LoCt-Instruct: An Automatic Pipeline for Constructing Datasets of Logical Continuous Instructions

Hongyu Sun, Yusuke Sakai, Haruki Sakajo, Shintaro Ozaki
Kazuki Hayashi, Hidetaka Kamigaito, Taro Watanabe
Nara Institute of Science and Technology (NAIST)
{sun.hongyu.sg6, sakajo.haruki.sd9, ozaki.shintaro.ou6}@naist.ac.jp
{sakai.yusuke.sr9, hayashi.kazuki.hl4, kamigaito.h, taro}@is.naist.jp

Abstract

Continuous instruction following closely mirrors real-world tasks by requiring models to solve sequences of interdependent steps, yet existing multi-step instruction datasets suffer from three key limitations: (1) lack of logical coherence across turns, (2) narrow topical breadth and depth, and (3) reliance on rigid templates or heavy manual effort. We introduce **LoCt-Pipeline**, a novel pipeline that leverages modern LLMs' reasoning capabilities to assemble rich, topic-related single-instruction data into multi-turn dialogues, producing chains that are logically coherent, progressively deepen in content, and span diverse domains without fixed templates or extensive human annotation. We employed this pipeline to construct **LoCt-Instruct** for assessing models' problem-solving abilities. The generated chains serve as a testbed for benchmarking a variety of models, including reasoning-oriented architectures, instruction-tuned variants, and state-of-the-art closed-source LLMs on their capacity to follow and correctly respond to each step. Our results reveal a substantial performance gap between current LLMs and human solvers. These findings highlight the need for more robust continuous instruction following. We publicly release the dataset[1] and end-to-end pipeline[2].

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in language comprehension, generation, and a wide range of downstream applications. However, despite their extensive linguistic prowess, they inherently lack robust capabilities to explicitly recognize and respond to user intentions (Ouyang et al., 2022; Lou et al., 2024b; Moon et al., 2025). To address this limitation, instruction-following tasks have been introduced, aiming to bridge the gap between shallow linguistic interpretation and the nuanced understanding of user intent necessary for executing diverse instructions. For instance, FLAN (Wei et al., 2022) constructed a diverse instruction-following dataset encompassing various NLP tasks, enhancing the zero-shot performance of LLMs (Chung et al., 2024; Kim et al., 2025; Nayak et al., 2024). WizardLM (Xu et al., 2024) and Self-Instruct (Wang et al., 2023) have iteratively enriched instruction complexity, progressively elevating task difficulty through automated data generation methods.

However, real-world instructions often appear sequentially, forming what we term "instructional chains" ordered sequences of related instructions that exhibit progressive depth and logical coherence, as shown in part 3 of Figure 1. Such sequences pose higher demands on language comprehension and intent recognition capabilities of LLMs. To date, comprehensive datasets capturing logically coherent instruction chains suitable for benchmarking LLM capabilities remain scarce. Although prior efforts such as SIFo (Chen et al., 2024) have attempted to generate instruction sequences based on predefined templates across multiple tasks, these datasets typically suffer from rigid structures and limited topical diversity.

In response, we introduce **LoCt-Instruct**, a novel dataset designed to reflect realistic, logically coherent instruction chains. As shown in Figure 1, LoCt-Instruct has the following characteristics:

1. Each instruction chain comprises multiple thematically related subtasks.
2. Instructions within each chain maintain logical continuity and progressively deepen in content complexity.
3. Evaluation is structured through a multiple-selection question answering (MSQA) format, enabling precise evaluation.
4. Extensive distractors of the options pool are

Corresponding authors: Hongyu Sun and Yusuke Sakai

[1] https://huggingface.co/datasets/Dubdabada/LoCt-Instruct

[2] https://github.com/Dearmer656/complex_task_dataset.git

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 34199–34218
November 4-9, 2025 ©2025 Association for Computational Linguistics

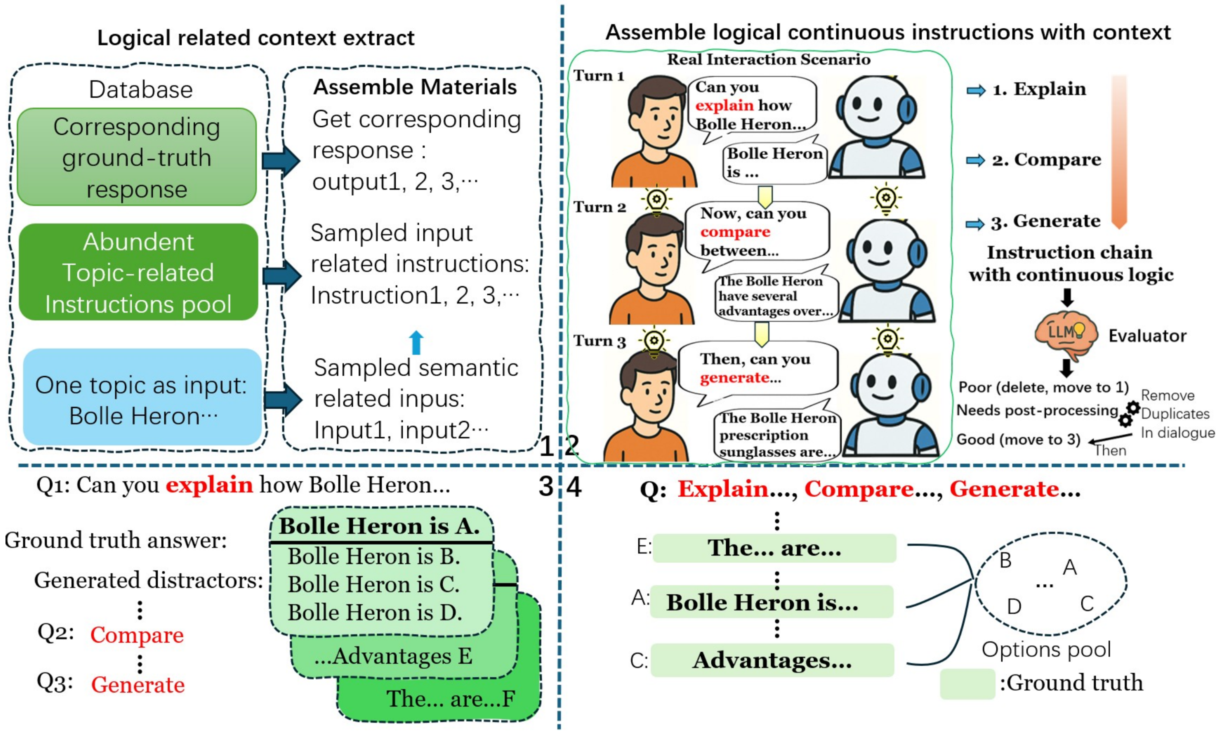

Figure 1: Overview of LoCt-Instruct creation. The upper half illustrates our two-stage instruction-based dialogue generation: (1) selecting topically and logically related instructions to construct deep, topic-centered conversations; (2) filtering logically inconsistent or reference-violating samples through automatic evaluation. The lower half depicts multi-selection QA construction; (3) for each question and its ground-truth answer, LLM generates three plausible distractors; (4) all answer options are randomly shuffled and assigned labels.

introduced to increase prompt length and rigorously assess the positional robustness.

Furthermore, LoCt-Instruct is created by **LoCt-Pipeline**, a fully automated, scalable construction pipeline. We leverage the availability and reliability of existing instruction-following task datasets. We begin by sampling thematically coherent topics and assembling multiple logically related instructions, then simulate realistic conversational dynamics, sequentially stitching together these sampled topics and instructions to ensure both logical continuity and progressive depth, as illustrated in Figure 1. Prior work (Chen et al., 2023; Ren et al., 2023) proved that self-evaluation by the LLM itself can improve generation accuracy and quality. Therefore, we deploy the self-evaluation to monitor the quality of the generated dialogue. To mitigate evaluation ambiguity, generated assistant responses from these dialogues serve as ground truth and are integrated with carefully constructed distractors, forming closed MSQA items. Model performance can thus be rigorously quantified through exact match (EM) and F1 score. Additionally, the intentional lengthening of instruction prompts through added distractors provides a stringent test of models' positional sensitivity, offering deeper insights into their robustness. The pipeline utilizes a structured high-low LLM setup that has two advantages: (1) limited cost and (2) mitigation of the potential bias (Wataoka et al., 2025).

