AmbigNLG: Addressing Task Ambiguity in Instruction for NLG

Ayana Niwa[1,2,3*] **Hayate Iso**[1]
[1] Megagon Labs [2] Recruit Co., Ltd. [3] MBZUAI
ayana@megagon.ai hayate@megagon.ai

Abstract

We introduce AmbigNLG, a novel task designed to tackle the challenge of *task ambiguity in instructions* for Natural Language Generation (NLG). Ambiguous instructions often impede the performance of Large Language Models (LLMs), especially in complex NLG tasks. To tackle this issue, we propose an ambiguity taxonomy that categorizes different types of instruction ambiguities and refines initial instructions with clearer specifications. Accompanying this task, we present AmbigSNI$_{\text{NLG}}$[1], a dataset consisting of 2,500 annotated instances to facilitate research on AmbigNLG. Through comprehensive experiments with state-of-the-art LLMs, we demonstrate that our method significantly enhances the alignment of generated text with user expectations, achieving up to a 15.02-point increase in ROUGE scores. Our findings highlight the importance of addressing task ambiguity to fully harness the capabilities of LLMs in NLG tasks. Furthermore, we confirm the effectiveness of our method in practical settings involving interactive ambiguity mitigation with users, underscoring the benefits of leveraging LLMs for interactive clarification.

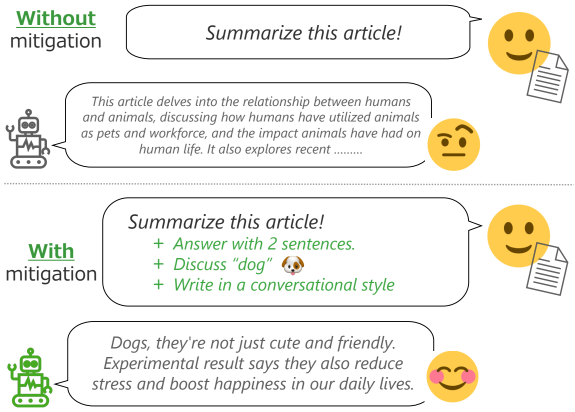

Without
mitigation
Summarize this article!
This article delves into the relationship between humans and animals, discussing how humans have utilized animals as pets and workforce, and the impact animals have had on human life. It also explores recent ....
With
mitigation
Summarize this article!
+ Answer with 2 sentences.
+ Discuss "dog"
+ Write in a conversational style
Dogs, they're not just cute and friendly. Experimental result says they also reduce stress and boost happiness in our daily lives.

Figure 1: Overview of our mitigation approach for the AmbigNLG task. We address task ambiguity by incorporating additional instructions into the initial instruction, thereby refining the task definition and improving the alignment of generated outputs with user expectations.

1 Introduction

Recent advancements in LLMs (Brown et al., 2020; Touvron et al., 2023; Jiang et al., 2023) and instruction-tuning techniques (Wei et al., 2022; Sanh et al., 2022; Ouyang et al., 2022; Rafailov et al., 2023) have significantly expanded the capabilities of these models to tackle a wide range of problems through natural language interactions. They now achieve near human-level performance on various benchmarks (Hendrycks et al., 2021; Zheng et al., 2023). However, the effectiveness of LLMs is highly dependent on the clarity and specificity of the instructions they receive (Wang et al., 2024). Ambiguous instructions often lead to suboptimal or unintended results, highlighting a critical challenge in the practical deployment of these models.

Crafting precise instructions that unambiguously specify the expected outputs is inherently challenging for humans, especially for complex tasks such as Natural Language Generation (NLG). For instance, the instruction for summarization in the Super-Natural Instruction (SNI) benchmark (Wang et al., 2022) is simply stated as "*Your task is to summarize them,*" which is fairly ambiguous. It lacks crucial details such as the desired length of the summary, the key points to include, and the intended style. This type of ambiguity, known as *task ambiguity* (Tamkin et al., 2022), is prevalent in various NLG tasks and must be addressed to effectively accomplish the task.

To address the issue of *task ambiguity in instructions* for NLG, we first introduce **AmbigNLG**, a novel task aimed at identifying and mitigating am-

[*] Work done at Megagon Labs & Recruit Co., Ltd. Intellectual property rights retained by Megagon Labs & Recruit Co., Ltd.

[1] AmbigSNI$_{\text{NLG}}$ dataset is available at `https://github.com/megagonlabs/ambignlg`

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 10733–10752
November 12-16, 2024 ©2024 Association for Computational Linguistics

biguities in various NLG instructions (§2). We then propose an ambiguity mitigation method that enhances initial instructions with clearer specifications (Figure 1, §3). This method involves establishing an ambiguity taxonomy to systematically categorize different types of instruction ambiguity in NLG tasks. Based on this taxonomy, we refine the initial instruction by appending additional instructions for each category. This approach is intended for human-in-the-loop ambiguity mitigation, enabling users to directly choose the most suitable clarifications suggested by the LLM to effectively mitigate ambiguities (§6). Furthermore, to support our proposed method, we construct the AmbigSNI$_{\text{NLG}}$ dataset, comprising 2,500 instances annotated with ambiguity taxonomy and corresponding additional instructions (§4).

We conducted a comprehensive analysis using several LLMs—including LLaMa-2, Mistral, Mixtral, and GPT-3.5—to evaluate the effectiveness of our proposed mitigation method. The results indicate that our approach of providing additional instructions successfully mitigates task ambiguity, as evidenced by significant improvements in the alignment of generated text with user expectations, as well as a reduction in output diversity (§5). Furthermore, a case study involving real human interaction confirms the practical utility, underscoring the importance of ambiguity mitigation in fully harnessing the capabilities of LLMs (§6).

2 Task: AmbigNLG

We address the challenge of task ambiguity in instruction, which arises from insufficiently defined tasks. Our aim is to enhance the accuracy of text generation to better meet users' expectations. Unlike previous studies that focus on ambiguity in Natural Language Understanding (NLU) tasks (Finn et al., 2018; Tamkin et al., 2022, 2023), our work uniquely concentrates on mitigating ambiguity in NLG task instructions. In the NLG setting, addressing ambiguity requires more adaptable strategies due to the multifaceted nature of ambiguities, such as summary length and content. To this end, we propose AmbigNLG task, specifically designed to tackle task ambiguity in NLG instructions.

2.1 Problem Definition

In instruction-based NLG tasks, the goal is to generate an output text y from a given input text x, following an instruction I (Wei et al., 2022; Wang et al., 2022). For a specific input x and instruction I, there often exists a range of valid output texts, denoted as $\mathcal{Y}_{\text{valid}}$. Modern NLG models such as LLMs are capable of generating such valid outputs $\hat{y} \in \mathcal{Y}_{\text{valid}}$. However, if the instruction I is not well specified, the LLMs may generate an output that, while valid $\hat{y} \in \mathcal{Y}_{\text{valid}}$, does not align with the user's actual intent—that is, $\hat{y} \notin \mathcal{Y}_{\text{desired}}$, where $\mathcal{Y}_{\text{desired}} \subseteq \mathcal{Y}_{\text{valid}}$. We define this phenomenon as ***task ambiguity in instructions*** for NLG, referring to unclear or insufficiently detailed instructions that hinder the LLM's ability to generate text aligned with user intentions. Conversely, if the set of valid outputs $\mathcal{Y}_{\text{valid}}$ matches the user's desired outputs $\mathcal{Y}_{\text{desired}}$, the instruction I is considered unambiguous for the input x.

2.2 Task Ambiguity Mitigation

Building on the definition above, we formulate *task ambiguity mitigation in instructions* as the process of refining an initial instruction I_{init} into a more precise instruction I_{refined}. This refinement aims to narrow the set of valid output texts $\mathcal{Y}_{\text{valid}}$ to more closely align with the user's desired outputs $\mathcal{Y}_{\text{desired}}$. Given the intractable nature of defining both valid and desired output sets, we simplify the problem by using a reference text y_{ref} as a proxy for the desired output. The objective is to refine the initial instruction so that the generated text $\hat{y}$ more closely matches the reference text y_{ref}.

3 Method for Ambiguity Mitigation

3.1 Ambiguity Taxonomy

To effectively mitigate task ambiguity in instructions, it is crucial to first identify and understand the types of ambiguities present in instruction-based NLG datasets. To this end, we conducted a comprehensive literature survey to explore the fundamental components in NLG systems (Reiter and Dale, 1997; McDonald and Pustejovsky, 1985; Kukich, 1983; Barzilay and Lapata, 2005; Reitter et al., 2006; Fan et al., 2018). Building upon insights from the literature, we manually analyzed 100 instruction-based NLG instances from Super-Natural Instruction (SNI) benchmark (Wang et al., 2022) to build an ambiguity taxonomy.[2] This analy-

[2] Specifically, each instance consists of a triplet: input, output, and instruction, with a total of 23,796 words across 100 randomly sampled instances. After comparing these triplets with a broad range of NLG literature and thorough detailed discussions, we identified 484 specific ambiguous points and categorized them.

