Structured List-Grounded Question Answering

Mujeen Sung[1*] **Song Feng**[2] **James Gung**[2]
Raphael Shu[2] **Yi Zhang**[2] **Saab Mansour**[2]
Kyung Hee University[1] AWS AI Labs[2]
`mujeensung@khu.ac.kr`
`{sofeng, gungj, zhongzhu, yizhngn, saabm}@amazon.com`

Abstract

Document-grounded dialogue systems aim to answer user queries by leveraging external information. Previous studies have mainly focused on handling free-form documents, often overlooking structured data such as lists, which can represent a range of nuanced semantic relations. Motivated by the observation that even advanced language models like GPT-3.5 often miss semantic cues from lists, this paper aims to enhance question answering (QA) systems for better interpretation and use of structured lists. To this end, we introduce the LIST2QA dataset, a novel benchmark to evaluate the ability of QA systems to respond effectively using list information. This dataset is created from unlabeled customer service documents using language models and model-based filtering processes to enhance data quality, and can be used to fine-tune and evaluate QA models. Apart from directly generating responses through fine-tuned models, we further explore the explicit use of Intermediate Steps for Lists (ISL), aligning list items with user backgrounds to better reflect how humans interpret list items before generating responses. Our experimental results demonstrate that models trained on LIST2QA with our ISL approach outperform baselines across various metrics. Specifically, our fine-tuned Flan-T5-XL model shows increases of 3.1% in ROUGE-L, 4.6% in correctness, 4.5% in faithfulness, and 20.6% in completeness compared to models without applying filtering and the proposed ISL method.

1 Introduction

Document-grounded dialogue systems aim to assist users in interactively seeking information from external documents to address various real-world problems with more complex scenarios as seen in customer support. While previous work has primarily treated these external documents as unstructured text (Campos et al., 2020; Feng et al., 2020, 2021; Wu et al., 2021; Gao et al., 2022; Zhao et al., 2023), a significant portion of real-world content is presented in structured formats like lists. For instance, approximately 45% of passages in public policies in the UK [1] comprise lists to effectively present conditions to be verified, action-based steps, or general itemized information (see examples in Table 1). Despite this prevalence, existing research has largely overlooked the nuanced challenges posed in understanding structured list data in relation to the complex background context of users accessing this information. Some studies have explored list information for condition verification purposes (Sun et al., 2021), but not in the realistic setup where retrieval is required to differentiate between conditional and non-conditional lists based on user backgrounds. Surprisingly, SOTA models such as GPT-3.5 (OpenAI, 2022) and Mixtral-8x7B (Jiang et al., 2024) show unsatisfactory performance on nuanced list information, as illustrated in Figure 1, despite their strong results on natural language inference (NLI) and reasoning tasks (Qin et al., 2023; Liu et al., 2023a; Guo et al., 2023).

Our work aims to address these limitations while testing LLM capabilities for more nuanced list-based content. While several benchmarks exist for evaluating QA models on tabular data (Chen et al., 2021; Nan et al., 2022; Osés Grijalba et al., 2024), much less attention has been given to assessing models' abilities on structured lists. Thus, we propose a novel benchmark called LIST2QA, designed to evaluate question answering (QA) systems on understanding list semantics with respect to user background. The dataset introduces diverse styles of list content for document grounding, such as specifying logical conditions for validation, describing actionable steps, or simply separating items without explicit logical relations. For QA samples, we

[*] Work performed during an internship at AWS AI Labs.

