for Computing Machinery.

Yizhou Chi, Jessy Lin, Kevin Lin, and Dan Klein. 2024. Clarinet: Augmenting language models to ask clarification questions for retrieval. *Preprint*, arXiv:2405.15784.

Cohere. 2023. cohere-embed-multi.

Yang Deng, Lizi Liao, Liang Chen, Hongru Wang, Wenqiang Lei, and Tat-Seng Chua. 2023. Prompting and evaluating large language models for proactive dialogues: Clarification, target-guided, and non-collaboration. *Preprint*, arXiv:2305.13626.

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

Tur Gokhan and De Mori Renato. 2011. Spoken language understanding: Systems for extracting semantic information from speech.

Hyuhng Joon Kim, Youna Kim, Cheonbok Park, Junyeob Kim, Choonghyun Park, Kang Min Yoo, Sang goo Lee, and Taeuk Kim. 2024. Aligning language models to explicitly handle ambiguity. *Preprint*, arXiv:2404.11972.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Clam: Selective clarification for ambiguous questions with generative language models. *Preprint*, arXiv:2212.07769.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 480–496, Barcelona, Spain (Online). International Committee on Computational Linguistics.

OpenAI. 2024. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Hassan Sajjad, Patrick Pantel, and Michael Gamon. 2012a. Underspecified query refinement via natural language question generation. ACL/SIGPARSE.

Hassan Sajjad, Patrick Pantel, and Michael Gamon. 2012b. Underspecified query refinement via natural language question generation. In *Proceedings of COLING 2012*, pages 2341–2356, Mumbai, India. The COLING 2012 Organizing Committee.

Prateek Sircar, Aniket Chakrabarti, Deepak Gupta, and Anirban Majumder. 2022. Distantly supervised aspect clustering and naming for e-commerce reviews. In *NAACL 2022*.

Tao Wang, Yi Cai, Ho fung Leung, Raymond Y.K. Lau, Qing Li, and Huaqing Min. 2014. Product aspect extraction supervised with online domain knowledge. *Knowledge-Based Systems*, 71:86–100.

Wang Ye-Yi, Deng Li, and Acero Alex. 2005. Spoken language understanding.

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*, WWW '20, page 418–428, New York, NY, USA. Association for Computing Machinery.

Michael J. Q. Zhang and Eunsol Choi. 2023. Clarify when necessary: Resolving ambiguity through interaction with lms. *Preprint*, arXiv:2311.09469.

Ziliang Zhao and Zhicheng Dou. 2024. Generating multi-turn clarification for web information seeking. In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 1539–1548, New York, NY, USA. Association for Computing Machinery.

A AsK-DSG - Clustering Details

To generate granular domain aspects, we cluster queries using agglomerative clustering, which does not require specifying the number of clusters beforehand. Instead, we define a distance threshold (3 for PT and 4 for PS) and employ Ward's linkage for clustering. Query embeddings are obtained using the Cohere embeddings model.

B Weakly Supervised Model Training of ASK-Ambiguity-Amalyzer

The AsK-AA model is a classifier based on the Longformer, designed to handle longer sequences by efficiently managing its attention mechanism. The model takes as input a user query q along with its corresponding top-k retrieved documents D^{topk}, and outputs one of the three ambiguity classes: *show-result*, *topk-clarify*, or *domain-clarify*.

To automate threshold selection for the ambiguity classes, we proceed as follows:

- We randomly select a validation set of 300 samples from $\mathcal{D}^{\text{Target}}$. Each sample consists of the query q, the top-k documents D^{topk}, and the associated features *num_aspects(q)* and *retrieval_rank(q)*.

- These samples are manually labeled with the target ambiguity classes corresponding to the three categories.

- A Decision Tree Classifier is then trained using only the features *num_aspects(q)* and *retrieval_rank(q)* to predict the ambiguity level. To determine the optimal parameters for training of the decision tree, we perform a cross-validation grid search to tune hyperparameters such as *max_depth*, *min_samples_leaf*, and *min_samples_split*.

- Once the optimal hyperparameters are identified, the trained classifier is used to weakly label the remaining queries in $\mathcal{D}^{\text{train}}$, thereby creating the weakly labelled training dataset.

Finally, $\mathcal{D}^{\text{train}}$ is used to train the AsK-AA model. The Longformer-based classifier processes the query q and its retrieved documents D^{topk}, outputting one of the three ambiguity classes. This approach efficiently integrates both query and document information while overcoming the 512-token limit inherent in models such as BERT.