Taxonomy	Definition	Template	Example of Filler
Context	Uncertainty of the situation or background	`Additional context: ___`	The main factors of climate change are natural phenomena and human activities.
Keywords	Not sure which words to include	`Include ___ in your response.`	global warming
Length	Underspecified length	`Answer with ___ words.`	10 to 20
Planning	Uncertainty of the text structure	`Please generate the output based on the following outline: 1. ___ 2.___, ...`	1. a brief definition, 2. causes, ...
Style	Underspecified writing style	`Write in a ___ style.`	persuasive
Theme	Uncertainty of the main subject	`Primarily discuss the following theme: ___.`	the impact of human activities

Table 1: Ambiguity taxonomy, definitions, templates, and examples of fillers for each template. The filler serves as an example given the instruction ‘Write a summary about climate change. This taxonomy helps in systematically categorizing and addressing different types of ambiguities in NLG tasks.

sis led us to identify six dominant types of task ambiguity: Context, Keywords, Length, Planning, Style, and Theme, as detailed in Table 1.

3.2 Instruction Refinement

To mitigate task ambiguity in instructions, we refine the initial instruction using our proposed taxonomy. Directly rewriting the initial instruction I_{init} to craft a refined instruction I_{refined} presents challenges in maintaining consistency and quality. Therefore, we simplify the process by appending *additional instructions* $\{I_{c_1}, \ldots, I_{c_n}\}$ to address each identified task ambiguity category $\{c_1, \ldots, c_n\}$ found in the initial instruction I_{init}. We concatenate these additional instructions with the initial instruction to create the refined instruction $I_{\text{refined}} = I_{\text{init}} \oplus I_{c_1} \oplus \cdots \oplus I_{c_n}$, where $\oplus$ denotes the text concatenation operator. These refined instructions I_{refined} serve as pseudo-references for unambiguous instructions, facilitating the study of ambiguity mitigation in NLG tasks.[3]

4 Dataset: AmbigSNI$_{\text{NLG}}$

To evaluate our mitigation method described in §3, we constructed the AmbigSNI$_{\text{NLG}}$ dataset.[4] AmbigSNI$_{\text{NLG}}$ is derived from the NLG dataset within the SNI benchmark, which encompasses 1,616 diverse NLP tasks. Each instance in this dataset is annotated with our ambiguity category c and the corresponding additional instruction I_c.

4.1 LLM-in-the-loop Annotation

Annotating ambiguity categories c and additional instruction I_c through crowdsourcing is challenging due to the open-ended nature of text generation tasks. To address this issue, we adopt an LLM-in-the-loop approach, where we manually *curate* and *verify* the dataset by guiding the LLM's generation (Ding et al., 2023; Gilardi et al., 2023; Zhang et al., 2024). To ensure consistency in the annotation of additional instructions, we developed specific templates t_c for each ambiguity category c, as shown in Table 1. These templates are filled out to create the additional instructions (Iso et al., 2020; Liu et al., 2023c; Zhou et al., 2023a; Iso, 2024). The data creation process is outlined in Figure 2. Note that for the Keywords and Length, additional instructions can be curated using a rule-based approach described in Appendix A.4; therefore, only validation is performed for these categories.

Curation We curated high-quality manual annotations of existing ambiguity in instructions. First, we manually analyzed 100 instruction-based NLG instances from SNI benchmark to identify types of ambiguities. These 100 samples were randomly selected to cover a wide variety of tasks, including question answering, summarization, and dialogue generation. Then, we annotated the additional instructions for each instance by filling in the blanks of the corresponding templates. To ensure the quality of the additional instructions, we employed a rigorous annotation process detailed in §A.3.

Generation Given the manual annotations, we fine-tuned GPT-3.5 to generate the additional instruction for each ambiguity category.[5] We provided initial instruction I_{init}, input text x, reference text y_{ref}, and the template t_c corresponding to each ambiguity category c as inputs to generate the addi-

[3] If multiple ambiguities are present, the additional instructions are concatenated in alphabetical order based on the ambiguity category names.

[4] Details on intended usage are provided in Appendix A.1.

[5] https://openai.com/blog/gpt-3-5-turbo-fine-tuning-and-api-updates

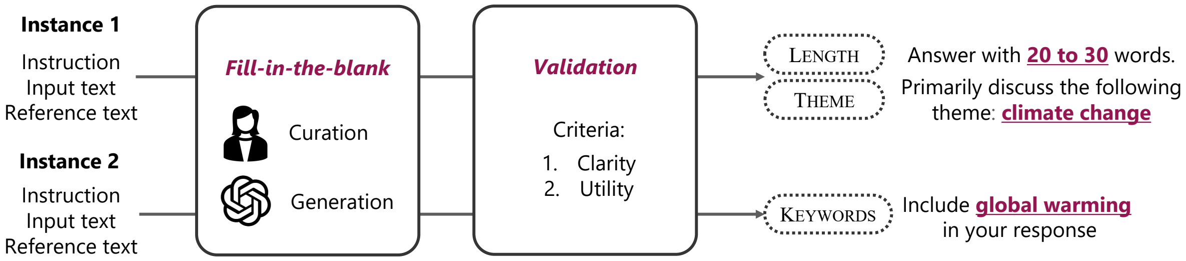


Figure 2: Dataset creation process. The process includes curating high-quality manual annotations, generating additional instruction candidates, and validating these candidates to ensure clarity and utility.

		# Additional Instructions %					# Ambiguity Type %					
Split	#data	0	1	2	3	4+	CONTEXT	KEYWORDS	LENGTH	PLANNING	STYLE	THEME
Demonstration	500	25.6	33.4	26.2	11.4	3.4	35.2	39.4	19.8	7.0	2.0	30.4
Evaluation	2,000	27.8	35.8	21.6	11.6	3.2	34.6	38.6	18.5	5.9	1.6	27.7

Table 2: Data statistics. Percentage of ambiguity categories assigned to each instance (# Additional Instructions), and percentage of instances assigned to each category (# Ambiguity Type).

tional instruction candidates $\hat{I}_c$ for all categories.[6]

Validation Finally, we validate the generated additional instruction candidates $\hat{I}_c$ and retain only those that meet the following criteria to obtain the final additional instructions I_c:

- **Clarity**: We assess whether the candidate $\hat{I}_c$ enhances the clarity of the initial instruction I_{init}. To facilitate scalability, we employ GPT-4 as an evaluator.[7] Only additional instructions that reduce ambiguity in the initial instruction are accepted under this criterion.
- **Utility**: We determine whether the generated candidate $\hat{I}_c$ helps generate output text that more closely aligns with the desired output. Specifically, we compare the ROUGE-L F1 scores of outputs generated before and after appending $\hat{I}_c$ to I_{init}, resulting in the refined instruction $\hat{I}_{\text{refined}} := I_{\text{init}} \oplus \hat{I}_c$. Using GPT-4, we generate 20 output samples for both I_{init} and $\hat{I}_{\text{refined}}$. We then perform statistical significance testing to evaluate whether the inclusion of $\hat{I}_c$ leads to output $\hat{y}$ that is significantly closer to the reference text y_{ref}. Only additional instructions demonstrating a significant improvement are retained.

4.2 Dataset Statistics

The AmbigSNI$_{\text{NLG}}$ dataset comprises 2,500 meticulously curated instances covering a wide range of NLG tasks as illustrated in Figure 3. The dataset is randomly split into 2,000 instances for evaluation and 500 for demonstrations. As shown in Table 2, approximately 75% of the instances present at least one category of task ambiguity in instructions, and around 35% contain multiple types of ambiguities.

Our dataset reveals a significant prevalence of categories such as CONTEXT, which encompasses background information about the task and necessary knowledge; KEYWORDS, which specifies words that should be included; and THEME which pertains to information about the content. This indicates that these aspects are particularly susceptible to ambiguity in NLG task instructions.

When analyzing the statistics for each task, Question generation is the most populated task, followed by Long-form QA, Sentence Compression, and Title Generation. Tasks requiring consideration of multiple topics—such as Question Generation, Title Generation, and Summarization—are predominantly associated with the THEME category. In contrast, tasks like Code to Text, designed to preserve content fidelity, exhibit a generally lower frequency of ambiguity categories except for CONTEXT. See examples and additional statistics in the Appendix.