Our experimental result on LoCt-Instruct reveals that both open-source models, e.g., Llama-3.1, Qwen-2.5, Mistral-v0.3, QwQ-32B, and even a proprietary model, i.e., GPT-4o, lag behind human performance with EM 0.642 vs 0.740. However, incorporating Chain-of-Thought (CoT) (Kojima et al., 2022) reasoning substantially improves EM and F1 scores, demonstrating that our dataset provides an objective benchmark for assessing LLMs' reasoning abilities on logical continuous instruction chains in MSQA format.

2 Related Work

Instruction-Following Datasets. Existing instruction-following datasets, such as Multi-Turn-Instruct (Sun et al., 2024) and UltraChat (Ding et al., 2023), support contextually coherent multi-turn interactions. Approaches like Logi-CoT (Liu et al., 2023) evolve simple prompts into

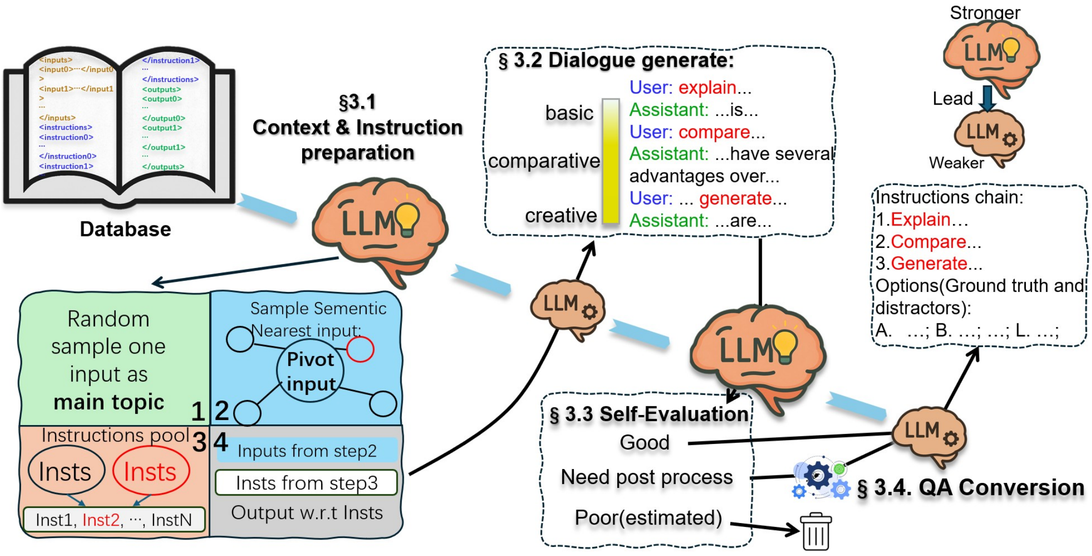

Figure 2: Pipeline overview: Two large-brain icons mark the high-capacity LLM used for reasoning—context sampling (§3.1) and self-evaluation feedback (§3.3); two small-brain icons mark the lightweight LLM handling retrieval & assembly—dialogue generation (§3.2) and QA conversion (§3.4). Light-bulb symbols above the large brains indicate their guiding role, while gear symbols on the small brains show they operate under guidance.

multi-step reasoning tasks, while benchmarks such as MuTual (Cui et al., 2020) focus on logical consistency. Retrieval-augmented frameworks, including Wizard-of-Wikipedia (Shuster et al., 2021), Retrieval-Augmented Generation (RAG) (Lewis et al., 2020), and WebGPT (Nakano et al., 2022), ground outputs in external knowledge to reduce hallucinations. However, reliance on template-driven paradigms, computationally intensive retrieval mechanisms, and limited domains restricts the diversity, explicit logical control, and scalability of these datasets.

Multi-Selection Sequential QA. Text-based multi-hop QA benchmarks, e.g., FanOutQA (Zhu et al., 2024) and MEQA (Li et al., 2024b), introduce multi-answer questions with explicit stepwise reasoning chains; likewise, MQMA (Tang et al., 2024) jointly predicts multiple answers for a sequence of linked questions over a single image, and MusTQ (Zhang et al., 2024) generates large-scale temporal KGQA (Huang et al., 2023) requiring time-aware sequential inference. However, these pipelines suffer from high annotation costs (Li et al., 2024c), rigid template-based generation (Welbl et al., 2018), and limited scale and open-domain generalization (Dua et al., 2019; Reddy et al., 2019; Choi et al., 2018; Yang et al., 2018).

MSQA Dataset Construction Pipelines. Existing MSQA efforts, e.g., MultiRC (Khashabi et al., 2018), FrenchMedMCQA (Labrak et al., 2022), KoBBQ (Jin et al., 2024), and AstroQA (Li et al., 2025), combine the LLM question synthesis with human or expert validation to ensure strict "select-all-that-apply" formats across general, medical, social-bias, and astrophysical domains; ConditionalQA (Sun et al., 2022) likewise includes some multi-selection items within its broader mix. Research dedicated to MSQA remains sparse. In contrast, our fully automated MSQA creation pipeline eliminates costly human intervention, provides fine-grained control over question difficulty and logical structure, and scales seamlessly to open-domain settings without template bias.

3 LoCt-Pipeline

We introduce the **LoCt-Pipeline** illustrated in Figure 2, a fully automated framework for constructing benchmarks composed of logically continuous instruction chains (**LoCt-Instruct**). Our goal is to construct high-quality MSQA data with logically continuous instruction chains by assembling contexts from an existing database. This requires sampling logically coherent and contextually truthful facets, leveraging the strong reasoning capabilities of the LLM. In addition, we leverage the LLM's reasoning capabilities to generate dialogues that progressively deepen the topic-related content. The pipeline leverages *logical reasoning* and *content retrieval & assemble* modules, and consists of a

# of inputs	1,463
# of instructions per input (avg.)	46.48
# of (input, instruction, output) tuples	68,014
Avg. input length (words)	119.26
Avg. instruction length (words)	84.74
Avg. output length (words)	71.32

Table 1: Statistics of the MUFFIN reference dataset.

four-stage end-to-end workflow that: 1. Context and Instruction Preparation, 2. Dialogue Generation, 3. Self-evaluation & feedback, 4. MSQA Conversion.

3.1 Context and Instruction Preparation

Our pipeline expects a seed database from a context-rich source, such as an instruction following corpus, to suit our purpose. For LoCt-Instruct, we selected the MUFFIN (Lou et al., 2024a), which pairs each input topic with multiple coherent instructions and corresponding gold responses. Table 1 shows the statistics of the MUFFIN dataset. As shown in Figure 2, the seed dataset is processed through the following four steps:

Step 1: We randomly select some *pivot inputs* from the database as our central context reference.

Step 2: For each pivot input, we retrieve the nearest neighbors based on embedding cosine similarity using the E5 (Wang et al., 2024), ensuring thematic and semantic relevance while avoiding context inputs that may disrupt logical continuity.

Step 3: Given a pivot input and its nearest neighbors, we draw 12–16 instructions for each, producing an instruction set of 24–32 items. This range is carefully chosen to balance cost, logical coherence, and instruction diversity (See § 3.5).

Step 4: We then enumerate all plausible instruction sequences from this pool using LLMs such as `gpt-4o` by prompt in Table 12 of Appendix D. The model is instructed to rank the sequences primarily based on logical coherence and continuity, with an explicit emphasis that logical consistency takes precedence over simply leveraging extensive context information.

Finally, we select the top-ranked instruction sequence as the optimal logical plan for dialogue generation. This structured approach ensures logical continuity and coherence while flexibly leveraging abundant context, thus optimizing the quality of the generated dialogue in subsequent stages.