C Clarification Evaluation

To evaluate the effectiveness of clarification questions, we employ an LLM-based user simulator. Given an ambiguous user query q, a clarification question cq, and a set of answer options $\{o_1, \ldots, o_m\}$, the user simulator (Prompt G.2) selects the most appropriate option. We refer to this simulator as the Answer Generator A_{gen}, which determines the selected answer ans based on the original document d_{orig}:

$$ans = A_{gen}(cq, \{o_1, \ldots, o_m\}, d_{orig}) \qquad (1)$$

In case of a single turn conversation, the answer generator selects one of the options for the given question, and the refined query is used for the next retrieval. The refined query is obtained by simply appending the ans to the q.

D Clarification Relevance Calculation

To assess the relevance of clarification questions generated by various methods, we leverage a large language model (LLM) using an in-context learning with chain-of-thought (*icl-cot*) approach. We define a set of objective metrics, each scored on a scale from 1 to 5, to evaluate both the questions and their associated options.

D.1 Metrics for Question Relevance

We consider the following metrics for evaluating the relevance of clarification questions:

1. **Question Redundancy** ($q_{\textbf{red}}$): Evaluates whether the question repeats information already present in the user query instead of providing new clarification.

2. **Question Simplicity** ($q_{\textbf{sim}}$): Assesses whether the question is simple and focused on a single aspect rather than addressing multiple aspects or being overly descriptive.

3. **Question Relevance** ($q_{\textbf{rel}}$): Determines how well the question targets the optimal clarification needed.

D.2 Metrics for Options Relevance

Similarly, the relevance of the options presented alongside the questions is evaluated using the following metrics:

1. **Options Simplicity** (o_{sim}): Checks whether the options are directly related to the question and remain simple.

2. **Options Independence** (o_{ind}): Measures the degree of independence among the options, ensuring they do not overlap excessively.

Refer to Prompt G.8 for the detailed instructions used to evaluate these metrics.

D.3 Aggregate Relevance Scores

Once the metrics have been scored over a set of n_{samples} samples, we compute the overall Question Relevance (QR) and Options Relevance (OR) as follows:

$$QR = \frac{q_{\text{red}} + q_{\text{sim}} + q_{\text{rel}}}{3 \times 5 \times n_{\text{samples}}} \times 100 \qquad (2)$$

$$OR = \frac{o_{\text{sim}} + o_{\text{ind}}}{2 \times 5 \times n_{\text{samples}}} \times 100 \qquad (3)$$

These formulas yield a percentage score indicating the average performance of the questions and options with respect to their defined metrics. A higher score represents better performance in terms of clarity, simplicity, and relevance.

E ASK Framework - Dive Deep

In this section, we provide a summary of notation for the paper followed by detailed algorithms for training, inference and evaluation in the ASK framework.

Symbol	Description
q, q_t	User query (at turn t)
q'	Refined query after clarification
d, d^*	Document; target (gold) document
D	Entire document corpus
D^{topk}, D_t^{topk}	Top-k retrieved docs (at turn t)
$\mathcal{D}^{\text{target}}$	Full IR dataset with gold documents
$\mathcal{D}^{\text{train}}$	Training subset of $\mathcal{D}^{\text{target}}$
$\mathcal{D}^{\text{test}}$	Test subset for evaluation
$R(q, D)$	Retrieval function over corpus
$A(q, D^{\text{topk}})$	Ambiguity classifier (AsK-AA)
$C(q, D^{\text{topk}})$	Clarification from retrieved docs
$C(q, \mathcal{A}_j)$	Clarification from domain aspects
a, a_t	Ambiguity label at turn t
a_t^{sim}	Simulated answer to clarification
c, c_t	Clarification question (at turn t)
$\{o_1, \dots, o_m\}$	Options for clarification question
$\mathcal{C}_j$	Query cluster / type
$\mathcal{A}_j$	Aspect set (with values) for cluster $\mathcal{C}_j$
$\mathcal{A}_{\text{gen}}$	LLM-based answer simulator
T	Max allowed clarification turns
AsK-AA	Ambiguity analyzer module $A(q, D^{\text{topk}})$
AsK-DSG	Domain aspect generator for $\mathcal{C}_j \to \mathcal{A}_j$
AsK-Clarify	Clarification generator
show_result	Ambiguity class: show D^{topk} directly
topk_clarify	Ambiguity class: clarify using D^{topk}
domain_clarify	Ambiguity class: clarify using $\mathcal{A}_j$
AsK-HR	Hard routing strategy for clarification
AsK-CM	Combined (unrouted) clarification
AsK-SR	Soft routing with blended sources

Table 4: Summary of notation and module names in the ASK framework.