5 Experiments

In this section, we empirically assess the effectiveness of our annotated additional instructions

[6] We minimized information leakage from the reference text by carefully designing the prompt. Details, analysis, and the prompt are described in the Appendix.

[7] Our in-house evaluation showed that GPT-4's assessment aligned with human judgments in 91% of cases.

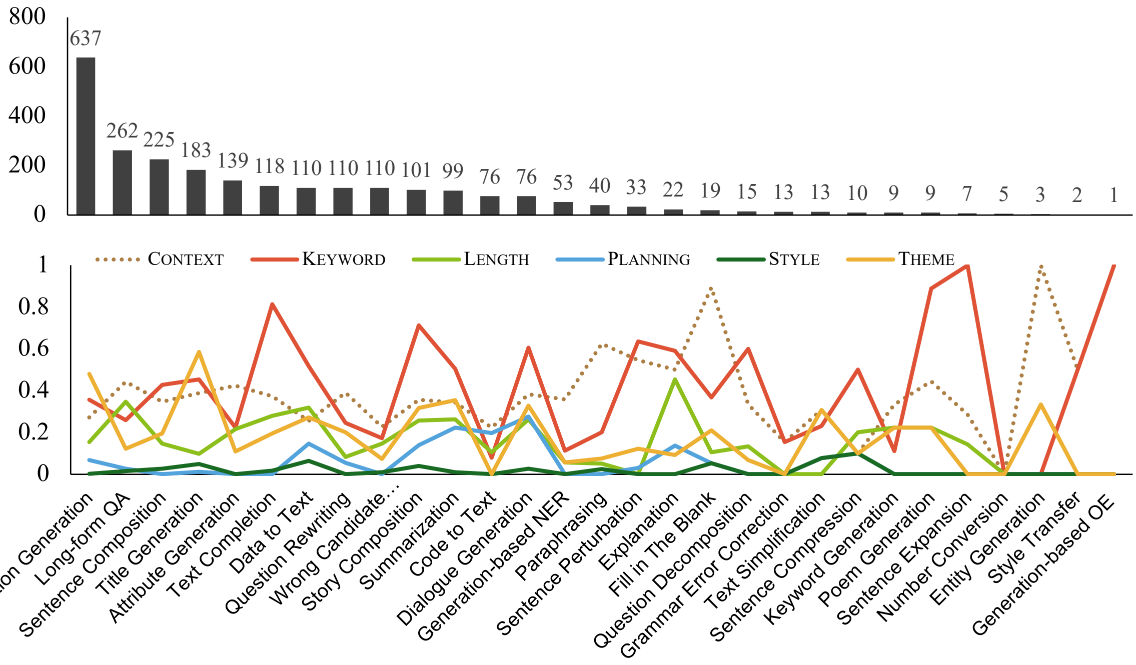


Figure 3: Distributions of the dataset. The upper bar graph displays the number of instances per task, while the lower line graph shows the proportion of instances assigned to ambiguous categories for each task.

presented in §4 in mitigating the *task ambiguity in instructions* defined in §2.2. Specifically, the goal of this section is to verify whether the model can utilize these additional instructions to mitigate ambiguities effectively.

5.1 Settings

Methods We evaluate two approaches for constructing refined instructions I_{refined}. The first approach, referred to as `Taxonomy`, involves concatenating our annotated additional instructions $\{I_{c_1}, \dots, I_{c_n}\}$ to the initial instruction I_{init}. Formally, the refined instruction is given by: $I_{\text{refined}} := I_{\text{init}} \oplus I_{c_1} \oplus \cdots \oplus I_{c_n}$.[8]

The second approach, termed `Generic`, constructs the refined instruction by appending a generic additional instruction I_{generic} to the initial instruction I_{init}: $I_{\text{refined}} := I_{\text{init}} \oplus I_{\text{generic}}$. This method serves as a baseline to evaluate the importance of our ambiguity taxonomy in mitigating ambiguity. Specifically, we employed the same generation pipeline described in §4, but used a generic template, 'Additional information: ____,' to create the additional instruction I_{generic}.[9]

Models We employ instruction fine-tuned LLaMA-2 (🦙 ; 7B) (Touvron et al., 2023), Mistral (▦ ; 7B) (Jiang et al., 2023) and Mixtral (▦ ; 8x7B) (Jiang et al., 2024) for open-sourced LLMs. Additionally, we utilize GPT-3.5 (֍ ; n/a) as a proprietary model.[10] To optimize space in our tables, each model is represented by an emoji along with its parameter size as an identifier.

Metrics To quantify the effect of task ambiguity mitigation in instructions on LLMs' responses, we measure two key aspects: `Alignment` and `Focus`.

For `Alignment`, we assess how well the LLMs generate responses that align with the user's expectations, as represented by the reference text y_{ref}, when additional instructions are provided. This is measured by the relative gains in reference-based metrics, specifically ROUGE-L and BERTScore.[11] We compare the outputs generated using only initial instructions I_{init} with those generated using the refined instructions I_{refined}.

For `Focus`,we evaluate the extent to which ambiguity mitigation narrows the output space $\mathcal{Y}_{\text{valid}}$. Our hypothesis is that effective ambiguity mitiga-

[8] We evaluated whether increasing instruction complexity by concatenating additional instructions affects the instruction-following capability of LLMs. Our experiment showed that this treatment does not impact their ability. See more details in the Appendix.

[9] For instance, in a summarization task, the generic additional instruction might be, "Please make sure to include the main points of the passage in your summary, even if they need to be slightly adjusted for conciseness."

[10] We exclude GPT-4 from our experiments as it serves as a data generator.

[11] `distilbert-base-uncased` with baseline re-scaling.

			Alignment		Focus
Model	# Param	Method	RL	BS	IntraRL
🦙	7B	Generic	0.44	1.13	-0.09
		Taxonomy	7.96	9.08	0.16
	7B	Generic	0.14	0.59	-0.08
		Taxonomy	6.83	7.78	0.25
	8x7B	Generic	0.46	1.14	-0.09
		Taxonomy	8.56	9.16	0.30
	n/a	Generic	1.47	1.69	0.16
		Taxonomy	15.02	13.62	0.66

Table 3: Relative gains in performance metrics for ambiguity mitigation. The table shows the relative gains in ROUGE-L (RL), BERTScore (BS), and Intra-RL for different models and methods.

tion will result in less diverse outputs. To quantify this, we compute the ROUGE-L score for each pair of sampled responses and average these scores, defined as the Intra-RL score (Shen et al., 2019; Iso et al., 2022): $\frac{2}{N(N-1)}\sum_{j<k}\text{ROUGE-L}(\hat{y}_j, \hat{y}_k)$, where N is the number of sampled responses. A higher Intra-RL score indicates that the sampled responses are more similar to each other, suggesting a narrower output space. We report the relative gains of Intra-RL scores to quantify the improvement, comparing outputs from initial instructions to those from refined instructions.

For these evaluations, we sample 20 responses per instance using a `temperature` setting of 1.0.

5.2 Results

Table 3 presents the relative gains in ROUGE-L, BERTScore, and Intra-RL metrics for ambiguity mitigation across different models. For `Alignment`, the results demonstrate substantial improvements in both ROUGE-L and BERTScore when using the `Taxonomy` compared to the `Generic`. Specifically, GPT-3.5 exhibits the highest gains, with a 15.02-point increase in ROUGE-L and a 13.62-point increase in BERTScore. This significant enhancement indicates that the `Taxonomy` effectively aligns generated responses with user expectations.

For `Focus`, the Intra-RL scores reveal that the `Taxonomy` consistently narrows the output space more effectively than the `Generic`. For instance, GPT-3.5 shows a significant gain of 0.66 in Intra-RL, while LLaMA-2 (7B) and Mistral (7B, 8x7B) also demonstrate positive gains of 0.16, 0.25, and 0.30, respectively. This suggests that the `Taxonomy` approach reduces variability in the generated outputs, focusing more closely on the desired content.