[1] `https://www.gov.uk`

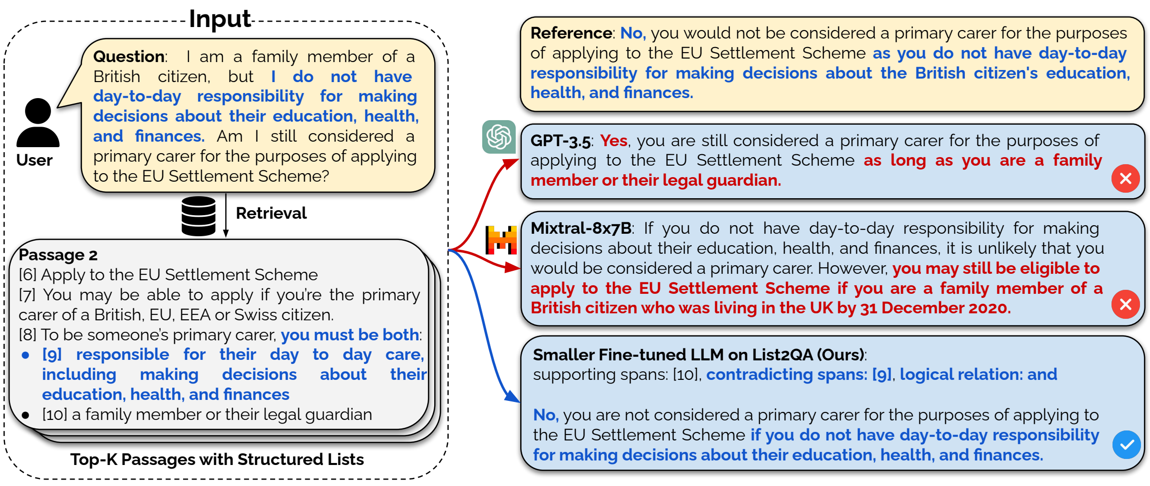

Input
Question: I am a family member of a British citizen, but I do not have day-to-day responsibility for making decisions about their education, health, and finances. Am I still considered a primary carer for the purposes of applying to the EU Settlement Scheme?
User
Retrieval
Passage 2
[6] Apply to the EU Settlement Scheme
[7] You may be able to apply if you're the primary carer of a British, EU, EEA or Swiss citizen.
[8] To be someone's primary carer, you must be both:
• [9] responsible for their day to day care, including making decisions about their education, health and finances
• [10] a family member or their legal guardian
Top-K Passages with Structured Lists
Reference: No, you would not be considered a primary carer for the purposes of applying to the EU Settlement Scheme as you do not have day-to-day responsibility for making decisions about the British citizen's education, health, and finances.
GPT-3.5: Yes, you are still considered a primary carer for the purposes of applying to the EU Settlement Scheme as long as you are a family member or their legal guardian.
Mixtral-8x7B: If you do not have day-to-day responsibility for making decisions about their education, health, and finances, it is unlikely that you would be considered a primary carer. However, you may still be eligible to apply to the EU Settlement Scheme if you are a family member of a British citizen who was living in the UK by 31 December 2020.
Smaller Fine-tuned LLM on List2QA (Ours):
supporting spans: [10], contradicting spans: [9], logical relation: and
No, you are not considered a primary carer for the purposes of applying to the EU Settlement Scheme if you do not have day-to-day responsibility for making decisions about their education, health, and finances.

Figure 1: An example of system responses to the user question grounding over list-based content. Blue texts indicate semantic cues for correct reasoning, while red texts indicate incorrect reasoning.

A Passage with a Step List
Title: Provide driving tests for your employees Qualifying as a delegated driving examiner Your employees must then: • complete an initial training course • reach an appropriate standard in the delegated driving examiner theory and practical tests
A Passage with an Option List
Title: Workplace pensions You can get free, impartial information about your workplace pension options from: • the Money Advice Service • the Pensions Advisory Service • Pension Wise if you're in a defined contribution pension scheme
A Passage with a Non-Action Info List
Title: Money and property when you divorce A mediator can help you and your ex-partner agree on how to split money and property. Mediation is not relationship counselling. It can help you agree on how you'll divide your assets, including: • pensions • property • savings

Table 1: Examples of passages with lists as steps, options, and non-action itemized information.

construct scenarios where the user background information may align with, contradict, or not address specific list items, which are oftentimes used to determine system responses.

Additionally, we explore pipeline approaches that focus on fine-tuning smaller, more efficient LLMs, demonstrating their potential to outperform larger LLMs on our benchmark dataset. Inspired by recent successes in automated data creation (He et al., 2024; Choi et al., 2024; Oh et al., 2024), we employ large language models to simulate user queries and system answers grounding over structured lists, and also investigate how to filter low-quality data to further improve performance.

Given that LLMs can often overlook logical relations among list items and their semantic alignment with user-to-item status (See Figure 1), we further investigate whether we can emphasize the semantic cues in lists and improve end-to-end performance. Thus, we introduce 'Intermediate Steps for Lists (ISL)', aligning better with how humans interpret list items before responding. By explicitly modeling structured list data and user contexts with ISL, our method outperforms baseline LLMs on the LIST2QA dataset. Specifically, our ISL fine-tuned Flan-T5-XL model (Chung et al., 2024) shows increases of 3.1% in ROUGE-L, 4.6% in correctness, 4.5% in faithfulness, and 20.6% in completeness compared to baseline fine-tuning.

Our contributions are summarized as follows:

- We present LIST2QA, a novel benchmark designed to evaluate question answering systems on nuanced, list-based content.
- We propose the Intermediate Steps for Lists (ISL) method, which enhances alignment with human interpretation of list items before generating responses.
- We demonstrate that smaller fine-tuned models using our ISL method significantly outperform larger LLMs on the LIST2QA dataset, setting a new state-of-the-art for this task.

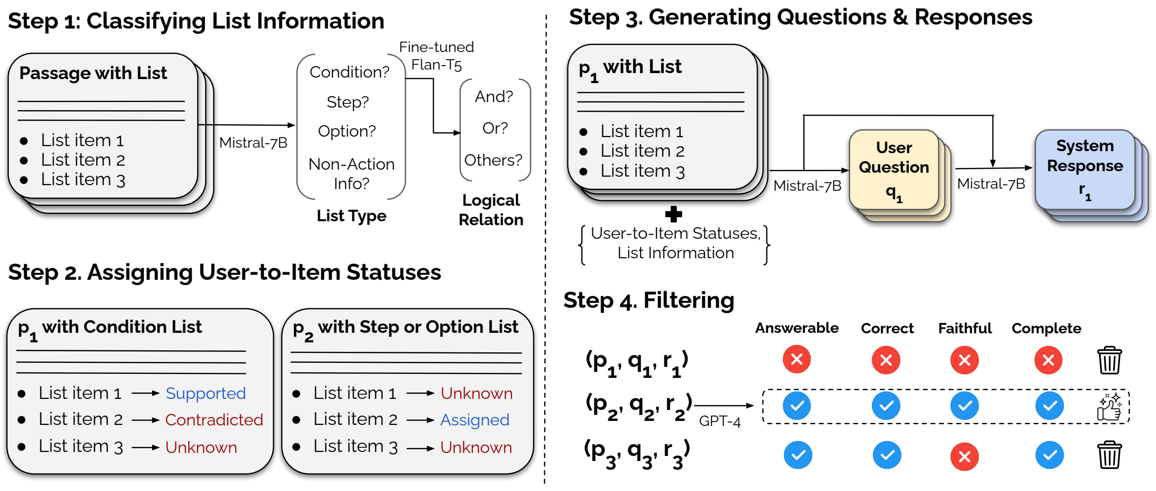


Figure 2: Overview of the LIST2QA dataset creation pipeline: (Step 1) classifying list types and logical relations in passages with lists, (Step 2) assigning user-to-item statuses for each list item, (Step 3) generating user questions and system responses from the previous steps, and (Step 4) filtering out noisy samples based on four metrics.