3.2 Dialogue Generation

We generate logically coherent dialogues by following the instruction sequences, which serve as the foundation for heavy logical reasoning and instruction chain planning. Since the content has already been defined by these sequences, the generation phase requires less reasoning capability. Therefore, we can employ lighter-weight LLMs such as `gpt-4o-mini`, than those used in Section 3.1.
In the generation phase, as shown in Table 11 of Appendix D, we prompt `gpt-4o-mini` with a semantically coherent context set consisting of the pivot input, its nearest neighbors, and the top-ranked instruction chain with corresponding gold responses, in order to produce a multi-turn dialogue.

3.3 Self-Evaluation & Feedback

To ensure the quality of the generated dialogues, we apply an automatic evaluation to filter out those that may compromise logical coherence or answer completeness. Inspired by Li et al. (2024a), each dialogue is evaluated from three perspectives: (1) *factual errors or inconsistencies*; (2) *breaks in logical flow or reasoning*; and (3) *dialogue completeness issues*, such as misalignment with the provided content, responses abruptly cut off mid-sentence, and redundant or duplicated content across turns. Based on these checks, we assign each dialogue one of three ratings and apply the corresponding processing steps:

***Good*:** Dialogues without any detected issues proceed directly to the MSQA generation stage.

***Needs Post-processing*:** Dialogues exhibiting only abrupt cut-offs or redundancy undergo automated removal of duplicate turns.

***Poor*:** Dialogues with factual errors, logical breaks, major incompleteness (aside from abrupt cut-offs), or irrelevance are discarded.

Table 13 in Appendix D presents detailed prompts and settings. This feedback mechanism ensures our final dataset remains reliable and coherent. In addition to this evaluation, we conducted multiple human assessments of dialogue quality across varying sample sizes, and the issue-reporting procedure in the final MSQA tests provided a further check on data integrity.

3.4 MSQA Conversion

Given the synthesized multi-turn dialogue, we treat each user turn as an instruction and its corre-

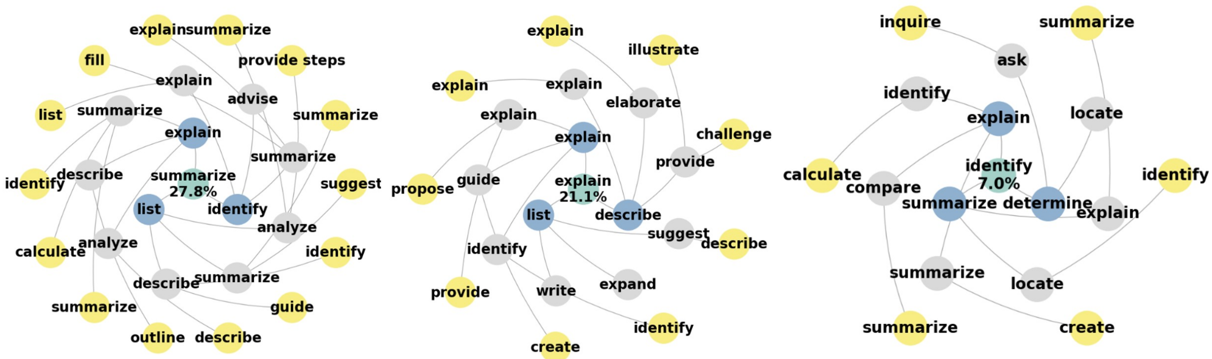

Figure 3: Instruction chains of maximum depth 3 constructed from the three most prevalent depth-1 instructions. Colors distinguish the different depths within each chain.

sponding assistant reply as the gold response. We then prompt a large language model to generate three distractor responses, yielding four candidate answers per instruction. This process produces MSQA dataset; however, during evaluation, we reframe the task as multiple-choice, requiring the model to select the optimal response from the candidates for each instruction. Figure 6 in Appendix D shows an example of the final output. This transformation serves multiple important purposes:

1. **Objective Evaluation:** Closed-form MSQA instances with clearly defined gold answers and distractors allow for accurate and consistent assessment of model performance.

2. **Context Length and Complexity:** Introducing distractor choices increases context length and complexity, enabling evaluation of model performance under more challenging.

3. **Robustness to Positional Bias:** The presence of distractors helps test the robustness of LLMs against positional biases, ensuring that models genuinely understand the context rather than relying on superficial or positional.

Using the dialogue as context, we create question–answer pairs where the gold answer corresponds to the pivot input. The distractor choices are generated through the following two sub-steps:

Attribute Extraction: We extract key textual attributes, e.g., entities, relations, and numerical values, from each distilled answer using an LLM such as `gpt-4o-mini`, forming the structured basis for distractor generation.

Distractor Generation: We use `gpt-4o-mini` to generate distractors by making minimal variations to the extracted attributes, following the protocol of Sakai et al. (2024). To ensure a balance between diversity and quality: (1) distractors must read naturally and be grammatically correct; and (2) three distractors are generated for each sub-question. We described the detailed prompts, settings, and examples in Appendix D.

Quality w.r.t. Instruction Count				
Instruction #	4–8	8–12	12–16	16–20
Proportion	0.25	0.375	0.125	0.25

Table 2: Distribution of poor-quality Dialogue Outputs by instruction count. Instruction count refers to the number of instructions sampled from the instruction pool corresponding to one input.

Next, to block shortcut answering, we pool all options for the entire question into one candidate set. For instance, a four-subquestion instance yields sixteen total options (one correct answer + three distractors per subquestion).

3.5 Statistics and Qualities of LoCt-Instruct

Dialogue Quality. Through the LoCt-Pipeline, we obtained 271 multiple-choice MSQA items with context dialogue. We conducted a human evaluation to systematically assess the quality of the generated dialogue data across the different sampling number settings, which also served as an indirect assessment of the overall pipeline. We evaluated based on the following two criteria: **user content**, focusing on logical continuity and coherence, and **assistant response**, focusing on context-matching accuracy, completeness, and alignment with the provided context. The evaluation consisted of two parts. First, we measured inter-annotator agreement on a small, shared subset: 10 examples sampled from the generated dialogues were independently annotated, yielding substantial agreement

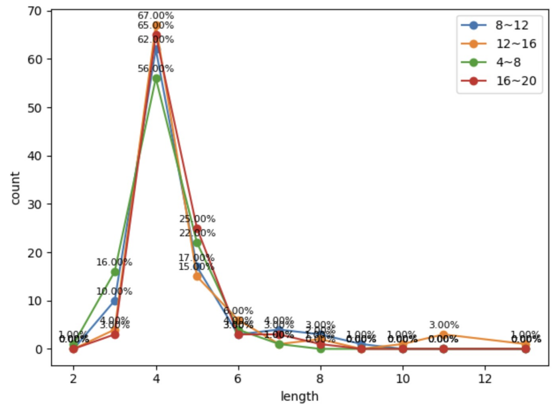

Figure 4: Distribution of instruction chain lengths constructed with varying numbers of sampled instructions. Different colors indicate different initial sampled numbers of instructions w.r.t each input.

Cohen's κ = 0.615 (Cohen, 1960). Second, we conducted a larger human evaluation on a non-overlapping set: we sampled 25 test instances for each of four length segments (four groups of 25, 100 in total), split them into two sets, and randomly assigned the sets to the evaluators. Dialogues from different length groups were presented in random order. Table 16 in Appendix E provides detailed evaluation guidelines. Table 2 presents the evaluation results, showing the proportion of cases that exhibit at least one issue. The evaluation results aggregated feedback from two evaluators reveal a clear trade-off between informational richness and the logical complexity of chain assembly. When only 4–8 instructions are provided, there is insufficient material to construct coherent, multi-turn dialogues, leading to a high *"Poor"* proportion (25 %). Conversely, supplying 16–20 instructions imposes a combinatorial burden on the LLM's assembly process, again yielding a 25 % *"Poor"* rate. Intermediate instruction counts (8–12) get worse (37.5 %), but the 12–16 bin achieves the best balance with the lowest *"Poor"* rate (12.5 %) and sufficient context for logical continuity. Therefore, to optimize dialogue quality while controlling computational cost, we adopt 12–16 instructions per instance for our pipeline.