Overall, the results highlight that the `Taxonomy` outperforms the baseline without ambiguity mitigation and the `Generic` method. It not only improves alignment with user expectations but also effectively narrows the output space, thereby mitigating task ambiguity in instructions for LLMs.[12]

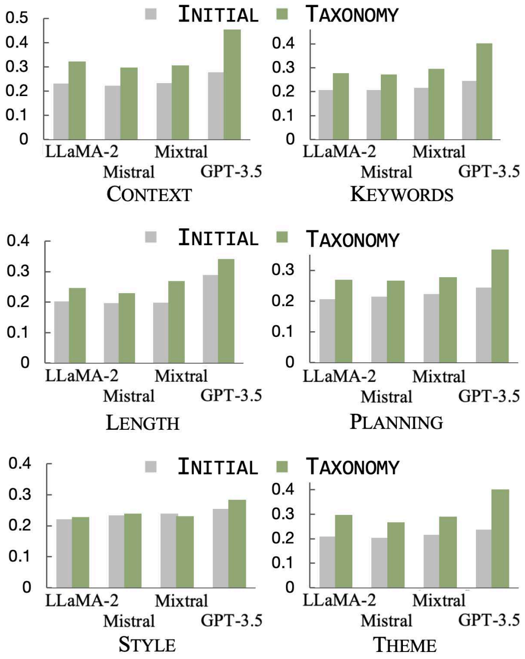


Figure 4: Mitigation results for each taxonomy.

Analysis We provide additional insights into the effectiveness of the `Taxonomy` method across different ambiguity categories and NLG tasks. Figure 4 illustrates the improvements in ROUGE-L scores across all categories and nearly all models. Notably, categories directly related to the content, such as CONTEXT, KEYWORDS, and THEME, show substantial improvements. This underscores the importance of an ambiguity taxonomy and explicit, category-specific additional instructions for effective ambiguity mitigation. Figure 5 presents the ROUGE-L score improvements for various NLG tasks, demonstrating that ambiguity mitigation consistently enhances performance regardless of the task type. This highlights the significance of ambiguity mitigation in fully leveraging the capabilities of LLMs for diverse NLG tasks. The comprehensive results are presented in Table 11.

6 Human-in-the-loop Ambiguity Mitigation

To assess the practical utility of our proposed ambiguity mitigation framework, we conducted a case

[12] Further analysis is provided in Appendix B.

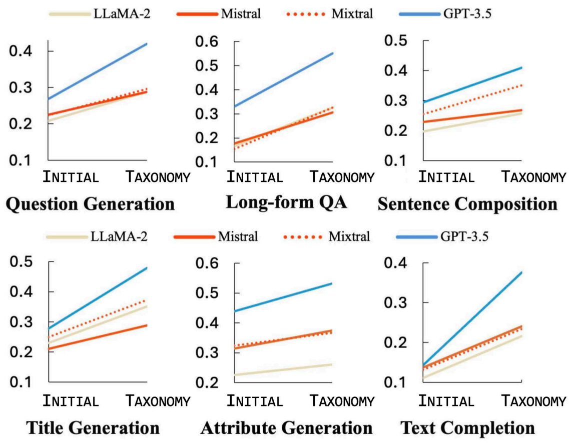
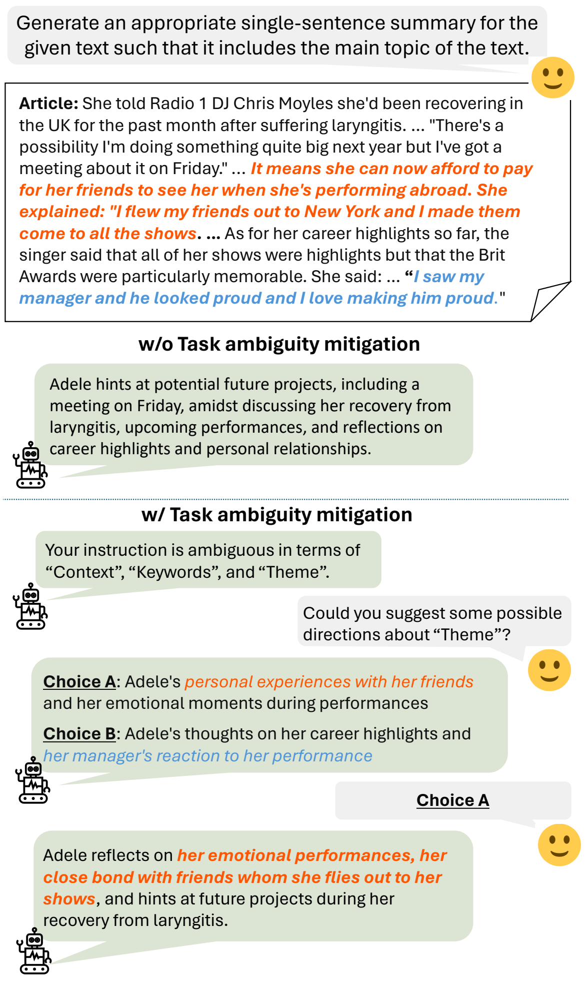


Figure 5: Mitigation results across the top-6 most frequent tasks in AmbigSNI$_{\text{NLG}}$. The figure demonstrates that ambiguity mitigation consistently enhances performance across different NLG tasks, as indicated by the ROUGE-L score improvements.



Generate an appropriate single-sentence summary for the given text such that it includes the main topic of the text.

Article: She told Radio 1 DJ Chris Moyles she'd been recovering in the UK for the past month after suffering laryngitis. ... "There's a possibility I'm doing something quite big next year but I've got a meeting about it on Friday." ... ***It means she can now afford to pay for her friends to see her when she's performing abroad. She explained: "I flew my friends out to New York and I made them come to all the shows*. ...** As for her career highlights so far, the singer said that all of her shows were highlights but that the Brit Awards were particularly memorable. She said: ... **"*I saw my manager and he looked proud and I love making him proud*.**"

w/o Task ambiguity mitigation

Adele hints at potential future projects, including a meeting on Friday, amidst discussing her recovery from laryngitis, upcoming performances, and reflections on career highlights and personal relationships.

w/ Task ambiguity mitigation

Your instruction is ambiguous in terms of "Context", "Keywords", and "Theme".

Could you suggest some possible directions about "Theme"?

Choice A: Adele's *personal experiences with her friends* and her emotional moments during performances

Choice B: Adele's thoughts on her career highlights and *her manager's reaction to her performance*

Choice A

Adele reflects on ***her emotional performances, her close bond with friends whom she flies out to her shows***, and hints at future projects during her recovery from laryngitis.

Figure 6: Example with pipeline mitigation.

study involving human interaction. This experiment aims to assess whether LLM-generated additional instructions can effectively guide the generation of desired outputs in real-world scenarios.

6.1 Experimental Design

As illustrated in Figure 6, our case study is designed to simulate real-world scenarios in which users engage with LLMs to clarify ambiguous instructions. The goal is to improve the alignment of the generated outputs with user intent through the following steps:

1. Given an initial instruction I_{init}, the LLM identifies a potential ambiguity c (§6.2) and suggests additional instructions $\{\hat{I}_c^1, \ldots, \hat{I}_c^N\}$ to address these ambiguities (§6.3).

2. The user then selects the most appropriate additional instruction provided by the LLM to mitigate the ambiguities in I_{init}.

3. Finally, the LLM generates the output based on the refined instruction I_{refined} (§6.4).

6.2 Identifying Ambiguity in Instructions

We begin by investigating the ability of LLMs to identify task ambiguity in instructions, framing this as a binary classification problem for each ambiguity category.

Settings Experiments were conducted in both zero-shot and in-context settings. In the in-context setting, we retrieved 8 similar examples from the demonstration set using `all-mpnet-base-v2` as the retriever and incorporated these examples along with their labels into the context provided to the LLMs. To address the imbalance in the distribution of ambiguity labels, we evaluated the models using True Positive Rate (TPR), True Negative Rate (TNR), and accuracy (Acc). Additionally, we used exact match accuracy (EM) to assess the overall success in identifying all ambiguity labels.

Results Table 4 illustrates that in zero-shot settings, all LLMs tended to classify instructions as ambiguous, resulting in high TPR but low TNR and consequently near-zero EM scores. However, with in-context demonstrations, all open-sourced LLMs exhibit a more balanced evaluation of ambiguity, leading to higher Acc and EM. This indicates that in-context demonstrations, rather than model size, play a crucial role in accurately identifying task ambiguity. Interestingly, GPT-3.5 did not follow this trend, implying it may prioritize its own decision over the influence of in-context demonstrations.

			Category Average			All
Model	#Param	ICL	TPR	TNR	Acc	EM
🦙	7B	✗	98.31	1.82	22.07	0.10
		✓	14.01	86.81	70.83	20.20
	7B	✗	99.93	0.15	21.27	0.00
		✓	55.01	49.31	50.15	13.30
	8x7B	✗	96.59	4.38	23.70	0.10
		✓	20.14	82.85	70.04	23.15
	n/a	✗	79.66	23.00	34.63	3.55
		✓	87.48	10.80	26.92	1.30

Table 4: Performance of ambiguity identification. The table shows the True Positive Rate (TPR), True Negative Rate (TNR), accuracy (Acc), and exact match accuracy (EM) for identifying task ambiguity across different models and settings.