2 LIST2QA

In this section, we first formulate the problem of generating system responses based on different types of lists and user scenarios. We then detail the methodology used to create our LIST2QA dataset, which involves an automated and pipeline-based simulation process utilizing language models. Lastly, we present more details about the dataset for training and validation.

2.1 Problem Formulation

To formulate the problem, we first examine the various ways lists can be structured and how user scenarios interact with these lists. Our goal is to develop QA systems capable of providing responses specific to user scenarios based on relevant structured lists. We categorize lists commonly found in support documents (Feng et al., 2020; Sun et al., 2021) into the following types: conditions for eligibility ('condition'), step-by-step instructions ('step'), options for users to choose from ('option'), and the rest being mostly non-action information without explicit logical relation ('non-action info').

We define our task with an input consisting of a set of passages $\mathcal{P} \in p_1, \ldots, p_N$, a user question q_i, and the system response r_i as the model output. Each passage p_n contains list items. Each user question q_i includes user scenarios or user-to-item statuses. We follow a retrieval-augmented generation (Lewis et al., 2020) pipeline where we employ a passage retriever to select the top-K most relevant passages for each user question. Based on retrieved passages and a user question, a language model generates system response by reasoning over the information described in the relevant text. Specially, we introduce 'Intermediate Steps for Lists (ISL)', including: (1) identifying relevant passages among top-K passages, (2) classifying list types (i.e., conditions, steps, items, non-action info), (3) identifying logical relations between condition list items (i.e., and, or), and (4) determining entailment based on user-to-item statuses for condition list items (e.g., supported, contradicted, unknown) or selecting list items aligned with user-to-item statuses for step/option list items. The final system response, r_i, is generated as free-form text based on these intermediate steps.

2.2 Document Corpus

We consider two corpus sources ConditionalQA (Sun et al., 2021) and MultiDoc2Dial (Feng et al., 2021), both of which contains numerous diverse list items in their passages due to the nature of customer support documents. This allows us to develop our own dataset based on their document content. Instead of utilizing their annotations, which focus on plain text information, we use only the unlabeled documents and create all new instances as specified in Section 2.3.

ConditionalQA (Sun et al., 2021) is a dataset designed for conditional reading comprehension tasks. It uses documents related to public welfare in the UK (e.g., "Apply for Visitor Visa") and includes annotations for yes/no or extractive questions. We use document content from ConditionalQA for creating training, validation, and test sets.

List Types	User Questions	System Responses
Condition	If there is no break clause mentioned in the tenancy agreement, **can I as a tenant end the tenancy early with your approval?**	Yes, you may be able to end the tenancy early with your landlord's approval, even if there is no break clause in the tenancy agreement.
Step	I'm setting up a business partnership. First, I need to decide on a name for the partnership. **What should I do next?**	Next, you need to choose a 'nominated partner' and register with HM Revenue and Customs (HMRC) to complete the setup of your business partnership.
Option	I need a private firm to conduct my drivers' medical exam for a lorry or bus license application. **Are there any other similar services that I can consider instead?**	You could also consider visiting your GP to complete the medical examination section on the D4 form for your lorry or bus driver license application.
Non-Action Info	I'm considering suspending or leaving my course. **Which types of student finance do I need to stop paying?**	You need to stop paying your student finance payments for Maintenance Loans, Tuition Fee Loans, and any grants or bursaries you may be receiving.

Table 2: Examples of generated questions and responses for each list type in the LIST2QA dataset.

MultiDoc2Dial (Feng et al., 2021) is based on public support documents such as 'ssa.gov' and 'va.gov'. We use the unlabeled documents from the MultiDoc2Dial corpus to increase the number of test samples in LIST2QA. We can further evaluate models on domains that were not seen during training using data samples created from this source.

2.3 Dataset Creation Pipeline

Given the lack of existing datasets designed for building QA systems focused on structured lists, we propose a dataset creation pipeline specifically for addressing this challenge. First, we extract passages containing lists from unlabeled documents described in Section 2.2. [2] We aim to automate the process based on the advances of LLMs with the pipeline illustrated in Figure 2.

Step 1. Classifying list information To generate user queries with different contexts, we first identify the type of list information in each passage. Passages are categorized into one of four list types: 'condition', 'step', 'option' or 'non-action info'. For passages under the condition type, we also classify their logical relations: 'and' or 'or'. To this end, we fine-tuned Flan-T5-XL (Chung et al., 2024) using 72 manually annotated training samples. This approach significantly improved performance, achieving an F1 score of 78.0% on 30 manually annotated validation samples. See the details of the logical relation classifier in Appendix A.

Step 2. Assigning user-to-item statuses Next, we assign user-to-item statuses to one or more list items. For condition lists, we determine whether each item supports, contradicts, or is unknown in the user scenario. For step and option lists, we randomly select an item and assign it as the user status. For 'non-action info' lists, we create questions without specific user background. For condition lists, we aim to provide concluded answers ('yes', 'no', 'uncertain') that can be derived by considering both the logical relations and the user-to-item statuses of list items. As illustrated in Table 3, if a list has an 'And (Conjunctive)' relation where item 1 supports and item 2 contradicts the corresponding user-to-item statuses, the deduced answer is 'no'.

Logical Relation	User-Item Status 1	User-Item Status 2	Short Answer
And (Conjunctive)	Supported	Supported	Yes
	Supported	Contradicted	No
	Supported	Unknown	Uncertain
Or (Disjunctive)	Supported	Contradicted	Yes
	Contradicted	Contradicted	No
	Contradicted	Unknown	Uncertain

Table 3: Concluded answers derived from the logical relations and the user-to-item statuses of list items.

