

Reasoning about Intent for Ambiguous Requests

Irina Saparina & Mirella Lapata

School of Informatics,

University of Edinburgh

Edinburgh, UK

i.saparina@sms.ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

Large language models often respond to ambiguous requests by *implicitly* committing to one interpretation, frustrating users and creating safety risks when that interpretation is wrong. We propose generating a single structured response that enumerates the different ways an ambiguous request can be interpreted, each coupled with a corresponding answer. Our models are trained with reinforcement learning using a dual reward objective: recall on ambiguous inputs to maximise coverage of valid interpretations, and precision on unambiguous ones to suppress spurious alternatives. Training requires only multiple valid answers per input as supervision, no clarification questions or explicit interpretations are needed. Experiments on conversational question answering and semantic parsing demonstrate that our method achieves higher coverage of valid answers than baseline approaches. Human evaluation confirms that predicted interpretations are meaningful and explain their corresponding answers. Our approach promotes transparency with *explicit* interpretations, avoids additional interaction rounds by producing all interpretations in a single generation, and supports downstream applications through its structured output format.

1 Introduction

Ambiguity poses a fundamental challenge for large language models (LLMs). Users routinely omit details either for the sake of brevity or unintentionally (Zipf, 1949; Piantadosi et al., 2012), creating requests with multiple plausible interpretations (see the examples in Table 1). LLMs often respond with a single answer, implicitly committing to one interpretation without acknowledging alternatives. LLMs can strategically exploit ambiguities, e.g., by deliberately misinterpreting scalar quantifiers or bracketing in logical statements to advance their own goals over user intentions (Choi et al., 2025). Ambiguity also presents a challenge for reasoning models, which can expend thousands of tokens on incorrect interpretations, simultaneously wasting computational resources and reinforcing the original error.

Prior work has explored different strategies for handling ambiguous inputs, each with distinct trade-offs (Tanjim et al., 2025). Clarification questions resolve ambiguity, but require additional interaction rounds (see Table 1). An alternative is to generate a longer answer that covers multiple interpretations in a single response (Stelmakh et al., 2022). Outputs in this case tend to be verbose, lacking explicit interpretations (see the examples in Table 1). As a result, the answers are difficult to parse for downstream applications and potentially overwhelming for users. Moreover, for certain ambiguity types (e.g., attachment ambiguity; see Table 1, right), interpretations may be mutually contradictory. In domains like SQL generation, there may be no natural way to express a unified response (e.g., for interpretations requiring different table structures).

In this work, we propose a structured alternative, namely generating multiple plausible answers, each paired with an explicit interpretation (Table 1, bottom). This format requires a single generation step (rather than a resolving dialogue or a multi-stage pipeline) while producing outputs that are easy to parse and use for downstream applications, e.g., in agentic systems where branching across interpretations can be beneficial. Moreover, it

Conversational Question Answering	Text-to-SQL Parsing
The first states to hold power in Silesia were Greater Moravia in the 9th c. and Bohemia in the 10th c. Later, Silesia became a part of the early Polish state and in the 14th c., the Holy Roman Empire... – When did Bohemia rule there? – 10th c. – What other state ruled it? <i>Clarification Question</i> – Do you mean what state ruled it in the 9th c? – No, I mean what state ruled it in the 14th c. – The Holy Roman Empire <i>Long-form Answer</i> Greater Moravia ruled in the 9th c., and later the Polish state, and the Holy Roman Empire. <i>Interpretation-Answer Pairs (Ours)</i> - Other state refers to the predecessor of Bohemia Greater Moravia - Other state refers to the successor of Bohemia The early Polish state - Other state refers to the 14th-century ruler The Holy Roman Empire	<pre>CREATE TABLE Films(id INTEGER, title TEXT, budget REAL); CREATE TABLE Genres(id INTEGER, genre TEXT); CREATE TABLE Film_Genre ...</pre> Show horror films and thrillers with a 20M budget <i>Clarification Question</i> – Does the budget apply to both genres? – Only to thrillers. – SELECT title ... WHERE genre = 'Horror' OR genre = 'Thriller' AND budget = 20; <i>Long-form Answer</i> N/A <i>Interpretation-Answer Pairs (Ours)</i> - Budget applies to both genres SELECT title ... WHERE (genre = 'Horror' OR genre = 'Thriller') AND budget = 20 - Budget applies only to thrillers SELECT title ... WHERE genre = 'Horror' OR genre = 'Thriller' AND budget = 20;

Table 1: Approaches to handling ambiguity. Examples (simplified for the sake of brevity) are from the Abg-CoQA (Guo et al., 2021) and Ambrosia (Saparin & Lapata, 2024) test sets, illustrating time-dependent ambiguity (left) and attachment ambiguity (right). Inputs and ground-truth answers are shown in blue, ambiguous questions and elements that create ambiguity are highlighted in red. Interaction rounds following clarification questions are shown in gray. Our approach pairs each answer with an explicit interpretation (shown in orange) of the ambiguous question.