Effect of The Dialogue Turns. We calculated the average length or number of instructions of the generated instruction chains for each group in Figure 4. The results indicate that the average instruction length does not correlate with the initial sampling number. However, increasing the number of input instructions considerably raises the associated computational cost and evaluation workload without a corresponding increase in dialogue quality. To balance cost-efficiency, the average instruction chain length, and dialogue quality, we determined that sampling 12 to 16 instructions is optimal for our dataset construction pipeline.
In Section 3.4, we visualize the instruction chain of the most frequent initial instructions, where each node connects to the next three most frequent instructions: *summarize* (27.8%), *explain* (21.1%), and *identify* (7.0%). This pattern matches our expectation that question exploration typically begins with a general overview.

Diversity. We examine the distribution of instruction categories across different positions within each chain. That is, we not only analyze the overall frequency of each instruction type, but also their relative positions (e.g., whether logical reasoning typically appears at the beginning, middle, or end). This analysis reveals three key observations:

1. **Instruction diversity across positions.** As shown in Figure 3, all instruction types are distributed throughout the chain, with no category strictly confined to a specific position. This indicates that the generated chains support flexible combinations of tasks, such as beginning with summarization, followed by reasoning, and ending with extraction or transformation.

2. **Compositional flexibility.** The variability in instruction types across positions suggests that the generated chains mirror the structure of real-world complex tasks, which often require dynamically switching between task types rather than following a rigid sequence. Moreover, across our dataset, no two instruction chains are identical, highlighting the structural uniqueness and diversity of each generated chain.

3. **Evaluator feedback and reasoning time.** We recruited two graduate-level evaluators. Before the main evaluation, we measured inter-annotator agreement on 10 shared examples, and Cohen's κ (Cohen, 1960) was 0.737. We then asked them to answer our constructed MSQA items and assign a topic label to each item. Their feedback was as follows: the 50 questions spanned 28 unique topics. Commonsense reasoning was the predominant category (19/50; 38.8%), followed by history (2/50; 4.0%), chemistry (2/50; 4.0%), and single instances in domains such as physics, medicine, geography, and market analysis, etc. The evaluators re-

quired an average of 3.2 minutes to complete each item, demonstrating that our questions necessitate substantial reasoning time to arrive at the correct answers.

These results underscore the richness of our dataset not only in task diversity but also in compositional structure. Such flexibility makes the dataset more representative of multi-step reasoning and planning tasks in real-world applications.

4 Experimental Settings

LLMs. We evaluated seven representative LLMs, encompassing both open-source and proprietary models, including reasoning-oriented variants: QwQ-32B (Team, 2025), and instructing-oriented variants: LLaMA3.1-8B-Instruct, Qwen-2.5-7B-Instruct (Yang et al., 2025a), Qwen-2.5-14B-Instruct (Yang et al., 2025b), Mistral-7B-Instruct (Jiang et al., 2023), GPT-4o-mini[3], GPT-4o (OpenAI et al., 2024). The used devices are introduced in Appendix A.2. Experiments were conducted across several prompting scenarios: zero-shot prompting, one-shot prompting with and without Chain-of-Thought (CoT) reasoning. For one-shot in-context learning, we enforce strict non-overlap between prompt examples and test instances: any candidate sharing input data with the test item is discarded and replaced. Because our generation process ties instructions and outputs directly to inputs, distinct inputs inherently produce distinct chains, thus preventing data leakage.

Human Evaluation. We also included human baselines by recruiting two graduate-level participants who independently answered the same questions without access to external tools. Their performances were averaged to establish a human reference. We randomly sampled 25 examples from our evaluation dataset for each evaluator. In addition, evaluators were instructed to report any issues that prevented them from responding. Appendix E provides detailed instructions for human evaluation.

Evaluation Metrics. In our evaluation of MSQA chains, we employ two complementary metrics: sequence-level Exact Match (EM) and micro-averaged F1. EM requires the model's entire answer string (e.g., "ABD...") to exactly match the reference, enforcing correct order and content across the chain, and is averaged over all chains (perfect matches total chains). F1 treats each question-option pair as an independent instance, counting true positives, false positives, and false negatives to compute precision, recall, and F1, allowing partial credit, and is averaged over all options.

[3]https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

Model	EM		avg. F1	
	w/o CoT	w/ CoT	w/o CoT	w/ CoT
Llama-3.1-8B	0.060	0.258	0.256	0.603
Llama-3.1-8B 1-shot	0.089	0.21	0.591	0.607
Qwen2.5-7B	0.052	0.129	0.553	0.632
Qwen2.5-7B 1-shot	0.114	0.202	0.636	0.660
Qwen2.5-14B	0.148	0.262	0.512	0.690
Qwen2.5-14B 1-shot	0.148	0.390	0.343	0.741
Mistral-7B-v0.3	0.074	0.052	0.438	0.4574
Mistral-7B-v0.3 1-shot	0.107	0.133	0.496	0.642
QwQ-32B 1-shot	–	0.524	–	0.753
GPT-4o-mini	0.369	0.439	0.742	0.760
GPT-4o-mini 1-shot	0.292	0.494	0.738	0.779
GPT-4o	0.531	0.579	0.824	0.853
GPT-4o 1-shot	0.561	0.653	0.851	**0.882**
Human evaluation	–	**0.74**	–	0.866

Table 3: Performance comparison on the LoCt-Instruct benchmark across three model categories: instruction, reasoning oriented, and closed-source LLMs.

5 Experimental Results and Discussions

Overall. Table 3 summarizes evaluation results. The EM–F1 gap arises because smaller LLMs often generate extra answer options, which artificially inflate the partial overlap (and hence the F1 score). Our results indicate a clear hierarchy in performance among the evaluated models. Closed LLMs, benefiting from extensive training resources, consistently outperformed their open-source counterparts, underscoring superior text comprehension and instruction-following capabilities. This performance gap highlights current LLM limitations in maintaining context, transferring information across multiple instruction turns, and coherently structuring responses in complex reasoning tasks.

Comparison to Human Evaluation. Human evaluators excelled in sequential instruction tasks, showing superior context retention, coherence, and logical structuring. In quantitative terms, they achieved an exact-match score (EM) of 0.74 and an average F1 of 0.866, compared to 0.653 EM and 0.882 F1 for GPT-4o, one of the strongest closed-source LLMs, we analysis this result in Appendix B. We also examined the evaluators' feedback: of the 13 questions they answered incorrectly, 1 was due to incomplete context, 3 were originated from commonsense topic(answer series of questions about one story), and the remaining 9 originated from domain-specific topics (physics, medicine, chemistry (2), geography, biography, web development, biology, and cybersecurity). These findings confirm that our dataset is comprehensive and of high quality. We aggregated feedback from two evaluators and summarized statistical information on accuracy, issues, and topics. The 9 out of 13 errors made by human evaluators were confined to out-of-domain questions, an entirely expected and reasonable outcome, further validating the dataset's robustness for benchmarking multi-turn instruction comprehension and reasoning in LLMs.

Prompting. Notable gains of up to +0.198 EM were observed, with Llama-3.1-8B showing the largest improvement. Reasoning-oriented models such as QwQ-32B deliver additional boosts of 0.524 (EM) and 0.753 (F1), underscoring the essential role of effective reasoning on our dataset.

In-Context Learning. Only GPT-4o, Qwen-2.5-7B, LLaMA-3.1-8B, and Mistral-7B-v0.3 show EM improvements from one-shot prompting, suggesting that in the absence of explicit reasoning, models struggle to extract task knowledge from a single example. Manual inspection of Qwen-2.5-7B, LLaMA-3.1-8B, and Mistral-7B-v0.3 outputs reveals that, when given a one-shot exemplar, they rigidly follow the prescribed format, whereas without an example, they produce intermediate analyses before answering. Moreover, their baseline performance under prompting is extremely low (EM: 0.06, 0.052 and 0.074 respectively), indicating limited understanding of our questions. We therefore attribute these modest gains to learning fixed answer templates rather than genuine comprehension.

In-Context Learning with CoT. Combining one-shot in-context learning with Chain-of-Thought prompting yields consistent gains. For example, Qwen2.5-7B achieves an additional +0.088 EM over one-shot alone, Qwen2.5-14B jumps to 0.390 from 0.148 with only one-shot, and models that degraded under isolated one-shot prompting recover or improve when CoT is applied. This suggests that CoT scaffolding enhances the model's ability to internalize and apply exemplar patterns, delivering performance improvements beyond the sum of each method.