Split	Condition	Step	Option	Non-Action Info	Total
Train	524	224	270	369	1,387
Dev	58	43	36	51	188
Test	346	161	215	201	923

Table 4: Data statistics of LIST2QA.

Step 3. Generating user questions and system responses Based on list types and assigned user-to-item statuses, we sequentially generate user questions encompassing specific user scenarios and system responses. For this process, we employ Mistral-7B-Instruct, using three-shot in-context examples for each list type.

[2]The corpus we use is in HTML format, so we employ the <h> tag as a passage splitter and the <li> tag to indicate passages that contains list items.

Input

Given the passages, generate the system response to the user's question, including intermediate steps:

Passage 1
[1] Master's Loan
[2] Healthcare and social work
[3] You can't get a Postgraduate Master's Loan if:
• [4] you are eligible for an NHS bursary
• [5] you get a Social Work Bursary

Passage 2
[6] Social work bursaries
[7] Eligibility
[8] Social work bursaries are available to eligible social work students who:
• [9] don't get funding from their employer
• [10] don't already have a higher education social work qualification

Passage 3
[11] Social work bursaries
[12] If you're training for social work you may get a bursary.
[13] Social work bursaries:
• [14] help with living costs and tuition fees
• [15] don't depend on your household income

User question: I hold a higher education social work qualification. Am I eligible for a social work bursary?

Output

Intermediate Steps:
Relevant Passage: 2
List Type: Condition
User-to-Item Status: [7]Unknown, [8]Contradicted
Logical Relation: And

Response: No, you are not eligible for a social work bursary because you hold a higher education social work qualification.

Table 5: A sample for response generation with intermediate steps. Text in blue highlights rationale information for validation.

Step 4. Filtering samples We improve the data quality by filtering out instances with inaccurate information or hallucinations. To validate the model-based filtering, we have three human annotators label 100 examples across four dimensions: question answerability, and response correctness, faithfulness, and completeness. We specify that samples are filtered out if questions are unanswerable or responses are incorrect, unfaithful, or incomplete. We then use GPT-4 (OpenAI, 2023) to evaluate the same examples and measure the inter-annotation agreement (IAA) between human and model-based filtering, yielding a Cohen's kappa score of 56.1 (see the prompt for verification in Appendix B). Finally, we apply model-based filtering to the entire dataset, retaining approximately 51.0% of the original samples. This results in 1.4K, 0.2K, and 0.9K samples for training, development, and test sets, respectively. We use MTLD (McCarthy and Jarvis, 2010) to assess the diversity of generated samples, achieving scores of 69.9 for questions and 64.4 for answers. In contrast, the original ConditionalQA dataset scores 26.6 for questions and 10.1 for answers, demonstrating significantly higher diversity in our samples. Examples of final samples and the dataset statistics are detailed in Table 2 and Table 4.

3 Experiment

3.1 Experiment Setup

We evaluate models of various sizes on LIST2QA. For the larger LLMs, we use GPT-3.5 [3] and Mixtral-8x7B-Instruct [4](Jiang et al., 2024) in the 0- and 4-shot setting, where 4-shot examples are randomly selected from samples with four different list types. For the smaller LLMs, we fine-tune Flan-T5-XL (Chung et al., 2024) and Mistral-7B-Instruct[5] (Jiang et al., 2023) on the LIST2QA training set. We adopt QLoRA (Dettmers et al.) with a learning rate of 5e-4 for Flan-T5-XL over 10 epochs and 2e-5 for Mistral-7B-Instruct over 2 epochs. For retrieval-augmented generation, we apply LlamaIndex (Liu, 2022) with 'all-mpnet-base-v2' (Reimers and Gurevych, 2019) as the passage retriever. We set the top-K to 3, achieving a recall@3 of 93.0% on our training set. Questions without relevant passages retrieved are considered as 'unanswerable'. An example input and output with intermediate steps are shown in Table 5.

3.2 Evaluation Metric

We evaluate responses generated by models on the LIST2QA test set using both non-LLM and LLM-based evaluation. For non-LLM evaluation, we select ROUGE-L (Lin, 2004), which measures the lexical overlap between reference and generated responses. Additionally, recent work (Liu et al., 2023b; Kim et al., 2024) shows that advanced LLMs can perform fine-grained evaluations, such as detecting hallucinations or missing information, aligning well with human judgments. Therefore, we adopt the LLM-based evaluation using GPT-4 (OpenAI, 2023) to measure whether models generate correct responses ('correctness'), whether the responses are solely based on the relevant passage ('faithfulness'), and whether the responses include all the necessary information ('completeness'). Details are in Appendix C.

[3]gpt-3.5-turbo-0125
[4]Mixtral-8x7B-Instruct-v0.1
[5]Mistral-7B-Instruct-v0.2

Method	Model Size	Filtering	ROUGE-L	Correctness	Faithfulness	Completeness	Average
GPT-3.5 (0-shot)	Unknown	-	48.5	76.1	81.0	19.4	56.3
GPT-3.5 (4-shot)	Unknown	✓	54.2	86.9	85.6	63.3	72.5
Mixtral-8x7B-Instruct (0-shot)	47B	-	42.6	78.0	74.1	48.3	60.8
Mixtral-8x7B-Instruct (4-shot)	47B	✓	49.7	83.2	78.7	56.6	67.1
Flan-T5-XL (FT)	3B	✗	56.8	83.0	83.3	51.0	68.5
Flan-T5-XL (FT)	3B	✓	58.9 (+2.1)	85.3 (+2.3)	85.6 (+2.3)	64.4 (+13.4)	73.6 (+5.0)
+ ISL (Ours)	3B	✓	**59.9 (+3.1)**	**87.6 (+4.6)**	**87.8 (+4.5)**	**71.6 (+20.6)**	**76.7 (+8.2)**
Mistral-7B-Instruct (FT)	7B	✗	51.4	89.7	82.2	78.4	75.4
Mistral-7B-Instruct (FT)	7B	✓	52.5 (+1.1)	**90.5 (+0.8)**	**85.4 (+3.2)**	81.2 (+2.8)	77.4 (+3.0)
+ ISL (Ours)	7B	✓	**53.9 (+2.5)**	89.6 (-0.1)	85.3 (+3.1)	**82.2 (+3.8)**	**77.8 (+3.4)**

Table 6: Main experiment results for response generation on the LIST2QA test set across four metrics. ISL refers to generating intermediate steps for lists before generating responses. 'FT' refers to fine-tuned models. Filtering indicates whether model-based filtering was applied to improve the quality of the training set.