Algorithm 1 Training Phase of ASK Framework

```
Require: Labeled IR dataset $\mathcal{D}^{\text{target}} = \{(q_i, d_i^*)\}$
 1: Split into train and test sets:
 2:     $\mathcal{D}^{\text{target}} = \mathcal{D}^{\text{train}} \cup \mathcal{D}^{\text{test}}$
            ▷ Train Ambiguity Analyzer (AsK-AA)
 3: for all $(q_i, d_i^*) \in \mathcal{D}^{\text{train}}$ do
 4:     Retrieve $D_i^{\text{topk}} \leftarrow R(q_i, D)$
 5:     Compute signals: num_aspects($q_i$), retrieval_rank($q_i$)
 6:     Derive weak label $a_i \in$ {show_result, topk_clarify, domain_clarify}
 7: end for
 8: Train classifier $A(q, D^{\text{topk}})$ on weakly labeled $\mathcal{D}^{\text{train}}$
            ▷ Generate Domain Aspects (AsK-DSG)
 9: Cluster queries in $\mathcal{D}^{\text{train}}$ into types $\{\mathcal{C}_j\}$
10: for all cluster $\mathcal{C}_j$ do
11:     Retrieve associated documents $\{d_1, \dots, d_n\}$
12:     Generate aspects and values: $\mathcal{A}_j \leftarrow$ LLM($\{d_1, \dots, d_n\}$)
13: end for
14: Store aspect taxonomy: $\mathcal{C}_j \to \mathcal{A}_j$
```

Algorithm 2 Inference Phase of AsK Framework (Multi-Turn)

```
Require: Initial query $q_0$, document set $D$, ambiguity analyzer $A$, aspect taxonomy $\{\mathcal{C}_j \to \mathcal{A}_j\}$, max turns $T$
 1: for $t = 0$ to $T - 1$ do
 2:     Retrieve top-k documents: $D_t^{\text{topk}} \leftarrow R(q_t, D)$
 3:     Predict ambiguity: $a_t \leftarrow A(q_t, D_t^{\text{topk}})$
 4:     if $a_t =$ show_result then
 5:         return Final answer from $D_t^{\text{topk}}$
 6:     else if $a_t =$ topk_clarify then
 7:         Generate clarification: $(c_t, \{o_1, \ldots, o_m\}) \leftarrow C(q_t, D_t^{\text{topk}})$
 8:     else if $a_t =$ domain_clarify then
 9:         Identify query cluster $\mathcal{C}_j$ and retrieve $\mathcal{A}_j$
10:         Generate clarification: $(c_t, \{o_1, \ldots, o_m\}) \leftarrow C(q_t, \mathcal{A}_j)$
11:     end if
12:     Get user answer $a_t^{\text{user}}$ (or simulate in evaluation)
13:     Refine query: $q_{t+1} \leftarrow q_t + a_t^{\text{user}}$
14: end for
15: return Final answer from last $D_T^{\text{topk}}$
```

Algorithm 3 Evaluation Phase (Single-Turn)

```
Require: Test dataset $\mathcal{D}^{\text{test}} = \{(q_i, d_i^*)\}$, answer simulator $\mathcal{A}_{\text{gen}}$
 1: for all $(q_i, d_i^*) \in \mathcal{D}^{\text{test}}$ do
 2:     Retrieve $D^{\text{topk}} \leftarrow R(q_i, D)$
 3:     Predict ambiguity $a \leftarrow A(q_i, D^{\text{topk}})$
 4:     if $a =$ show_result then
 5:         Use $D^{\text{topk}}$ for evaluation
 6:     else if $a =$ topk_clarify then
 7:         Generate $(c, \{o_i\}) \leftarrow C(q_i, D^{\text{topk}})$
 8:         Simulate answer $a_{\text{sim}} \leftarrow \mathcal{A}_{\text{gen}}(c, \{o_i\}, d_i^*)$
 9:         Refine query $q' \leftarrow q_i + a_{\text{sim}}$
10:         Retrieve $D'^{\text{topk}} \leftarrow R(q', D)$
11:     else if $a =$ domain_clarify then
12:         Identify aspects $\mathcal{A}_j$, generate $(c, \{o_i\})$
13:         Simulate answer $a_{\text{sim}} \leftarrow \mathcal{A}_{\text{gen}}(c, \{o_i\}, d_i^*)$
14:         Refine query $q' \leftarrow q_i + a_{\text{sim}}$
15:         Retrieve $D'^{\text{topk}} \leftarrow R(q', D)$
16:     end if
17:     Compute metrics: Recall@k, MRR, Rank Gain, QR, OR
18: end for
```