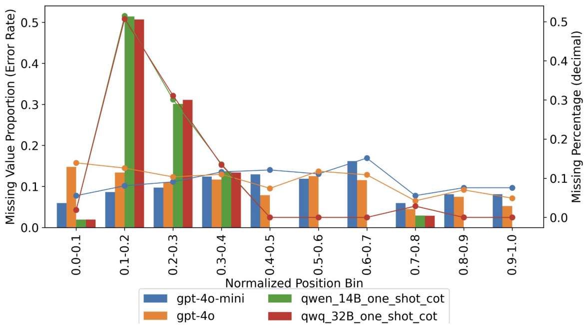

Figure 5: Distribution of missed correct answers by normalized position for each model. Positions are computed by dividing the option's index by the total number.

Category	Avg. length	Min. length	Max. length
Question	57.35	7	299
Options	681.85	39	1,715
Context	1,356.22	79	8,091

Table 4: Length statistics of LoCt-Instruct.

Positional Bias. We analyzed model sensitivity to the positional bias (Liu et al., 2024) of correct answers. Figure 5 illustrates the percentage of missed correct options relative to their positions. Errors predominantly occurred at earlier option positions (e.g., options A and B), even for closed-source models. This demonstrates that our dataset effectively reveals LLMs' positional biases due to its extensive coverage of instruction lengths. We analyzed each model's sensitivity to answer position. Figure 5 shows the percentage of missed correct options at each position. To analyze positional bias, we exclude any model with F1 < 0.7, since its errors stem from misunderstanding the options rather than genuine positional effects. Our results indicate that errors are predominantly concentrated in the earliest answer positions, a trend especially pronounced in open-source models. In contrast, closed-source models exhibit a substantially milder position bias on our dataset. These findings confirm that the broad range of sequence lengths in our dataset effectively exposes the positional biases inherent to open-source LLMs.

6 Conclusion

We introduced LoCt-Pipeline, a fully automated, scalable dataset construction framework that synthesizes LoCt-Instruct, a logically continuous MSQA corpus. We apply LoCt-Pipeline to generate a diverse set of evaluation instances, benchmarking them on reasoning-oriented, instruction-oriented, and closed LLMs. A human evaluation by graduate-level annotators further validates both the quality of our generated data and the reproducibility of our protocol. Analysis of the annotated cases confirms high data quality, no duplicate instruction chains, and broad topical coverage, while revealing a notable gap between human and machine performance. As for Scalability and Future Directions, although our current evaluation is based on a moderate dataset, LoCt-Pipeline readily extends to large-scale instruction-following corpora (Xu et al., 2024). In future work, we will scale our method to construct massive, logically continuous MSQA datasets and train models on this expanded data to further enhance multi-step reasoning capabilities.

7 Limitations

Scale and Topical Diversity. Despite demonstrating the promise of LoCt-Pipeline, our evaluation is subject to two key limitations. First, since the Muffin dataset provides only 1,460 unique inputs, we generated just 271 logically continuous, non-overlapping evaluation instances. This limited dataset scale may underrepresent rarer reasoning phenomena and constrain the topical diversity of our benchmarks, potentially affecting the robustness and generalizability of our findings.

Scalability and Hyperparameter Optimization. Although LoCt-Pipeline can theoretically be applied to arbitrarily large instruction-following corpora, scaling up to significantly larger datasets will necessitate careful redesign of sampling strategies and hyperparameter schedules (e.g., sampling ratios, chain-length distributions). Such optimizations are crucial to maintaining logical coherence and computational efficiency at larger scales.

Reasoning Ability Evaluation. We just make MSQA format to test the LLMs, so it is limited. For reasoning evaluation, we need a sentence-by-sentence evaluation method.

8 Ethical Considerations

Mitigating Bias in Data Generation. This study introduces a novel instruction-following dataset created by enhancing publicly available corpora with outputs generated by state-of-the-art large language models (LLMs). Although this methodology facilitates scalability and enriches dataset diversity, it inherently risks propagating biases and inaccuracies from the original datasets or biases present in the pretraining of the utilized LLMs. To address these risks, we implement systematic bias detection and filtering mechanisms, rigorously select source materials to ensure comprehensive topical and demographic representation, and perform manual audits on randomly sampled subsets of the generated data.

Transparency and Responsible Dataset Release. Upon releasing the dataset, comprehensive documentation will accompany it, detailing the provenance of source materials, model configurations, filtering strategies, and acknowledged limitations. Such transparency is crucial for ensuring reproducibility and encouraging responsible usage within both academic and industrial contexts. Additionally, this documentation serves as a guideline for ethical considerations, assisting third parties in understanding and responsibly reusing the dataset.

Use of AI Assistance. We employed GPT-4o only as a writing aid in the preparation of specific manuscript components, particularly for enhancing linguistic clarity and refining prompt templates. All other aspects, such as the research proposal, core technical decisions, dataset construction methodologies, analytical interpretations, and manuscript writing, were exclusively developed and executed by the research team. The scope of AI tool utilization was strictly limited, ensuring it did not affect the integrity of human-driven decision-making processes. We confirm that this work contains no harmful content and fully complies with all aspects of the ACL Ethics Policy.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This work has been supported by JSPS KAKENHI Grant Number 25K24369, and JST SPRING. Grant Number JPMJSP2140.

References

Jiefeng Chen, Jinsung Yoon, Sayna Ebrahimi, Sercan Arik, Tomas Pfister, and Somesh Jha. 2023. Adaptation with self-evaluation to improve selective prediction in LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5190–5213, Singapore. Association for Computational Linguistics.

Xinyi Chen, Baohao Liao, Jirui Qi, Panagiotis Eustratiadis, Christof Monz, Arianna Bisazza, and Maarten de Rijke. 2024. The SIFo benchmark: Investigating the sequential instruction following ability of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1691–1706, Miami, Florida, USA. Association for Computational Linguistics.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

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, and 16 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1406–1416, Online. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiang Huang, Sitao Cheng, Yuheng Bao, Shanshan Huang, and Yuzhong Qu. 2023. MarkQA: A large scale KBQA dataset with numerical reasoning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10241–10259, Singapore. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Jiho Jin, Jiseon Kim, Nayeon Lee, Haneul Yoo, Alice Oh, and Hwaran Lee. 2024. KoBBQ: Korean bias benchmark for question answering. *Transactions of the Association for Computational Linguistics*, 12:507–524.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Jungwoo Kim, Minsang Kim, and Sungjin Lee. 2025. Sedi-instruct: Enhancing alignment of language models through self-directed instruction generation. *Preprint*, arXiv:2502.04774.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Yanis Labrak, Adrien Bazoge, Richard Dufour, Beatrice Daille, Pierre-Antoine Gourraud, Emmanuel Morin, and Mickael Rouvier. 2022. FrenchMedMCQA: A French multiple-choice question answering dataset for medical domain. In *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI)*, pages 41–46, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

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. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Huayang Li, Siheng Li, Deng Cai, Longyue Wang, Lemao Liu, Taro Watanabe, Yujiu Yang, and Shuming Shi. 2024a. TextBind: Multi-turn interleaved multimodal instruction-following in the wild. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9053–9076, Bangkok, Thailand. Association for Computational Linguistics.

Jie Li, Fuyong Zhao, Panfeng Chen, Jiafu Xie, Xiangrui Zhang, Hui Li, Wang Yanhao Chen, Mei, and Ming Zhu. 2025. An astronomical question answering dataset for evaluating large language models. *Scientific Data*, 12(447):1–14.

Ruosen Li, Zimu Wang, Son Quoc Tran, Lei Xia, and Xinya Du. 2024b. Meqa: A benchmark for multi-hop event-centric question answering with explanations. In *Advances in Neural Information Processing Systems*, volume 37, pages 126835–126862. Curran Associates, Inc.

Yaoyiran Li, Xiang Zhai, Moustafa Alzantot, Keyi Yu, Ivan Vulić, Anna Korhonen, and Mohamed Hammad. 2024c. Calrec: Contrastive alignment of generative llms for sequential recommendation. In *Proceedings of the 18th ACM Conference on Recommender Systems*, RecSys '24, page 422–432, New York, NY, USA. Association for Computing Machinery.