3.3 Experimental Results

We present evaluation results in Table 6. Notably, fine-tuned language models significantly outperform larger language models. For instance, the performance of Mixtral-8x7B-Instruct with 4-shot examples lags behind fine-tuned Flan-T5-XL and Mistral-7B-Instruct by approximately 10.0% in average score. This underscores the importance of fine-tuning models to deepen the understanding of nuanced semantic relations in list information for generating responses. Our findings further confirm the ability of fine-tuned efficient language models to outperform larger ones on specific tasks, consistent with Li et al. (2024); Fu et al. (2024).

The results also demonstrate that model-based filtering of the training data consistently results in performance improvements across two models and four metrics, despite using almost half the number of training samples. Specifically, Flan-T5-XL trained on the filtered dataset outperforms the baseline by up to 5.0% on average. Notably, filtering particularly helps to reduce incomplete response generation, achieving a 13.4% improvement.

Additionally, our ISL method, which generates intermediate steps for list information, helps further improve performance. For instance, Flan-T5-XL with ISL achieves a 3.2% higher performance than without ISL across four metrics on average. Although the correctness and faithfulness of Mistral-7B-Instruct slightly decrease after applying ISL, its overall performance still improves by 0.4%.

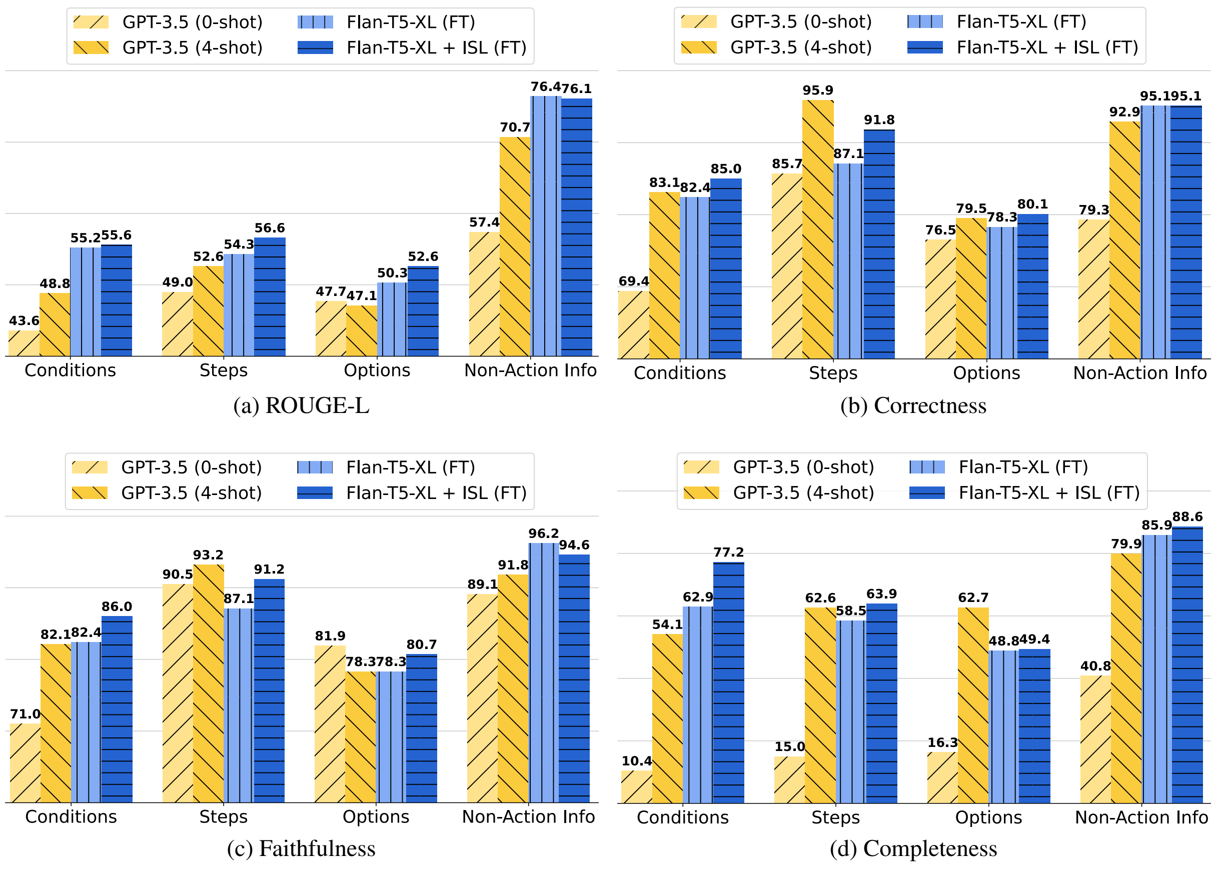
Moreover, models based on Flan-T5-XL achieve higher ROUGE-L (59.9% vs. 53.9%) and faithfulness (87.8% vs. 85.3%) scores compared to those trained on Mistral-7B-Instruct. This could be partially because Mistral-7B-Instruct is more verbose than Flan-T5-XL, often producing unnecessary phrases. Conversely, this verbosity of Mistral-7B-Instruct models rather helps produce more correct (87.6% vs. 89.6%) and complete (71.6% vs. 82.2%) responses than Flan-T5-XL, highlighting a trade-off between base language model choices.