Algorithm 4 Evaluation Phase (Multi-Turn)

```
Require: Test dataset $\mathcal{D}^{\text{test}} = \{(q_i, d_i^*)\}$, answer simulator $\mathcal{A}_{\text{gen}}$, max turns $T$
 1: for all $(q_i, d_i^*) \in \mathcal{D}^{\text{test}}$ do
 2:     Initialize $q_0 \leftarrow q_i$
 3:     for $t = 0$ to $T - 1$ do
 4:         Retrieve $D_t^{\text{topk}} \leftarrow R(q_t, D)$
 5:         Predict $a_t \leftarrow A(q_t, D_t^{\text{topk}})$
 6:         if $a_t =$ show_result then
 7:             return Evaluation on $D_t^{\text{topk}}$
 8:             break
 9:         else if $a_t =$ topk_clarify then
10:             Generate $(c_t, \{o_i\}) \leftarrow C(q_t, D_t^{\text{topk}})$
11:         else if $a_t =$ domain_clarify then
12:             Generate $(c_t, \{o_i\}) \leftarrow C(q_t, \mathcal{A}_j)$
13:         end if
14:         Simulate answer: $a_t^{\text{sim}} \leftarrow \mathcal{A}_{\text{gen}}(c_t, \{o_i\}, d_i^*)$
15:         Update query: $q_{t+1} \leftarrow q_t + a_t^{\text{sim}}$
16:     end for
17:     Retrieve final $D_T^{\text{topk}} \leftarrow R(q_T, D)$
18:     Compute metrics: Final Recall@k, MRR, R@k, Mean Turns
19: end for
```

F Error Analysis

To better understand the limitations of the AsK framework, we conduct a qualitative error analysis and identify three recurring patterns that impact system performance: errors in aspect selection, challenges arising from subtle document variations, and multi-turn drift.

- **Aspect Selection Errors:** In some cases, the ambiguity analyzer correctly routed the query to the *domain-clarify* mode. However, the LLM occasionally selected suboptimal aspects for clarification, often due to limited grounding in domain-specific nuances or incomplete world knowledge. As a result, the system initially asked less relevant clarification questions before eventually arriving at the right aspect. While this still led to successful disambiguation, it introduced additional conversational turns and a slight delay in resolution.

- **Fine-Grained Document Variants in the Knowledge Base:** In domains like troubleshooting, the knowledge base often contains several near-duplicate documents differing only by fine-grained product variations (e.g., different models of the same smartphone brand). When the top-k retrieved set includes documents that are close but not an exact match, the ambiguity analyzer may incorrectly assume low ambiguity, leading to a premature resolution attempt. This is particularly problematic when the actual ground truth document is just outside the top-k, resulting in misclassification and degraded retrieval accuracy.

- **Multi-Turn Accumulated Drift:** In multi-turn interactions, early-stage misclassifications by the ambiguity analyzer can have cascading effects. For example, if the analyzer incorrectly invokes a *topk-clarify* path when domain-level clarification is needed, the system may ask unnecessary or tangential questions. These irrelevant clarifications can lead to a misaligned user context and ultimately retrieval of incorrect documents, even after multiple turns. Such drift underscores the need for better robustness and correction mechanisms across turns.

These observations highlight the critical role of accurate ambiguity classification and precise aspect grounding. Future improvements could focus on incorporating domain specific aspect importance weights, more robust aspect disambiguation strategies, and confidence-aware decision mechanisms in the ambiguity analyzer to reduce conversational detours and enhance retrieval fidelity.

G Prompts

Prompt G.1: ASK-Aspects Based Clarification

Instruction:
You are tasked at generating a clarification question for an ambiguous customer query.
You are provided as input the following:
1) Conversation: This is the conversation between the user and the assistant. This is enclosed within the XML tags <conversation>.
2) Aspects: These are the aspects (with descriptions and values) that are relevant to the user query, and will help in framing the right clarification question. This will be enclosed within <aspects> XML tags.
3) Top-K: These are the top-k documents retrieved for the ambiguous user query. This is enclosed within the XML tags <top_k_docs>.