Hanmeng Liu, Zhiyang Teng, Leyang Cui, Chaoli Zhang, Qiji Zhou, and Yue Zhang. 2023. LogiCoT: Logical chain-of-thought instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2908–2921, Singapore. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Renze Lou, Kai Zhang, Jian Xie, Yuxuan Sun, Janice Ahn, Hanzi Xu, Yu Su, and Wenpeng Yin. 2024a. MUFFIN: Curating multi-faceted instructions for improving instruction following. In *The Twelfth International Conference on Learning Representations*.

Renze Lou, Kai Zhang, and Wenpeng Yin. 2024b. Large language model instruction following: A survey of progresses and challenges. *Computational Linguistics*, 50(3):1053–1095.

Hyeonseok Moon, Jaehyung Seo, Seungyoon Lee, Chanjun Park, and Heuiseok Lim. 2025. Find the intention of instruction: Comprehensive evaluation of instruction understanding for large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5944–5964, Albuquerque, New Mexico. Association for Computational Linguistics.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. Webgpt: Browser-assisted question-answering with human feedback. *Preprint*, arXiv:2112.09332.

Nihal Nayak, Yiyang Nan, Avi Trost, and Stephen Bach. 2024. Learning to generate instruction tuning datasets for zero-shot task adaptation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12585–12611, Bangkok, Thailand. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Jie Ren, Yao Zhao, Tu Vu, Peter J. Liu, and Balaji Lakshminarayanan. 2023. Self-evaluation improves selective generation in large language models. In *Proceedings on "I Can't Believe It's Not Better: Failure Modes in the Age of Foundation Models" at NeurIPS 2023 Workshops*, volume 239 of *Proceedings of Machine Learning Research*, pages 49–64. PMLR.

Yusuke Sakai, Hidetaka Kamigaito, and Taro Watanabe. 2024. mCSQA: Multilingual commonsense reasoning dataset with unified creation strategy by language models and humans. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14182–14214, Bangkok, Thailand. Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haitian Sun, William Cohen, and Ruslan Salakhutdinov. 2022. ConditionalQA: A complex reading comprehension dataset with conditional answers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3627–3637, Dublin, Ireland. Association for Computational Linguistics.

Yuchong Sun, Che Liu, Kun Zhou, Jinwen Huang, Ruihua Song, Xin Zhao, Fuzheng Zhang, Di Zhang, and Kun Gai. 2024. Parrot: Enhancing multi-turn instruction following for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9729–9750, Bangkok, Thailand. Association for Computational Linguistics.

Peng Tang, Srikar Appalaraju, R. Manmatha, Yusheng Xie, and Vijay Mahadevan. 2024. Multiple-question multiple-answer text-VQA. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 73–88, Mexico City, Mexico. Association for Computational Linguistics.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024. Text embeddings by weakly-supervised contrastive pre-training. *Preprint*, arXiv:2212.03533.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. 2025. Self-preference bias in llm-as-a-judge. *Preprint*, arXiv:2410.21819.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025a. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, Junyang Lin, Kai Dang, Kexin Yang, Le Yu, Mei Li, Minmin Sun, Qin Zhu, Rui Men, Tao He, and 9 others. 2025b. Qwen2.5-1m technical report. *Preprint*, arXiv:2501.15383.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Tingyi Zhang, Jiaan Wang, Zhixu Li, Jianfeng Qu, An Liu, Zhigang Chen, and Hongping Zhi. 2024. MusTQ: A temporal knowledge graph question answering dataset for multi-step temporal reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11688–11699, Bangkok, Thailand. Association for Computational Linguistics.

Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. 2024. FanOutQA: A multi-hop, multi-document question answering benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 18–37, Bangkok, Thailand. Association for Computational Linguistics.

A Additional Information

A.1 Detailed Model Settings

Table 5 provides detailed model settings which we used for our experiments.

LLMs	Hugging Face ID /API NAME
Llama-3.1-8B-Instruct	meta-llama/Llama-3.1-8B-Instruct
Qwen2.5-7B-Instruct	Qwen/Qwen2.5-7B-Instruct
Qwen2.5-14B-Instruct	Qwen/Qwen2.5-14B-Instruct
Mistral-7B-Instruct-v0.3	mistralai/Mistral-7B-Instruct-v0.3
QwQ-32B	Qwen/QwQ-32B

Table 5: Lists of the LLMs we used in this study and their corresponding Hugging Face IDs.

A.2 Used Device for Evaluation

All evaluated models were uniformly quantized to 8 bits to ensure consistency across experiments. Inference for 14 B-parameter models was executed on a single NVIDIA RTX A6000 GPU (48 GB), while inference for 7B and 8B parameter models was conducted on a single NVIDIA GeForce RTX 3090 GPU.

A.3 API Usage and Cost

We provide the LoCt-Instruct cost in Table 6.

B Analysis of Model Performance on LoCt-Instruct

Our dataset consists of multi-step, logically chained questions, each requiring the selection of

Step	Model	Cost (USD)
Context sampling	gpt-4o-0806	$0.1215
Candidates generation	gpt-4o-mini-0718	$0.1970
Attribute extract	gpt-4o-mini-0718	$0.0494
QA conversion	gpt-4o-mini-0718	$0.0704
Dialogue generation	gpt-4o-mini-0718	$0.1205
Evaluation	gpt-4o-0806	$0.0718
Total (271 datasets)		**$0.6307**

Table 6: API costs for each step in constructing LoCt-Instruct and the total cost.

multiple correct answers while excluding distractors in a coherent context. As shown in Table 3, the large model achieves a higher F1 score (0.882 vs. 0.866 for humans) but a substantially lower EM (0.653 vs. 0.740). This gap reveals several key limitations:

- **Global Consistency.** In a multi-step logical chain, each answer depends on the previous step. The model often omits or adds distractors at individual steps, causing its final answer set to diverge from the ground truth and lowering EM.
- **Precision and Recall Trade-off.** Although the model attains higher precision (0.878 vs. 0.866 for humans) and higher recall (0.898 vs. 0.866), indicating strong per-option accuracy and coverage, any single extra or missing selection still yields an EM score of zero.
- **Error Propagation in Chained Reasoning.** Because each step's output feeds into the next, a minor mistake can be amplified downstream, misaligning the ultimate answer set.
- **Human Logical Deduction vs. Pattern Matching.** Comparing with LLMs, human evaluators rely on deductive reasoning across sub-questions. If a human fails to correctly answer one sub-question in the chain, subsequent sub-questions become inaccessible, causing the entire answer chain to collapse. This highlights the fundamental difference: humans perform step-wise logical inference, whereas LLMs tend to match and aggregate independent signals.
- **Long Range Dependency and Context Tracking.** For deep, multi-step reasoning or cross-question links (e.g., coreference resolution, conditional constraints), the model struggles to maintain all prior information as flexibly as humans, undermining overall completeness.

Implementing these improvements will help LLMs deliver both high coverage and exact-match accuracy in logically continuous, multi-answer scenarios.

C Prompt for Evaluation

Table 7 illustrates the combined prompt used for MSQA with both one-shot learning and CoT. Table 8 shows the prompt used for answering MSQA directly, without additional context or guidance. Table 9 shows the prompt used for MSQA with chain-of-thought (CoT) prompting. Table 10 presents the prompt for MSQA under one-shot learning with CoT.

D Prompt for LoCt-Pipeline

Figure 6 shows an instance of generated MSQA. Table 11 presents topic and instruction sampling strategies, while Table 12 illustrates representative instruction samples used during dialogue generation. The self-evaluation and feedback mechanism is outlined in Table 13. Additionally, Table 14 and Table 15 detail the procedures for extracting textual attributes and generating distractor candidates, respectively.

E Human Evaluation Metrics

We employed in-house, graduate-level annotators. Human evaluation was carried out in two distinct stages. The guidelines for assessing dialogue quality in the first stage are detailed in Table 16, while those for evaluating final MSQA answers in the second stage are provided in Table 17.

Please answer the following multiple-selection questions.
Example:
{example}
######

Now considering the question below:
{context}
<questions>
{question}
</questions>
For each question:
1. Write your step-by-step thought process under a section titled ### Reasoning:.
2. Then state your chosen options letter for each question.