4 Analysis

4.1 Performance on Different List Types

Different list types necessitate distinct styles of user questions and corresponding intermediate steps. To understand the performance disparities, we analyzed four list types from the LIST2QA test set: conditions, steps, options, and non-action information. Figure 3 illustrates that GPT-3.5 particularly struggles with responding over condition and non-action information lists compared to fine-tuned Flan-T5-XL by a large margin.

We observe that leveraging ISL consistently and significantly improves performance, with two exceptions in the non-action information list type regarding ROUGE-L and faithfulness. Specifically, Flan-T5-XL with ISL outperforms the baseline by up to 2.6% in correctness, 3.6% in faithfulness, and 14.3% in completeness on condition lists, highlighting the benefit of generating intermediate steps for more accurate responses. However, the exceptions observed in non-action information lists suggest that these lists do not typically require complex intermediate steps, such as tracking user-to-item status or logical relations. As a result, training the model on these lists might lead to the generation of unnecessary information in responses, thereby decreasing performance.



(a) ROUGE-L (b) Correctness

(c) Faithfulness (d) Completeness

Figure 3: Performance breakdown across the different list types.

Method	R-L	Correct	Faithful	Complete	Avg
		Seen Domain			
GPT-3.5 (0-shot)	50.4	76.5	82.2	20.5	57.4
Flan-T5-XL (FT)	60.8	87.4	87.6	68.6	76.1
+ ISL	**61.4**	**89.8**	**89.8**	**75.9**	**79.2**
Mistral-7B-Ins (FT)	52.6	90.1	83.1	83.8	77.4
+ ISL	**54.5**	**90.3**	**84.9**	**86.2**	**79.0**
		Unseen Domain			
GPT-3.5 (0-shot)	48.2	75.6	79.5	18.0	55.3
Flan-T5-XL (FT)	56.4	82.8	83.1	59.3	70.4
+ ISL	**58.0**	**84.8**	**85.3**	**66.5**	**73.6**
Mistral-7B-Ins (FT)	52.5	**91.1**	**88.4**	**78.1**	**77.5**
+ ISL	**53.2**	88.6	85.9	77.3	76.3

Table 7: Performance comparison on seen/unseen data.

4.2 Performance on Seen and Unseen Domains

To evaluate the generalizability of our approach on unseen domains, we use samples from the ConditionalQA corpus for the seen domain and samples from the MultiDoc2Dial corpus for the unseen domain. As shown in Table 7, the fine-tuned models with ISL consistently outperform those without ISL on seen data, achieving up to a 3.1% higher average score. This trend extends to Flan-T5-XL with ISL on the unseen domain, showing a 3.2% increase, which demonstrates the robustness of fine-tuning Flan-T5-XL with ISL. However, Mistral-7B-Instruct with ISL struggles on the unseen domain, with a 0.8% decrease in predicting correct intermediate steps, highlighting the need for further improvements in generalizability.

4.3 Error Cases

Analyzing error cases provides valuable insights into the limitations of our models. In Table 8, fine-tuned Flan-T5-XL with ISL generates the phrase ‘within 30 days or you could face a financial penalty’, which is not present in the gold passage. Additionally, the model omits crucial information such as ‘check with the financial aid office ...’, which should have been included in the responses. Future work could explore more sophisticated approaches, such as preference optimization (Wu et al., 2024; Rafailov et al., 2024), to improve the generation of more faithful and complete responses grounding over list information.

Gold Passages	**Evaluation on Generated Responses**
Title: Running a limited company Other changes you must report You must tell HMRC if: • your business' contact details change - for example, your name, gender, business name • you appoint an accountant or tax adviser	**Question:** I have recently changed the name of my business. Do I need to inform HMRC about this update? **Generated Response:** Yes, you need to inform HMRC about the name change of your business within 30 days or you could face a financial penalty. **Evaluation:** Correct, Unfaithful, Complete
Title: How to Correct or Update Your FAFSA Application How do I make changes to my FAFSA form? Submit your new information: • Write in the corrections or updates on your paper SAR, sign it , and mail it to the address provided on the SAR . • Check with the financial aid office at the school you plan to attend; the school might be able to make changes for you electronically.	**Question:** I need to make some corrections and updates to my FAFSA form. What should I do next with my paper SAR? **Generated Response:** Next, you should write in the corrections or updates on your paper SAR, sign it, and mail it to the address provided on the SAR. **Evaluation:** Correct, Faithful, Incomplete

Table 8: Sample errors, e.g., 'unfaithful' or 'incomplete', in generated responses by fine-tuned Flan-T5-XL with ISL. Text in red marks the information related to the errors.

5 Related Work

Document-Grounded Dialogue Our work is generally related to the document-grounded dialogue task (Feng et al., 2020, 2021; Wu et al., 2021; Gao et al., 2022; Zhao et al., 2023; Le et al., 2023). While previous research has largely concentrated on dialogue systems that respond to information-seeking user questions based on plain text knowledge, they often overlook user requests that involve verifying conditions found in support documents. Although some work, such as ShARC (Saeidi et al., 2018) and ConditionalQA (Sun et al., 2021), has begun to address this by focusing on conditional content presented as lists within documents, these tasks are still somewhat distant from real-world scenarios involving differentiating from other types of text and structured content. Our work bridges this gap by recognizing a broader range of list types and nuanced semantic relationships indicated by lists. We propose a novel approach that leverages large language models (LLMs) to better handle these complexities, thereby supporting further research in LLM-based dialogue systems.