			Relevance↑		Diversity↓
Model	#Param	Method	RL@10	Para@10	IntraRL
🦙	7B	sampling	0.183	0.487	0.308
		batch	0.230	0.469	0.314
	7B	sampling	0.251	0.523	0.322
		batch	0.229	0.449	0.346
	8x7B	sampling	0.363	0.615	0.456
		batch	0.379	0.615	0.387
	n/a	sampling	0.544	0.719	0.545
		batch	0.422	0.629	0.433

Table 5: Performance of instruction suggestions. Relevance is measured by the highest ROUGE-L score (RL@10) and semantic similarity (Para@10) with the reference instruction, while diversity is measured by the Intra-RL score among the candidates.

6.3 Suggesting Addition Instructions

We next evaluate the ability of LLMs to generate useful additional instructions for mitigating task ambiguity. Specifically, we investigate whether LLMs can suggest suitable options for an additional instruction I_c based on the identified ambiguity category c, allowing users to choose the most appropriate one.

Settings We employed templates specific to each ambiguity category to generate candidates by either sampling or batching N suggestions simultaneously. We framed this suggestion task as a recommendation problem, assessing the candidates based on their **Relevance** and **Diversity**. For **Relevance**, we measured the highest ROUGE-L score (RL@N) and semantic similarity (Para@N) between the generated candidates and the reference I_c in AmbigSNI$_{\text{NLG}}$. For **Diversity**, we calculated the Intra-RL score among the candidates to assess the variety of the suggestions.

Results Table 5 presents the efficacy of LLMs in suggesting additional instructions to mitigate ambiguity when $N = 10$. The results indicate that for LLaMA-2, Mistral, and Mixtral, generating more diverse outputs leads to higher surface-level and semantic similarity with the reference I_c, confirming the benefit of generating varied suggestions to address ambiguity. Conversely, for GPT-3.5, enhancing diversity through batch generation significantly decreases relevance, indicating that while GPT-3.5 excels at generating optimal additional instructions, forcing it to generate diverse outputs can impair this capability. This underscores the importance of tailoring generation settings to each model's strengths.

6.4 Generation with Ambiguity Mitigation

To assess the practical effectiveness of our ambiguity mitigation framework, we conducted a final evaluation using LLM-generated additional instructions. Human annotators manually selected the most appropriate additional instruction $\hat{I}_{c_i}$ from N options $\{\hat{I}_{c_i,j}\}_{j=1}^{N}$ generated in § 6.3. The selected additional instruction was intended to facilitate the more accurate generation of the reference text y_{ref}. We then appended the best additional instructions across all categories to the initial instruction I_{init}, forming the refined instruction $\hat{I}_{\text{refined}}$ used for the downstream NLG task.

Settings We utilized additional instruction options generated by GPT-3.5 through sampling, as it demonstrated superior performance in § 6.3. We randomly selected 100 test instances, resulting in a total of 2,140 additional instruction options. To evaluate the effectiveness of these refined instructions, we measured the similarity between the generated text $\hat{y}$ (produced using $\hat{I}_{\text{refined}}$) and the reference text y_{ref}, employing the ROUGE-L F1 score and BERTScore.

Results Incorporating LLM-generated additional instructions led to significant improvements: approximately 5.2-point increase in ROUGE-L (0.165 to $\underline{0.217}$) and a 4.6-point increase in BERTScore (0.273 to $\underline{0.319}$).[13] This demonstrates that LLM-generated instructions can significantly enhance the alignment of generated text with user expectations. Furthermore, we manually checked the outputs and found that in 94% of cases where

[13] The underline denotes significant gains over baseline at $p < 0.05$.

the quality of the output texts changed due to the additional instructions, the outputs more closely matched the reference texts. These findings confirm that our framework for mitigating task ambiguity is effective in practical settings, highlighting its potential for real-world applications.

7 Related Work

7.1 Ambiguity in NLP

Ambiguity has long been a fundamental challenge in NLP (Jurafsky, 1996; Carpuat and Wu, 2007), manifesting across a variety of tasks (Min et al., 2020; Pilault et al., 2023; Bhaskar et al., 2023; Liu et al., 2023a). In this study, we specifically focused on *task ambiguity* (Finn et al., 2018; Tamkin et al., 2022, 2023) that arises when a model faces unclear and incomplete instructions or data. Previous studies have addressed task ambiguities within the realm of natural language understanding (NLU) (Finn et al., 2018; Tamkin et al., 2022, 2023). However, these approaches are insufficient for the complex and diverse context of NLG tasks, where mitigating ambiguity often requires more nuanced, instance-specific strategies. To address this gap, we tackle task ambiguity across a wide range of NLG tasks.

7.2 Prompt Optimization

Our study can also be positioned within the scope of prompt optimization, including techniques such as prompt paraphrasing (Zhou et al., 2023b; Pryzant et al., 2023; Cho et al., 2023) and detailed instruction integration (Li et al., 2023; Bsharat et al., 2023; Zhou et al., 2023a; Wang et al., 2024). We align with the latter approach by incorporating additional instructions to mitigate ambiguity in the initial prompts. The primary distinction is that we uniquely focus on an *instance-level* prompt optimization *via a human-in-the-loop approach* for ambiguity mitigation, as opposed to the others' focus on optimizing a *dataset-level* prompts or generating them *automatically*.

8 Conclusion

We introduced AmbigNLG, a novel task designed to address the challenge of task ambiguity in instructions for NLG. We developed an ambiguity taxonomy that systematically categorizes types of ambiguities present in NLG instructions and proposed a method to refine initial instructions by providing clearer specifications. We also constructed AmbigSNI$_{\text{NLG}}$ dataset, comprising 2,500 annotated instances, to facilitate the AmbigNLG task.

Our comprehensive experiments with general LLMs demonstrated that our method significantly improves the alignment of generated text with user expectations. Furthermore, a case study involving real human interaction confirmed the practical utility of our approach. These findings underscore the critical importance of addressing task ambiguity to fully harness the capabilities of LLMs in NLG tasks, paving the way for more precise and effective natural language interactions.

Acknowledgements

We thank Yuki Arase from Tokyo Institute of Technology for her valuable feedback on this work. We are also thankful to Estevam Hruschka, Takuya Makino, and the other members of Megagon Labs for their insightful comments and suggestions.

Limitation

While our proposed method effectively mitigates ambiguities based on the predefined taxonomy observed in the dataset, it currently does not address ambiguities that fall outside these categories. Extending our approach to encompass additional types of ambiguities would require systematizing other ambiguity categories and verifying their effectiveness.

In this study, we did not implement mechanisms to handle situations where the provided additional instructions might not fully meet user requirements. Recognizing this, incorporating mechanisms for iterative user interaction to refine instructions could further enhance the effectiveness of our approach.

Moreover, when presenting multiple additional instructions to users, optimizing their selection through reranking could further enhance the effectiveness of the interaction. Developing methods to automatically select the more appropriate and promising additional instructions remains an open question. Addressing this challenge could significantly improve user experience and the overall efficacy of ambiguity mitigation strategies.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Adithya Bhaskar, Tushar Tomar, Ashutosh Sathe, and Sunita Sarawagi. 2023. Benchmarking and improving text-to-SQL generation under ambiguity. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7053–7074, Singapore. Association for Computational Linguistics.

Steven Bird. 2006. NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72, Sydney, Australia. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Sondos Mahmoud Bsharat, Aidar Myrzakhan, and Zhiqiang Shen. 2023. Principled instructions are all you need for questioning llama-1/2, gpt-3.5/4.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic. Association for Computational Linguistics.

Sukmin Cho, Soyeong Jeong, Jeong yeon Seo, and Jong Park. 2023. Discrete prompt optimization via constrained generation for zero-shot re-ranker. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 960–971, Toronto, Canada. Association for Computational Linguistics.

Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Re. 2022. Flashattention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*.

Budhaditya Deb, Ahmed Hassan Awadallah, and Guoqing Zheng. 2022. Boosting natural language generation from instructions with meta-learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6792–6808, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Boyang Li, Shafiq Joty, and Lidong Bing. 2023. Is GPT-3 a good data annotator? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11173–11195, Toronto, Canada. Association for Computational Linguistics.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Chelsea Finn, Kelvin Xu, and Sergey Levine. 2018. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Hayate Iso. 2024. AutoTemplate: A simple recipe for lexically constrained text generation. In *Proceedings of the 17th International Natural Language Generation Conference*, pages 1–12, Tokyo, Japan. Association for Computational Linguistics.

Hayate Iso, Chao Qiao, and Hang Li. 2020. Fact-based Text Editing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 171–182, Online. Association for Computational Linguistics.