Task Related Instructions:
- Select one aspect from <aspects> that should lead to most reduction in the ambiguity within the top-k documents, and hence disambiguiating the query.
- Use the selected aspect to frame a valid question. Provide exhaustive and relevant options from <values> associated with the aspect.
- Your clarification response should be enclosed within the <response> XML tags.
- Enclose the question within <question> and the option should be enclosed within <option1>, <option2> etc, followed by *none of these* option.
- Make sure that the clarification question does not clarify something that is already part of the conversation.
- Before generating the response, you will state your reasoning of the aspect selection within <thinking>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<conversation> {conversation} </conversation>
<aspects> {aspects_taxonomy} </aspects>
<top_k_docs> {top_docs} </top_k_docs>

Prompt G.2: Answer Generator

Instruction:
Your are a customer whose task is to answer a clarification question. You should answer the clarification question by selecting one of the options from the question.
You are given as input the following:
1. Oracle Document: This is the actual document basis which you will answer the question. This is enclosed within the XML tags <oracle>.
2. Question: This is the clarification question asked to you. This is enclosed within the XML tags <question>. The clarification question is provided with options each within <options>.

Instructions:
1. Your answer will always be in the form of an option. You will just output the most appropriate / closest option within <answer>, basis the oracle document.
2. Never output answer in the form of text. Always output the option index.
3. If none of the options is valid as per the oracle, you can select none.
4. Before answering the question, reason in brief within <thinking> XML tags.

Output Format:
<thinking>[Brief Reason Here]</thinking>
<answer>1</answer>
- Always output 1 most relevant answer. Never output more than one options for a clarification question.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<oracle> {oracle_doc} </oracle>
<question> {question} </question>

Prompt G.3: ASK-Top-K Based Clarification

Instruction:
Given the user query and retrieved documents, ask a valid clarification question. If the query is ambiguous, select the key information from the retrieved documents that is relevant to the query. Then, ask a clarifying question based on the selected key information. Your clarifying question should always contain some options in the format of '1. option1, 2. option2...' accompanied with *none of these* option.
- Firstly analyze the query within the <thinking> XML tags.
- Then enclose your subsequent responses within <response> XML tags.
- Output a clarification question within the XML tags <question>. The options should be enclosed within <option1>, <option2> etc. XML tags.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<conversation> {conversation} </conversation>
<top_k_docs> {top_docs} </top_k_docs>

Prompt G.4: Query Aspects Prompt

Instruction:
You are provided the user query and a taxonomy of aspects related to the domain of the query.
Your task is to tell what aspects present within the taxonomy is contained in the query. The query is enclosed within the <query> XML tags, while the taxonomy is enclosed within <taxonomy>. The taxonomy is a dictionary with keys as aspects and values as the aspects' description and values it can take up.
- Analyze the user query and output the aspect names (keys in the dict) that are explicitly (clearly) present in the user query without ambiguity.
- Output the name of the aspects within the XML tags <output>.
- Each aspect should be separated with commas ",". Before generating the aspects, provide your reasoning within the XML tags <thinking>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<query> {query} </query>
<taxonomy> {taxonomy} </taxonomy>

Prompt G.5: ASK Soft Routing Prompt

Instruction:
You are tasked at generating a clarification question for an ambiguous customer query.
You are provided as input the following: 1) Conversation: Conversation: This is the conversation between the user and the assistant. This is enclosed within the XML tags <conversation>.
2) Aspects: These are the aspects (with descriptions and values) that are relevant to the user query, and will help in framing the right clarification question. This will be enclosed within <aspects> XML tags.
3) Top-K: These are the top-k documents retrieved for the ambiguous user query. This is enclosed within the XML tags <top_k_docs>.
4) Clarification Type: This is the type of clarification you need to perform. This is enclosed within the XML tags <clarify_type>.

There are two types of clarification:
1. top_k_clarify: This clarification type is done when the provided query is somewhat ambiguous and the top-k documents holds some relevant to the query. In this clarification type, you will refer to the top-k documents to form the clarification questions.
2. domain_clarify: This clarification type is done when the provided query is highly ambiguous, rendering the top-k documents not very relevant and coherent. In this clarification type, you will refer to the defined aspects to ask the clarification question.