Intermediate Steps Our approach involving Intermediate Steps for Lists (ISL) for QA systems is generally related to generating intermediate reasoning for large language models (Wei et al., 2022; Kojima et al., 2022; Zhang et al., 2022; Zhou et al., 2023; Yao et al., 2023; Huang and Chang, 2023; Yu et al., 2023; Wang and Lu, 2023), which enhances the reasoning ability of large language models. While most previous works focus on using intermediate reasoning in free-form text, structured approaches to intermediate reasoning have been proposed for mainly for specific tasks such as code generation (Li et al., 2023). Our work specifically focuses on understanding nuanced semantic relations for QA tasks based on list information. Additionally, our setup is within the context of data augmentation, featuring a development data simulation pipeline. We emphasize a pipelined approach integrating data augmentation and efficient fine-tuning to enhance the performance of smaller LLMs, particularly in handling list semantics.

6 Conclusion

We present an novel pipeline-based approach to enhance question answering systems by addressing the nuanced challenges posed by list-based content. Our primary contributions include the introduction of the LIST2QA dataset, a novel benchmark designed to assess QA systems' ability to effectively handle and respond to list information. Additionally, we develop the Intermediate Steps for Lists (ISL) method, which mirrors human interpretive processes for list items. Our experiments show that our approach, based on efficient fine-tuned models, consistently outperforms baseline approaches. By emphasizing the importance of QA systems' ability to handle list-based content with dynamic and nuanced semantics, our work paves ways for future research to further refine QA systems and expand their applicability across various domains.

Limitations

There are certain limitations in current scope of this work: (1) Although we currently handle only two types of logical relations in conditional lists, namely 'and' and 'or', there are more diverse logical relation types, such as 'nor' or nested relations, in passages containing lists. We plan to investigate these in future work. (2) While we focus only on single-turn QA tasks in this paper, multi-turn dialogues grounding over structured lists, where systems need to respond considering dialogue history, can be more practical. We leave this exploration of multi-turn dialogues grounding over structured lists for future work. (3) Evaluating models' generation across 'correctness', 'faithfulness', and 'completeness' using GPT-4 is costly (Tang et al., 2024), which hinders more extensive evaluations, and is somewhat less accurate compared to human evaluation. In the future, we aim to develop an automatic evaluation method, which is less expensive and more accurate, for structured list-grounded question answering systems.

Ethical Considerations

The dataset and models presented in this work have some ethical considerations: (1) The data simulation process should ensure diversity and avoid representation biases by incorporating input from humans with diverse backgrounds; (2) The question answering systems should provide transparent explanations for its responses to build appropriate trust with users; (3) Further testing is needed to proactively evaluate fairness and safety issues before deployment to real users, in order to prevent harm.

Acknowledgements

We gratefully acknowledge the members of AWS AI for providing valuable feedback on the project. We would also like to thank the anonymous reviewers for their insightful comments. Part of this work, conducted by Mujeen Sung as a faculty member at Kyung Hee University, was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2022-00155911, Artificial Intelligence Convergence Innovation Human Resources Development (Kyung Hee University)) (No. RS-2024-00509257, Global AI Frontier Lab), and the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2024-RS-2024-00438239) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation).

References

Jon Ander Campos, Arantxa Otegi, Aitor Soroa, Jan Deriu, Mark Cieliebak, and Eneko Agirre. 2020. DoQA - accessing domain-specific FAQs via conversational QA. In *ACL*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021. FinQA: A dataset of numerical reasoning over financial data. In *EMNLP*.

Juhwan Choi, Eunju Lee, Kyohoon Jin, and YoungBin Kim. 2024. GPTs are multilingual annotators for sequence generation tasks. In *Findings of EACL*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling instruction-finetuned language models. *JMLR*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *NeurIPS*.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021. MultiDoc2Dial: Modeling dialogues grounded in multiple documents. In *EMNLP*.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In *EMNLP*.

Xue-Yong Fu, Md Tahmid Rahman Laskar, Elena Khasanova, Cheng Chen, and Shashi Tn. 2024. Tiny titans: Can smaller large language models punch above their weight in the real world for meeting summarization? In *NAACL*.

Chang Gao, Wenxuan Zhang, and Wai Lam. 2022. Unigdd: A unified generative framework for goal-oriented document-grounded dialogue. In *ACL*.

Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, Deyi Xiong, et al. 2023. Evaluating large language models: A comprehensive survey. *ArXiv*.

Xingwei He, Zhenghao Lin, Yeyun Gong, Alex Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, Weizhu Chen, et al. 2024. Annollm: Making large language models to be better crowdsourced annotators. In *NAACL*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. In *Findings of ACL*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *ArXiv*.

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

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. 2024. Prometheus: Inducing evaluation capability in language models. In *ICLR*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *NeurIPS*.

Duong Minh Le, Ruohao Guo, Wei Xu, and Alan Ritter. 2023. Improved instruction ordering in recipe-grounded conversation. In *ACL*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *NeurIPS*.

Beibin Li, Yi Zhang, Sébastien Bubeck, Jeevan Pathuri, and Ishai Menache. 2024. Small language models for application interactions: A case study. *ArXiv*.

Jia Li, Ge Li, Yongming Li, and Zhi Jin. 2023. Structured chain-of-thought prompting for code generation. *ArXiv*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*.

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. 2023a. Evaluating the logical reasoning ability of chatgpt and gpt-4. *ArXiv*.