Hayate Iso, Xiaolan Wang, Stefanos Angelidis, and Yoshihiko Suhara. 2022. Comparative opinion summarization via collaborative decoding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3307–3324, Dublin, Ireland. 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.

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, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts.

Daniel Jurafsky. 1996. A probabilistic model of lexical and syntactic access and disambiguation. *Cognitive science*, 20(2):137–194.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Zekun Li, Baolin Peng, Pengcheng He, Michel Galley, Jianfeng Gao, and Xifeng Yan. 2023. Guiding large language models via directional stimulus prompting. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Alisa Liu, Zhaofeng Wu, Julian Michael, Alane Suhr, Peter West, Alexander Koller, Swabha Swayamdipta, Noah Smith, and Yejin Choi. 2023a. We're afraid language models aren't modeling ambiguity. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 790–807, Singapore. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Yixin Liu, Budhaditya Deb, Milagro Teruel, Aaron Halfaker, Dragomir Radev, and Ahmed Hassan Awadallah. 2023c. On improving summarization factual consistency from natural language feedback. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15144–15161, Toronto, Canada. Association for Computational Linguistics.

David D. McDonald and James D. Pustejovsky. 1985. A computational theory of prose style for natural language generation. In *Second Conference of the European Chapter of the Association for Computational Linguistics*, Geneva, Switzerland. Association for Computational Linguistics.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online. Association for Computational Linguistics.

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.

Jonathan Pilault, Xavier Garcia, Arthur Bražinskas, and Orhan Firat. 2023. Interactive-chain-prompting: Ambiguity resolution for crosslingual conditional generation with interaction. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–483, Nusa Dua, Bali. Association for Computational Linguistics.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

David Reitter, Frank Keller, and Johanna D. Moore. 2006. Computational modelling of structural priming in dialogue. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 121–124, New York City, USA. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine

Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2019. Mixture models for diverse machine translation: Tricks of the trade. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5719–5728. PMLR.

Alex Tamkin, Kunal Handa, Avash Shrestha, and Noah Goodman. 2023. Task ambiguity in humans and language models. In *The Eleventh International Conference on Learning Representations*.

Alex Tamkin, Dat Nguyen, Salil Deshpande, Jesse Mu, and Noah Goodman. 2022. Active learning helps pretrained models learn the intended task. In *Advances in Neural Information Processing Systems*, volume 35, pages 28140–28153. Curran Associates, Inc.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P. Xing, and Zhiting Hu. 2024. Promptagent: Strategic planning with language models enables expert-level prompt optimization. In *The Twelfth International Conference on Learning Representations*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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*.

Haopeng Zhang, Hayate Iso, Sairam Gurajada, and Nikita Bhutani. 2024. XATU: A fine-grained instruction-based benchmark for explainable text updates. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 17739–17752, Torino, Italia. ELRA and ICCL.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023a. Instruction-following evaluation for large language models.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023b. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.

A Additional Details about Dataset Creation

A.1 Dataset Usage

The AmbigSNI$_{\text{NLG}}$ dataset, with its ambiguity taxonomy and additional instructions, provides a foundation for research aimed at developing more reliable, efficient, and user-friendly NLG applications by mitigating the task ambiguity in NLG instructions. Key uses of our dataset include:

Ambiguity Mitigation in NLG Tasks Indeed, by leveraging the taxonomy and additional instructions, developers and researchers can design systems that identify and mitigate ambiguities. This functionality is essential for generating more accurate and contextually relevant responses.

Instruction-Based NLG Model Training The dataset can be used to train models to interpret complex instructions that may contain ambiguities. This training helps models enhance their usability in real-world applications.

Request Clarification Model Development AmbigSNI$_{\text{NLG}}$ enables the development of models that can clarify users' requests when faced with ambiguous instructions. This functionality is vital for interactive systems that engage in dialogues with users to refine their requests, enhancing the overall effectiveness and user experience.

Benchmarking and Model Evaluation As a benchmark tool, the dataset enables an in-depth evaluation of how various NLG systems manage the task ambiguity in instructions. Researchers can use the provided taxonomy and annotations to compare how different models address ambiguities, allowing for a detailed assessment of nuanced aspects of model performance.

A.2 Preprocessing the SNI Benchmark

The SNI benchmark comprises a wide variety of datasets, including both NLG and NLU datasets. For this study, we extracted only the NLG datasets from the SNI. We began by using the list of NLG datasets provided by Deb et al. (2022). We then refined this list by applying the following rules to clearly differentiate between NLG and NLU datasets. A dataset qualifies as an NLG dataset only if it meets all the following criteria:

1. If the output text neither directly incorporates the input text nor the instruction.
2. If the output text consists of more than two words.
3. If the output is not composed solely of symbols or numbers.

After completing this process, we renamed certain task names to more accurately reflect their content for our study, as detailed below:

- Question answering → Long-form question answering (QA)
- Information extraction → Attribute Generation
- Named Entity Recognition → Generation-based Named Entity Recognition (NER)
- Keyword Tagging → Keyword Generation
- Overlap Extraction → Generation-based Overlap Extraction (OE)

A.3 Annotation Step in Curation

In the curation process in §4.1, we ensured the quality of the additional instructions through a three-step process:

1. An author crafted additional instructions for the sampled instances, following the same guidelines used to fine-tune GPT-3.5, as outlined in Table 12.
2. The same author then carefully refined these instructions, ensuring that:
   - The content remained consistently relevant
   - No explicit answers were included within the additional instructions
   - There was no content overlap with additional instructions for other ambiguity categories
   - There was no content overlap between the additional instructions and the initial instructions or input text
3. Other authors reviewed and revised the additional instructions as necessary.

A.4 Rule-based Annotation

Additional instructions for the KEYWORD and LENGTH categories can be derived solely from the output text based on predefined rules, without an LLM. The annotation process for each is as follows:

KEYWORD We utilize the lightweight unsupervised keyword extraction method Yake (Campos et al., 2020) to extract the Top-n most significant keywords or key phrases from the output text. These extracted keywords or key phrases are then used to fill the template 'Include ___ in your response.' However, selecting an excessively high value of n can result in an impractical setup. Therefore, we define n based on the output length, ensuring that only a reasonable number of keywords or key phrases are provided.

$$n = \max \left\{ m \,|\, m \leq 4, \sum_{i=1}^{m} w_i \leq 0.4 \cdot W \right\}$$

where W is the total word count in the output text and w_i is the word count in the i-th key phrase.

Length Using NLTK (Bird, 2006), we extract the word count n from the output text and fill in the template 'Answer with ___ words' accordingly. However, configuring an LLM to generate exactly n words is impractical. Instead of specifying an exact count, we define a range using the phrase 'a to b words.'

$$(a, b) = \left(\left\lfloor \frac{n}{10} \right\rfloor \times 10, \left(\left\lfloor \frac{n}{10} \right\rfloor + 1 \right) \times 10 \right)$$

In situations where n is 10 or less, we modify the template to use the phrase 'less than b words.'

A.5 Examples from AmbigSNI$_{\text{NLG}}$ dataset

Table 6 and 7 present the examples from the AmbigSNI$_{\text{NLG}}$ dataset, illustrating the instruction, input text, reference text, assigned ambiguity category, and the corresponding additional instruction for the category.

A.6 Further Statistics of Additional Instruction

Sequence Length We display the length distribution of additional instruction for each ambiguity category in Figure 7. The sequence length of the concatenated additional instructions (All), which encompass all assigned ambiguity categories, averages 49 words, with a maximum of 276 words. The length varies significantly depending on the assigned ambiguity category, tending to be longer when Context is included, as this category typically results in the longest sequence length.

Minimized Information Leakage We confirmed that information leakage of the reference text is minimized by enforcing a constraint on the prompt (in Table 12) to ensure that the answer itself is not included in the additional instruction $\hat{I}_c$. To validate this, we assessed the overlap between $\hat{I}c$ and y_{ref} using the ROUGE score, which resulted in a score of 0.177. This is notably lower than the ROUGE score of 0.229 between input text x and reference text y_{ref}, indicating the effectiveness of the constraint.

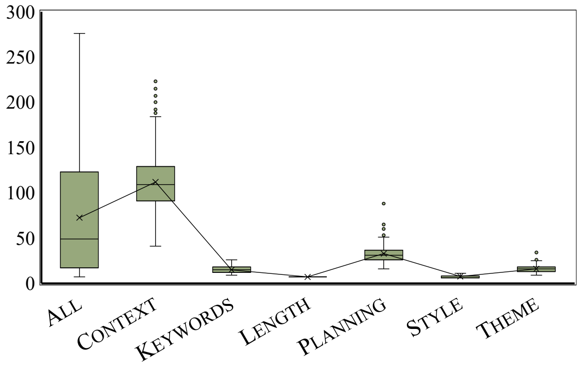


Figure 7: Length distribution of the additional instruction.