After you've reasoned through all questions, output exactly this block (no extra text): ### final answers:
<LETTER_SEQUENCE>{eos}

Table 7: Prompt for answering MSQA with CoT and one shot learning.

You will get: {context}
<questions>
{question}
</questions>
Please answer each question by selecting one option. Do NOT output any reasoning, analysis, or chain-of-thought. Then output exactly: ### final answers:
<YOUR_LETTER_SEQUENCE>{eos}

Table 8: Prompt for answering MSQA.

```
You will get:
{context}
<questions>
{question}
</questions>
For each question:
1. Write your step-by-step thought process under a section titled ### Reasoning:
2. Then state your chosen option letter for each question.
After you've reasoned through all questions, output only the sequence of selected letters in this exact format (without any extra text):
### final answers:
<LETTER_SEQUENCE>{eos}
```

Table 9: Prompt for answering MSQA with CoT.

```
You will get:
{context}
<questions>
{question}
</questions>
For each question:
1. Write your step-by-step thought process under a section titled ### Reasoning:.
2. Then state your chosen option letter for each question.

After you've reasoned through all questions, output only the sequence of selected letters in this exact format (without any extra text):

### final answers:
<LETTER_SEQUENCE>{eos}
```

Table 10: Prompt for answering MSQA with one-shot learning.

You are given three tagged sections:
`<inputs>. . . </inputs>`,
`<instructions>. . . </instructions>`
`<outputs>. . . </outputs>`.
{input, instruction and output pair} Step-by-step Reasoning:

1. Which <inputX> or <instructionY> does the next User turn need to reference?
2. How does this question build on the previous dialogue to probe deeper?
3. What exact information from the context must be used—no more, no less?

Generation Rules:

- Generate a multi-turn dialogue (User–Assistant) that gradually deepens the topic.
- User turns should explicitly reference prior content if multiple inputs exist.
- Assistant responses must only use information from <inputs>, <instructions>, and <outputs>, with no hallucinations or marker abbreviations.
- It is allowed to interpolate logically coherent connective content (e.g., clarification questions), but without contradiction.

Summary Annotation:

- After dialogue generation, summarize the user-side actions using: <tasks> 1.task1; 2.task2; ... </tasks>

Final Output Format:

```
User: ...
Assistant: ...
...
<tasks> ... </tasks>
<topic domain> ... </topic domain>
```

{input, instruction and output pair}
<inputs> <input0> An essay isn't disagreeing with themselves. </input0><input1> qutting may be a flawed film , but it is nothing if not sincere . </input1> </inputs>
<instructions> <instruction1> Given an essay, check if it contains any logical inconsistencies. The output should be either True if there are no contradictions, or False if there are any. </instruction1> <instruction3> Identify if the essay contains any contradictory arguments or statements. Output True if there are no contradictions, otherwise output "False.'13134': True.'1664': False. Your answer must be a single letter chosen from '1664, 13134'. </instruction3> <instruction4> Identify the sentiment of the input. Your task is to generate either positive, negative or neutral as output, depending on whether the text expresses a positive, negative or neutral sentiment respectively. </instruction4> <instruction11> Categorize the following sentence into one of the five different emotions: anger, confusion, joy, sadness and neutral. </instruction11> </instructions>
<outputs> <output1> True </output1> <output3> 13134 </output3> <output4> positive </output4> <output11> neutral </output11> </outputs>

Table 11: Prompt template for generating dialogue with progressive deepen content.

You are given a series of input blocks, each wrapped in <inputs></inputs>, with numbered <instructions></instructions> and corresponding <outputs></outputs>. Instructions are numbered sequentially from 1 to M across all blocks. Your task is to extract instruction subsets that can be ordered into coherent, logically continuous chains.
Requirements:
- Identify subsets of instructions where each step directly builds on or depends upon the context established by the previous one.
- For each chain, include a <contextX> tag listing the input-block index(es) (e.g. [0,2]) from which those instructions were drawn.
- Reasoning which instructions can connect logically step by step, then output the final chain within <contextX></contextX> and <chainX></chainX> tags.
- Only include chains of four or more instructions. Prefer chains that cover the maximum number of distinct input blocks and include as many instructions as possible.
- Instructions may come from the same block or from different blocks; combining instructions from multiple inputs is encouraged.
- Exclude any instruction whose <outputs> does not provide a definitive answer (e.g., “cannot answer,” “no clear answer,” etc.).
- If no valid chains exist, output only: "None"
{Sampled input, instruction and output}
Output in the following format:
<context1>[0, 1]</context1><chain1>[5, 2, 1]</chain1>
<context2>[1]</context2><chain2>[3, 1, 11]</chain2>
...
or
"None"
If there are multiple choices, list all chains in descending order of logical coherence (the most continuous chain first).

Table 12: Prompt for sampling logical-related instruction.

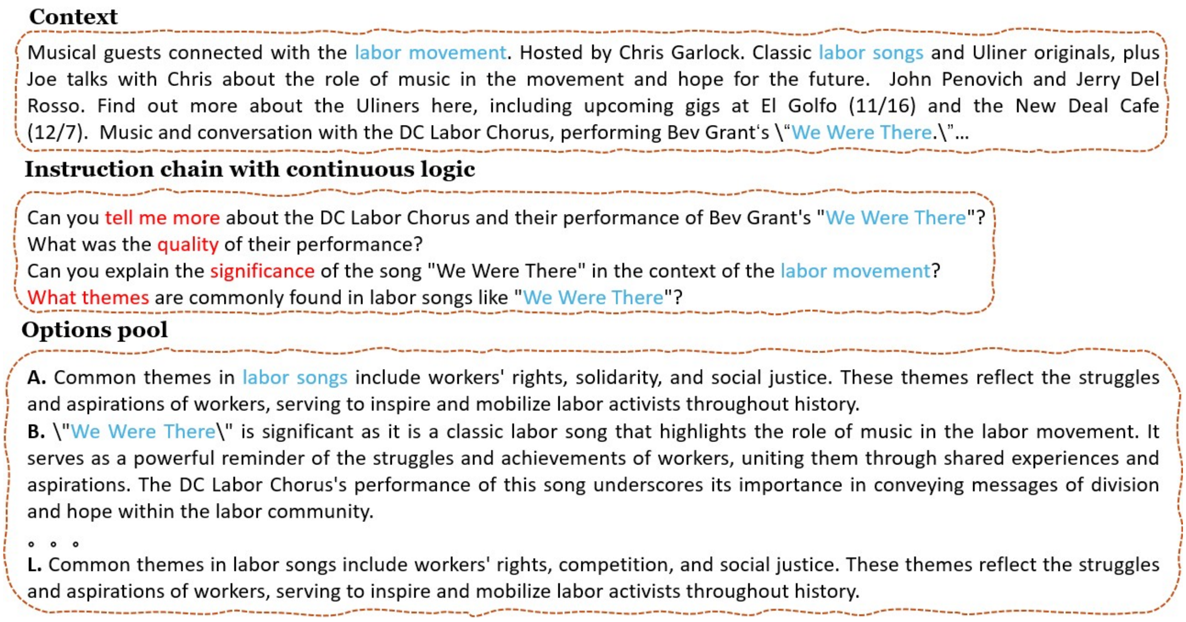

Figure 6: An example of a generated instruction chain with coherent and logically continuous content, presented in the closed multi-choice QA format.

You are an evaluation assistant.
{Instruction and output pair}
{Dialogue to be evaluated}
Given a dialogue and its associated <instructions>, and <outputs>, assess the dialogue according to the following criteria:
Instruction and Output pair
1. Global: Detect any factual errors or inconsistencies.
2. User (Logical): Identify breaks in logical flow or reasoning.
3. Assistant (QA Completeness):
a. Misalignment with <instructions>, or <outputs>.
b. Abrupt cut-offs (e.g., stopping mid-sentence or immediately after a colon).
c. Redundant or duplicated content across turns.
Please consider the quality of the dialogue step by step.
Finally, assign an overall rating: **Poor**, **Needs Post-processing**, or **Good**. If it ranked as Needs Post-processing you should give the repeated turn within <repeated turn></repeated turn> if not don't give this tag.
Please put only rank follow "request=[N]: [rank]" at the end of response.
Strictly obey the format: "<repeated turn>[index of the later repeated turn]</repeated turn> request-[N]: [evaluation]", N is provided at beginning of the dialogue.