- Leverage either the top-k documents or the provided aspects to clarify, basis the provided clarification type.
- Enclose the question within <question> and the option should be enclosed within <option1>, <option2> etc, followed by *none of these* option.
- Provide exhaustive options to the customer to select from.
- Before generating the response, you will state your reasoning of the aspect selection within <thinking>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<conversation> {conversation} </conversation>
<aspects> {aspects_taxonomy} </aspects>
<top_k_docs> {top_docs} </top_k_docs>
<clarify_type> {clarify_type} </clarify_type>

Prompt G.6: ASK Combined Prompt

Instruction:
You are tasked at generating a clarification question for an ambiguous customer query.
You are provided as input the following:
1) Conversation: This is the conversation between the user and the assistant. This is enclosed within the XML tags <conversation>.
2) Aspects: These are the aspects (with descriptions and values) that are relevant to the user query, and will help in framing the right clarification question. This will be enclosed within <aspects> XML tags.
3) Top-K: These are the top-k documents retrieved for the ambiguous user query. This is enclosed within the XML tags <top_k_docs>.

- Provided to you context in the form of the top-k documents and the aspects related to the domain, generate a clarification question.
- Enclose the question within <question> and the option should be enclosed within <option1>, <option2> etc, followed by *none of these* option.
- Provide exhaustive options to the customer to select from.
- Before generating the response, you will state your reasoning of the aspect selection within <thinking>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<conversation> {conversation} </conversation>
<aspects> {aspects_taxonomy} </aspects>
<top_k_docs> {top_docs} </top_k_docs>

Prompt G.7: Aspects Taxonomy Generation

Instruction:
You are an aspects to values taxonomy generator for a given domain of documents. You are provided as input a list of documents of a specific type related to the domain. This will be enclosed with <documents>. Your task is to generate a taxonomy in the form of aspects mapped to its possible values as mentioned in the documents.

Instructions: 1. You will identify all the specific aspects in the documents. Note that these aspects should be asked as clarification questions to the customer for clarifying their queries who will be looking for these documents.
2. Note that for clarification, you will need to clarify aspects related to the product (brand, model_type etc) [PRODUCT ATTRIBUTES] or aspects related to user queries related to the product [QUERY ATTRIBUTES].
3. You will identify all the possible values of the aspects as seen in the issues. If the list of aspect values seems incomplete, use your world knowledge to complete the list.
4. You will generate each aspect within <aspect> and its values within <values>. Also provide a description regarding the aspect.
5. Provide your reasoning within <thinking> before generating the aspects. 6. Your actual response should be enclosed within <response>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<documents> {domain_documents} </documents>

Prompt G.8: Clarification Relevance Prompt

Instruction:
Given to you an ambiguous customer query and a clarification question asked by an AI assistant. Your task is to score the quality of the clarification question for the ambiguous query over a set of aspects in a scale of (1-5).

The inputs to you will be the following:
1. Query: This is the ambiguous customer query within the XML tags <query>.
2. Question: This is the clarification question asked by the AI assistant to the query, within the XML tags <question>.

You will need to score the question over these aspects:

1. Question Redundancy: Is the question asking something that is already present in the user query, and not clarifying something new ?
2. Question Simplicity: Is the question simple enough - i.e. it asks about a single aspect, rather than clarifying multiple aspects or asking a descriptive question ?
3. Question Relevance: Is the question most relevant to ask in order for most optimal clarification ?
4. Options Simplicity: Are the options related to the questions simple ? 5. Options Independence: How varying are the options ?

Output Format:
- For each of the aspects, provide a score within the XML tags between 1-5.
- Before producing the scores, think within the XML tags <thinking>.
- Then provide the scores within the <response> XML tags.
- The scores should be outputted within the XML tags - <question_redundancy>, <question_simplicity>, <question_relevance>, <options_simplicity>, <options_independence>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<query> {domain_documents} </query>
<question> {question} </question>

Prompt G.9: Query Aspects Prompt

Instruction:
You are provided the user query and a taxonomy of aspects related to the domain of the query.
Your task is to tell what aspects present within the taxonomy is contained in the query. The query is enclosed within the <query> XML tags, while the taxonomy is enclosed within <taxonomy>. The taxonomy is a dictionary with keys as aspects and values as the aspects' description and values it can take up.
- Analyze the user query and output the aspect names (keys in the dict) that are explicitly (clearly) present in the user query without ambiguity.
- Output the name of the aspects within the XML tags <output>.
- Each aspect should be separated with commas ",". Before generating the aspects, provide your reasoning within the XML tags <thinking>.

In-context examples:
Here are some examples:
<example> ... </example>
<example> ... </example>

Input:
Now here is the input to you:
<query> {query} </query>
<taxonomy> {taxonomy} </taxonomy>