B Further Experimental Details

B.1 Computational Details

We performed all experiments to run on eight 80GB A100 GPUs. For the open-sourced LLMs, we used vLLM (Kwon et al., 2023), which implements a variety of efficiency tricks for the transformer model to make the LLMs' inference faster (Pope et al., 2023; Dao et al., 2022). For the proprietary LLMs, we used the official OpenAI library to call the API.

B.2 Results about Ambiguity Mitigation

Additional Cost by the Concatenation Our mitigation method involves augmenting the initial instruction with the additional instruction, which increases the sequence length. To quantify the cost, we use the OpenAI API as an example, which represents the highest-cost option in our experiments. Using the `gpt-3.5-turbo` model at \$0.0005 per 1,000 tokens, the average additional cost per instance is \$0.0000245, with a maximum of \$0.000138. For the more expensive `gpt-4-32k` model, priced at \$0.06 per 1,000 tokens, the average additional cost per instance rises to \$0.00294 and a maximum of \$0.01656. These results indicate that the proposed framework enhances performance while incurring only minimal additional costs.

Results about instruction following To determine whether additional instructions make instruction too complex for LLMs to follow, we evaluated

Instruction	In this task, we ask you convert a data table of restaurant descriptions into fluent natural-sounding English sentences. The input is a string of key-value pairs; the output should be a natural and grammatical English sentence containing all the information from the input.
Input	name[The Golden Palace], eatType[coffee shop], food[Indian], priceRange[moderate], customer rating[3 out of 5], area[city centre]
Reference	The Golden Palace is a coffee shop in city centre that serves Indian food. It has a customer rating of 3 out of 5 and moderately priced.
Ambiguity categories	Planning
Additional instruction	Please generate the output based on the following outline: 1. Description and location of The Golden Palace 2. Customer rating and pricing of The Golden Palace

Table 6: Example 1 (id: `task957-75dd6eba92a649ba81524c3a0594d57c`) from AmbigSNI$_{\text{NLG}}$ dataset. The input table contains multiple contents, making it ambiguous in the initial instructions how each content should be represented in the output. Therefore, an additional instruction regarding Planning was assigned to specify that the customer ratings and pricing should be explained after describing the restaurant's information.

Instruction	In this task, you are given an article. Your task is to summarize the article in a sentence.
Input	Aslef and RMT members are due to walk out for six days from 9 January. The Confederation of Passenger Transport (CPT) said bus operators from Cornwall to Northumberland were ready to send vehicles to the South East. Southern said it was still deciding what services might be offered. A CPT spokeswoman said: "We have had a very good response from quite a few members." It has sent Southern's parent company, the Go-Ahead Group, a list of operators including family-run firms which are ready to provide buses. Southern said it planned to announce on Wednesday what rail replacement services might be offered "to some commuters" but warned there would be no trains at all during the strike. Three weeks ago the government said officials were liaising with CPT "to determine how bus and coach operators can best assist with providing alternative transport". BBC South East understands the Army was asked before Christmas to prepare contingency plans for soldiers to drive buses. ...
Reference	Dozens of bus and coach companies across England have offered vehicles for rail replacement services during the next Southern train drivers' strike.
Ambiguity categories	Theme
Additional instruction	Primarily discuss the following theme: Provision of alternative transport during a train drivers' strike.

Table 7: Example 2 (id: `task1290-643d125a902345fca21b2c8a83ff4006`) from AmbigSNI$_{\text{NLG}}$ dataset. The input article includes multiple sub-themes, such as strike schedules, alternative transportation, and government collaboration, making it ambiguous which theme should be focused on in the summary. Therefore, an additional instruction regarding the Theme was assigned to specify focusing on alternative transportation.

Instruction	Llama-2	Mistral	Mixtral	GPT-3.5
I_{init}	3.87	4.41	4.46	4.12
I_{refined}	**4.15**	**4.59**	**4.70**	**4.43**

Table 8: Instruction following (IF) score.

the Instruction Following (IF) score for models both without mitigation (using the initial instructions I_{init}) and with mitigation (using the refined instructions I_{refined}). Similar to (Liu et al., 2023b), we employed GPT-4 as the evaluator, utilizing a five-point scale. We randomly selected 100 instances for this analysis. The results, shown in Table 8, indicate that the IF scores for I_{refined} consistently exceeded those for I_{init}. This suggests that our additional instructions do not overcomplicate the refined instruction. We hypothesize that the higher IF scores with the refined instructions are due to the clearer and more specific criteria they provide, which enhance the models' ability to follow instructions accurately.

B.3 Results about Ambiguous Category Identification

Overall Results We display the results for each taxonomy in §6.3 in Table 9.

B.4 Results about the Instruction Suggestion

Further Results In Section 6.3, we employed an approach that fills in templates when suggesting additional instructions. Here, for comparison, we examine the results of an open-ended approach where additional instructions are generated without using templates. Table 10 showcases that open-ended generation is more diverse because it doesn't follow a single template to generate suggestions, but generally less relevant than the fill-in-the-blank approach (Table 5). Therefore, we adopted the template-based approach in the main experiment.

C List of prompts

Prompts for annotation used in §4.1. For CONTEXT

```
{Category prompt in Table 12}

# Instruction
{instruction}

# Input text:
{input}

# Output text:
{output}

# Template:
```

Prompts for annotation used in §4.1. For THEME, PLANNING, STYLE

```
{Category prompt in Table 12}

# Task Category
{task category}

# Input text:
{input}

# Output text:
{output}

# Template:
```

Prompt for validation (**Clarity**) used in §4.1.

```
# Instruction
{instruction}

For the instruction above, please assess that
combining the additional instruction below with the
instruction either increases, decreases, or maintains
the ambiguity level in the instruction to lead the
precise generation of output text from the input text.
More specifically, focus on the aspect of '{ambiguity
category}' ({description}).
Answer with 'More ambiguous', 'Less ambiguous', or
'Unchanged'.

# Input text:
{input}

# Output text:
{output}

# additional instruction:
{additional instruction}

# Answer:
```

Prompt for validation (**Utility**) used in §4.1.

```
Below is an input text that provides further context,
paired with an instruction that describes a task.
Write a response that appropriately completes the
request.

# Input text:
{input}

# Instruction:
{instruction}

# Response:
```

Prompt for executing downstream tasks used in §5.2 and §6.4

```
Below is an input text that provides further context,
paired with an instruction that describes a task.
Provide a direct response that appropriately completes
the request without additional explanations or details.

# Input text:
{input}

# Instruction:
{instruction}

# Response:
```

Model	#Param	ICL	Context TPR	Context TNR	Context B-Acc	Context Acc	Keywords TPR	Keywords TNR	Keywords B-Acc	Keywords Acc	Length TPR	Length TNR	Length B-Acc	Length Acc
🦙	7B	✗	98.12	2.45	50.29	35.60	97.15	1.87	49.51	38.70	98.65	1.84	50.24	19.75
	7B	✓	12.70	88.75	50.73	62.40	10.09	85.25	47.67	56.20	12.97	83.87	48.42	70.75
	7B	✗	99.86	0.23	50.04	34.75	100.00	0.08	50.04	38.70	99.73	0.12	49.93	18.55
	7B	✓	55.41	52.95	54.18	53.80	57.44	51.75	54.60	53.95	55.68	49.69	52.68	50.80
	8x7B	✗	90.04	11.17	50.61	38.50	98.97	0.90	49.93	38.80	98.11	4.05	51.08	21.45
	8x7B	✓	19.19	84.62	51.91	61.95	25.36	79.22	52.29	58.40	17.30	85.09	51.19	72.55
	n/a	✗	68.83	35.58	52.20	47.10	80.47	23.96	52.21	45.80	84.32	21.10	52.71	32.80
	n/a	✓	84.70	17.83	51.27	41.00	89.39	8.96	49.18	40.05	90.54	7.24	48.89	22.65

Model	#Param	ICL	Planning TPR	Planning TNR	Planning B-Acc	Planning Acc	Style TPR	Style TNR	Style B-Acc	Style Acc	Theme TPR	Theme TNR	Theme B-Acc	Theme Acc
🦙	7B	✗	99.15	1.49	50.32	7.25	100.00	1.63	50.81	3.15	96.75	1.66	49.21	28.00
	7B	✓	21.19	88.10	54.64	84.15	19.35	86.24	52.80	85.20	7.76	88.66	48.21	66.25
	7B	✗	100.00	0.32	50.16	6.20	100.00	0.10	50.05	1.65	100.00	0.07	50.03	27.75
	7B	✓	48.31	47.02	47.66	47.10	67.74	50.89	59.32	51.15	45.49	43.57	44.53	44.10
	8x7B	✗	94.07	4.14	49.11	9.45	100.00	3.30	51.65	4.80	98.38	2.70	50.54	29.20
	8x7B	✓	11.86	84.96	48.41	80.65	25.81	85.68	55.74	84.75	21.30	77.52	49.41	61.95
	-	✗	76.27	20.56	48.42	23.85	74.19	17.17	45.68	18.05	93.86	19.64	56.75	40.20
	-	✓	85.59	9.78	47.69	14.25	87.10	10.87	48.98	12.05	87.55	10.10	48.82	31.55

Table 9: Overall results of category identification in §6.2.