Jerry Liu. 2022. Llamaindex. `https://github.com/jerryjliu/llama_index`.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. Gpteval: Nlg evaluation using gpt-4 with better human alignment. In *EMNLP*.

Philip M McCarthy and Scott Jarvis. 2010. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, et al. 2022. Fetaqa: Free-form table question answering. *TACL*.

Hanseok Oh, Haebin Shin, Miyoung Ko, Hyunji Lee, and Minjoon Seo. 2024. Ktrl+ f: Knowledge-augmented in-document search. In *NAACL*.

OpenAI. 2022. Chatgpt blog post. `https://openai.com/blog/chatgpt`.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*.

Jorge Osés Grijalba, L. Alfonso Ureña-López, Eugenio Martínez Cámara, and Jose Camacho-Collados. 2024. Question answering over tabular data with DataBench: A large-scale empirical evaluation of LLMs. In *LREC-COLING*.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? In *EMNLP*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading. In *EMNLP*.

Haitian Sun, William W. Cohen, and Ruslan Salakhutdinov. 2021. Conditionalqa: A complex reading comprehension dataset with conditional answers. In *ACL*.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024. Minicheck: Efficient fact-checking of llms on grounding documents. *ArXiv*.

Tianduo Wang and Wei Lu. 2023. Learning multi-step reasoning by solving arithmetic tasks. In *ACL*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *NeurIPS*.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2024. Fine-grained human feedback gives better rewards for language model training. In *NeurIPS*.

Zeqiu Wu, Bo-Ru Lu, Hannaneh Hajishirzi, and Mari Ostendorf. 2021. Dialki: Knowledge identification in conversational systems through dialogue-document contextualization. In *EMNLP*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *ICLR*.

Ping Yu, Tianlu Wang, Olga Golovneva, Badr AlKhamissi, Siddharth Verma, Zhijing Jin, Gargi Ghosh, Mona Diab, and Asli Celikyilmaz. 2023. ALERT: Adapt language models to reasoning tasks. In *ACL*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *ArXiv*.

Yingxiu Zhao, Bowen Yu, Haiyang Yu, Bowen Li, Jinyang Li, Chao Wang, Fei Huang, Yongbin Li, and Nevin L Zhang. 2023. Causal document-grounded dialogue pre-training. In *EMNLP*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. *ICLR*.

A Details of Logical Relation Classifier

We found that classifying logical relations between list items is surprisingly difficult for large language models using in-context learning. Table 9 shows that Mistral-7B-Instruct, and even the larger Mixtral-8x7B-Instruct, with 8-shot in-context examples, struggle with the seemingly simple task of classifying 'And' or 'Or,' achieving F1 scores lower than 30 on the validation samples. To address this issue, we fine-tuned Flan-T5-XL using 72 manually annotated training samples and achieved an F1 score of 78.0 on 32 manually curated validation samples.

Model	**And F1**	**Or F1**	**Avg F1**
Flan-T5-XL (72-shot FT)	**77.6**	**78.4**	**78.0**
Mistral-7B-Instruct (8-shot IC)	34.9	22.7	28.8
Mixtral-8x7B-Instruct (8-shot IC)	49.4	4.1	26.8

Table 9: Comparison of models for classifying logical relations on 32 manually curated validation samples. 'FT' refers to fine-tuning, and 'IC' refers to in-context learning.

B Prompt for Model-based Filtering

Table 10 describes the prompt used for filtering out noisy samples in which user questions are considered unanswerable or the system responses are found to be incorrect, unfaithful, or incomplete on the given context.

C Prompt for Response Evaluation

Table 11 describes the prompt evaluating whether generated responses are correct, faithful, or complete based on the given context and the user question.

You will be evaluating a system's response to a user question, given some context. Here is the context:

<context>
{{CONTEXT}}
</context>

Here is the user's question:

<question>
{{QUESTION}}
</question>

And here is the system's response:

<response>
{{RESPONSE}}
</response>

First, determine if the question can be answered based solely on the information provided in the context. Output your reasoning inside <answerability_reasoning>. Then output "answerable" or "unanswerable" inside <answerable> tags.

Next, if the question is answerable, evaluate the system's response across three dimensions:
- Correctness: Is the response factually correct based on the context?
- Faithfulness: Does the response avoid claiming anything not directly supported by the context?
- Completeness: Does the response include all relevant information from the context to fully answer the question?

If the question is unanswerable, output "NA" for each of the three dimensions. For each dimension, first output your reasoning inside <correctness_reasoning>, <faithfulness_reasoning> and <completeness_reasoning> tags. Then output your assessment (correct/incorrect/NA, faithful/unfaithful/NA, complete/incomplete/NA) inside <correctness>, <faithfulness> and <completeness> tags.

Table 10: Prompt for model-based filtering. This involves checking whether questions are answerable, and system responses are correct, faithful, and complete.

You will be evaluating a system's response to a user question, given some context. Here is the context:

<context>
{{CONTEXT}}
</context>

Here is the user's question:

<question>
{{QUESTION}}
</question>

And here is the system's response:

<response>
{{RESPONSE}}
</response>

Evaluate the system's response across three dimensions:
- Correctness: Is the response factually correct based on the context?
- Faithfulness: Does the response avoid claiming anything not directly supported by the context?
- Completeness: Does the response include all relevant information from the context to fully answer the question?

If the question is unanswerable, output "NA" for each of the three dimensions. For each dimension, first output your reasoning inside <correctness_reasoning>, <faithfulness_reasoning> and <completeness_reasoning> tags. Then output your assessment (correct/incorrect/NA, faithful/unfaithful/NA, complete/incomplete/NA) inside <correctness>, <faithfulness> and <completeness> tags.

Table 11: Prompt for response evaluation. This involves checking whether system responses are correct, faithful, and complete.