{Instruction and Output pair}
<instructions><instruction1> In this task, you are given information about the user's inspiration for their art. Your task is to identify the source of the sea glass used in the user's jewelry. The output should be the location where the sea glass was found. </instruction1> <instruction2> The task is to write a blog post about the benefits of using genuine sea glass in jewelry-making. The output should discuss the history and properties of sea glass, why it makes a unique material for jewelry, and how the user incorporates it into their designs. </instruction2>...**</instructions><outputs>**<output1> A beach in the Caribbean </output1> <output2> Sea glass, also known as beach glass, is a type of glass that has been weathered by the ocean waves and sand over time. It is a unique material that is highly sought after by jewelry makers for its beauty and history. The history of sea glass dates back to ancient times when glass was first invented. In the past, people would dispose of their glass waste by throwing it into the ocean. Over the years, the glass would be tumbled and polished by the waves, resulting in the smooth and frosted appearance that we see today. One of the main benefits of using genuine sea glass in jewelry-making is its uniqueness. Each piece of sea glass is one-of-a-kind, with its own shape, color, and texture. This makes it perfect for creating one-of-a-kind jewelry pieces that stand out from the crowd. Incorporating sea glass into jewelry designs can be a creative and rewarding process. As mentioned in the input, the user's wire work is Art Nouveau in style... </output2> ...**</outputs>**
{Dialogue to be evaluated}
User: Can you tell me where you found the sea glass that you use in your jewelry? **Assistant:** A beach in the Caribbean.**User:** That's interesting! Can you explain why using genuine sea glass in jewelry-making is beneficial?**Assistant:** Sea glass, also known as beach glass, is a type of glass that has been weathered by the ocean waves and sand over time. It is a unique material that is highly sought after by jewelry makers for its beauty and history ...
User: Can you describe your design process in more detail?**Assistant:** My design process starts with a rough idea and a long piece of 20 gauge wire. From there, the piece grows as I work on it. My wire work is Art Nouveau in style and is inspired by nature and the sea...

Table 13: Evaluation prompt.

questions answers pairs: **{QA pair}**
Instruction: Given the above pairs of questions and answers, what kind of textual attributes do answers have?
Requirements:
1. Please brainstorm as many textual attributes as possible. If you think there are no more suitable attributes, end up with 'None'.
2. Be creative. Any interesting perspectives are welcome!
3. Each attribute must concisely summarize one specific aspect of this input, such as language, length, intent, etc.
4. Feel free to ignore the tedious and specific content. Just focus on some general textual attributes!
5. Please prioritize your most confident predictions. Generate attribute for each question answer pair separately in the format:
<answer1>1: ; 2: ...</answer1>
...

{QA pair}
<questions>((1)). Can you tell me where you found the sea glass that you use in your jewelry?((2)). That's interesting! Can you explain why using genuine sea glass in jewelry-making is beneficial?((3)). How do you incorporate sea glass into your designs?((4)). Can you describe your design process in more detail?**</questions>**

<answers>((1)). A beach in the Caribbean.((2)). Sea glass, also known as beach glass, is a type of glass that has been weathered by the ocean waves and sand over time. It is a unique material that is highly sought after by jewelry makers for its beauty and history. The history of sea glass dates back to ancient times when glass was first invented. In the past, people would dispose of their glass waste by throwing it into the ocean. Over the years, the glass would be tumbled and polished by the waves, resulting in the smooth and frosted appearance that we see today. One of the main benefits of using genuine sea glass in jewelry-making is its uniqueness. Each piece of sea glass is one-of-a-kind, with its own shape, color, and texture. This makes it perfect for creating one-of-a-kind jewelry pieces that stand out from the crowd.((3)). Incorporating sea glass into jewelry designs can be a creative and rewarding process. As mentioned, my wire work is Art Nouveau in style. Sea glass can complement this style beautifully, as its organic shapes and natural colors align with the nature-inspired motifs of Art Nouveau. I start with a rough idea and a long piece of 20 gauge wire, just like I do with my other designs. I then wrap the wire around the sea glass, creating a unique and intricate wire-wrapped pendant or charm.((4)). My design process starts with a rough idea and a long piece of 20 gauge wire. From there, the piece grows as I work on it. My wire work is Art Nouveau in style and is inspired by nature and the sea. I primarily work with wire and glass. I began making jewelry in 2008, starting with simple stringing techniques. Over time, I taught myself how to repair and recreate Victorian beaded purses, which captivated me with their beauty. I am entirely self-taught and have always had the ability to visualize and bring my visions to life. Jewelry making is a significant part of my life, providing me with peace and serving as my therapy.
</answers>

Table 14: The prompt for attributes extracting of answers.

{QA pair}
You are given a composite text that contains several sub-questions within `<questions></questions>` and their corresponding answers within `<answers></answers>`. Your task is to generate four candidate answer options for each sub-question individually, following these instructions:

- For each sub-question, create four options labeled A., B., C., and D.
- Exactly one option must be fully correct, using the provided answer **verbatim**.
- The other three must be plausible distractors that:
  - Appear semantically reasonable;
  - Reflect a common misconception or subtle logical flaw;
  - Read naturally and remain grammatically correct.
- Only generate distractors. Do not include any explanations or reasoning.
- Format your output as follows:

```
<Q1>[question1]</Q1>
A. ...
B. ...
C. ...
D. ...
Answer: [A/B/C/D]
```

The corresponding textual attributes of the answers are provided within <attributes></attributes>.
{Attributes}
These include language style, tone, level of detail, and structure. You may use them as cues to generate distractors.

QA pair consistent with the prompt for attributes extracting. **{Attributes}**
`<answer1>` Language: Informal and conversational; Length: Short and concise; Intent: To provide a specific location; Clarity: Clear and straightforward. `</answer1>`
`<answer2>` Language: Informative and descriptive; Length: Medium; Intent: To explain the benefits of using sea glass; Clarity: Detailed and thorough; Tone: Educational. `</answer2>`
`<answer3>` Language: Creative and artistic; Length: Medium; Intent: To describe incorporation of sea glass into designs; Clarity: Focused on technique; Tone: Inspirational. `</answer3>`
`<answer4>` Language: Reflective and personal; Length: Long; Intent: Detail design process; Clarity: Comprehensive; Tone: Passionate; Structure: Sequential explanation. `</answer4>`
`</attributes>`

Table 15: Prompt template and attribute annotations used for distractor generation.

Your task is to evaluate the quality of the generated dialogues and classify each dialogue as either Poor or OK.
A dialogue should be rated as Poor if it exhibits any of the following issues:

1. Factual errors – The assistant response contains information that is factually incorrect.
2. Contradiction with the reference – The response contradicts the provided <inputs>, <instructions>, or <outputs>.
3. Discontinuity in user queries – The user's turns are not logically connected; later queries do not build upon earlier one.
4. Mismatch between query and response – The assistant's reply does not address or answer the user's question or instruction.
5. Incomplete responses – The assistant's output is clearly truncated or unfinished (e.g., "The answer is:" without follow-up content).

Otherwise, label it as OK.

Table 16: Prompt for human evaluation of multi-turn dialogue.

Human Evaluator Instructions
For each test item, evaluators are presented with a context and a single multiple-choice question composed of several logically connected sub-questions; each sub-question has exactly one correct answer. Evaluators must: Select Answers for each sub-question. Record the total time taken to complete the question (e.g., in seconds). Annotate the high-level topic domain (e.g., "Physics," "Commonsense," etc.). Report any issues encountered, choosing one if any of the following apply: Necessary information is missing from the context. The quality-label criteria do not apply or are ambiguous. The set of options is clearly truncated or unfinished. Output format (exactly these fields): Answers: Time (mins:secs): Topic domain: Issue: Leave Issue: blank if there were no problems; otherwise name the one issue you encountered (you may assign randomly if you must). Note: Do not use any external tools to answer the question.

Table 17: Instructions for human evaluation of the MSQA task.