			Relevance↑		Diversity↓
Model	#Param	Method	RL@10	Para@10	IntraRL
🦙	7B	sampling	0.128	0.454	0.313
		batch	0.165	0.440	0.306
	7B	sampling	0.152	0.455	0.313
		batch	0.171	0.400	0.345
	8x7B	sampling	0.183	0.516	0.370
		batch	0.215	0.517	0.284
	n/a	sampling	0.216	0.544	0.384
		batch	0.201	0.508	0.286

Table 10: Instruction suggestions performance generated in a open-ended manner.

Prompt for ambiguity identification used in §6.2

```
Your task involves identifying the category of
ambiguity in the given instruction to generate output
text from the given input text.
Ambiguity in instruction means that there are several
possible output texts from the single input text.
On the other hand, when the ambiguity is clarified,
the task becomes straightforward, leading to a nearly
single output.
Here are the available categories: {category list}.

{task_definition in Table 13}

If there are multiple ambiguities, please provide your
answer as a comma-separated list.

# Instruction:
{instruction}

# Input text:
{input}

# Response:
```

Prompt for instruction suggestion used in §6.3

```
To resolve the specified ambiguity in the instruction,
provide an additional specific instruction by
infilling the provided template. Ensure this added
information aligns with the primary objective of the
task, supports understanding of complex concepts, or
aids in narrowing down the scope to generate more
precise responses.

# Input Text:
{input_text}

# Instruction:
{instruction}

# Ambiguity to Resolve:
{ambiguity_category}: {ambiguity_definition}

# Template to Infill:
{template}

# Additional Instruction:
```

Tasks	CONTEXT	KEYWORDS	LENGTH	PLANNING	STYLE	THEME
Question Generation	0.272	0.356	0.154	0.068	0.002	0.480
Long-form QA	0.443	0.260	0.347	0.027	0.015	0.122
Sentence Composition	0.347	0.427	0.147	-	0.027	0.196
Title Generation	0.388	0.454	0.098	0.011	0.049	0.585
Attribute Generation	0.424	0.223	0.216	-	-	0.108
Text Completion	0.373	0.814	0.280	-	0.017	0.195
Data to Text	0.255	0.518	0.318	0.145	0.064	0.273
Question Rewriting	0.391	0.245	0.082	0.055	-	0.200
Wrong Candidate Generation	0.227	0.173	0.145	-	0.009	0.073
Story Composition	0.356	0.713	0.257	0.139	0.040	0.317
Summarization	0.343	0.505	0.263	0.222	0.010	0.354
Code to Text	0.224	0.079	0.105	0.197	-	-
Dialogue Generation	0.382	0.605	0.263	0.276	0.026	0.329
Generation-based NER	0.358	0.113	0.057	-	-	0.057
Paraphrasing	0.625	0.200	0.050	-	0.025	0.075
Sentence Perturbation	0.545	0.636	-	0.030	-	0.121
Explanation	0.500	0.591	0.455	0.136	-	0.091
Fill in The Blank	0.895	0.368	0.105	0.053	0.053	0.211
Question Decomposition	0.333	0.600	0.133	-	-	0.067
Grammar Error Correction	0.154	0.154	-	-	-	-
Text Simplification	0.308	0.231	-	0.077	0.077	0.308
Sentence Compression	0.100	0.500	0.200	0.100	0.100	0.100
Keyword Generation	0.333	0.111	0.222	-	-	0.222
Poem Generation	0.444	0.889	0.222	-	-	0.222
Sentence Expansion	0.286	1.000	0.143	-	-	-
Number Conversion	-	-	-	-	-	-
Entity Generation	1.000	-	-	-	-	0.333
Style Transfer	0.500	0.500	-	-	-	-
Generation-based OE	1.000	1.000	-	-	-	-

Table 11: Overall results of ambiguity mitigation across all tasks in §5.2. '-' indicates that the category is not assigned to the instances in the task.

G-Eval Prompt for instruction following evaluation used in §5.2

```
Below is an instruction for evaluating the
instruction-following ability of a language model
in the context of generating text based on specific
instructions. The evaluation ranges from 1 to 5, with 1
being the lowest and 5 the highest in terms of accuracy
and adherence to the given instruction. If there are
parts of the task instructions enclosed in asterisks
(*), please focus your evaluation particularly on
whether it adheres to those highlighted sections.

# Evaluation Criteria:
1. The output is unrelated to the given instruction.
2. The output vaguely relates to the instruction but
misses key elements.
3. The output is somewhat accurate but lacks detail or
has minor inaccuracies.
4. The output is accurate and detailed, with only
negligible issues.
5. The output perfectly matches the instruction with
high accuracy and detail.

# Instruction:
{instruction}
{additional instruction}

# Input Text:
{input text}

# Output Text:
{output text}

# Evaluation Form (scores ONLY):
```

System Message	
	You are an AI assistant addressing various ambiguities in NLP task instructions. Your role involves complementing incomplete information by filling in the blanks within the provided template. The template you have filled in is used as the additional instruction.
User Message	
Taxonomy	Prompt
CONTEXT	Please identify what additional context, such as background or external knowledge, will encourage the accurate generation from input to output text. Subsequently, write a concise paragraph containing the required context and other related content. Fill in the blank of the following template: ’Additional context: paragraph’. Ensure that it’s not clear which part of the paragraph corresponds to the output text. Please answer with the additional context needed to solve the task, not the solution to the task itself.
KEYWORDS	-
LENGTH	-
PLANNING	Please describe the output text structure by listing a concise topic for each sentence. Fill in the blank of the following template: ’Please generate the output based on the following outline: 1. topic1 2. topic2 ...’. Ensure that the number of items in the list matches the number of sentences in the output text. Make sure the response is brief and generalized, not detailed.
STYLE	Please select the writing style of the output text from the following options: descriptive, expository, narrative, persuasive, directive, conversational, technical, journalistic, review, poetic, formal, informal, optimistic, assertive, dramatic, humorous, sad, passive-aggressive, worried, friendly, curious, encouraging, surprised, cooperative. Fill in the blank of the following template: ’Write in a style style.’. You are allowed to select multiple styles if necessary. If none of the styles align with the text, please respond with ’neutral’
THEME	Please identify the single, most dominant content of the output text and provide a clear and succinct description of it. Fill in the blank of the following template: ’Primarily discuss the following theme: theme’. Make sure the response is brief and generalized, not detailed. Concentrate on the theme of the output text, rather than on the input text, instruction, or the overall task. The reply may contain hints of the output text, but should refrain from encapsulating its full content.

Table 12: Category prompts for fill-in-the-blank in dataset creation. (For KEYWORDS and LENGTH, we adopted the rule-based annotation as described in §A.4.)

Taxonomy	Task Definition
LENGTH	Length: Opt for this category if the instruction does not provide specifics about the desired length of the output, whether in terms of words or sentences. Clearing this ambiguity will lead to a more precise length output.
KEYWORDS	Keyword: Select this category if the instruction does not mention specific keywords to be used in the output text. Resolving this ambiguity will ensure that the necessary keywords are incorporated in the output.
CONTEXT	Context: Choose this category if the instruction lacks the required context information, such as background or external knowledge crucial for task completion. Resolving this ambiguity will provide the crucial context for the task.
THEME	Theme: Choose this category if the instruction does not clearly define the specific theme to be discussed in the output text. Clearing this ambiguity will provide a clear direction for the output.
PLANNING	Plan: Select this category if the instructions doesn't provide guidance on content planning for the output document. Resolving this ambiguity will result in the desired structured output.
STYLE	Style: Choose this category if the instruction does not specify the style of the output text. Clearing this ambiguity will ensure that the output aligns with the desired style.
NONE	None: Choose this category if the instructions are clear, define all aspects of the task well, and lead to a nearly single output.

Table 13: Prompt for ambiguity identification used in §6.2.