extends beyond textual answers to other forms of output, e.g., code, while allowing users to quickly identify their intended answer. Recognizing that ambiguity relates to the broader problem of pluralistic alignment (i.e., generating responses that reflect diverse perspectives and interpretations; Sorensen et al. 2024), our approach promotes *overt pluralism* by predicting multiple interpretations within one answer. Methods embracing *distributional pluralism* rely on parallel samples and often struggle to produce substantially different outputs (Zhang et al., 2025c). Instead, we generate interpretations sequentially within one response (where subsequent interpretations are conditioned on previous ones), without changes to sampling or confidence calibration.

Teaching models to handle ambiguity is challenging due to the lack of suitable supervision. While ambiguity is pervasive in natural language, humans can communicate without clarifying their intent. The most readily available supervision signal comes from annotator disagreement, which manifests as multiple answers for a single instance. Other forms of annotation, such as clarifications or explicit interpretations, are expensive and difficult to obtain at scale. Building on the success of reinforcement learning under sparse supervision (Shao et al., 2024), we frame ambiguity resolution as reasoning about *what* to answer rather than *how*: we encourage the model to reason about user intent, entertaining different interpretations of the input before committing to an answer. We train a reasoning model through reinforcement learning with specialized reward functions that only require answer annotations as supervision. For ambiguous inputs, we optimize for *recall* to maximize the coverage of interpretations. For unambiguous inputs, we optimize for *precision* to avoid generating spurious alternatives. This dual objective teaches the model to enumerate interpretations when necessary *and* to recognize when a question has a single intent.

We evaluate our approach on both verifiable and non-verifiable tasks, including text-to-SQL generation and conversational question answering. Results show that our model effectively

handles unambiguous and ambiguous cases, generalizing to out-of-distribution data while generating diverse, plausible answers with high coverage and meaningful interpretations. By teaching models to reason about user intent before answering, we move toward more robust, user-aligned LLMs that embrace rather than obscure ambiguity.

2 Related Work

Ambiguity in Question Answering A growing body of work recognises that many questions admit multiple valid answers, each corresponding to a distinct interpretation (see [Tanjim et al. \(2025\)](#) for an overview). AmbiQA ([Min et al., 2020](#)) was one of the first large-scale benchmarks to explicitly annotate multiple answers for individual questions, expanding on NaturalQuestions ([Kwiatkowski et al., 2019](#)). The open-domain QA setting has prompted numerous retrieval-augmented generation techniques (e.g., [Min et al., 2021](#)). ASQA ([Stelmakh et al., 2022](#)) extends AmbiQA by requiring long-form answers that cover multiple interpretations in a single response; while intuitive, this format lacks explicit structure, complicating both evaluation and downstream use. [Amplayo et al. \(2023\)](#) infer multiple interpretations and then generate long-form answers, though they rely on prompting only. [Guo et al. \(2021\)](#) use clarification questions as a disambiguation strategy in conversational QA, and more recently [Chen et al. \(2025\)](#) introduce a preference optimization method that generates clarification questions for ambiguous instances and direct answers for unambiguous ones. [Zhang et al. \(2025b\)](#) train models to ask clarifying questions by modelling future conversation turns, while concurrent work learns steerable clarification policies via collaborative self-play ([Berant et al., 2025](#)). In contrast, our approach produces structured interpretation-answer pairs and handles both cases in a *single* turn, without requiring additional interaction rounds.

Ambiguity in Semantic Parsing Ambiguity also presents challenges for semantic parsing across a range of settings ([Rasmussen & Schuler, 2020](#); [Li et al., 2023](#); [Mu et al., 2024](#); [Stengel-Eskin et al., 2024](#)). Standard datasets often provide only a single interpretation when multiple queries are equally plausible ([Pourreza & Rafiei, 2023](#); [Floratos et al., 2024](#)), motivating benchmarks that include multiple SQL queries per input ([Bhaskar et al., 2023](#); [Saparina & Lapata, 2024](#)) or interactive scenarios with clarification questions ([Dong et al., 2025](#); [Huo et al., 2025](#)). Proposed solutions include counterfactual examples ([Wang et al., 2023](#)), specialized decoding strategies ([Bhaskar et al., 2023](#)), generating interpretations and then answers in two stages ([Saparina & Lapata, 2025](#)), and interactive clarification ([Ding et al., 2025](#)). Unlike multi-stage approaches, our method generates all plausible interpretations and corresponding SQL queries in one response.

Diversity in Model Output Our work also relates to efforts to improve output diversity in LLMs, which have been shown to exhibit narrow behaviours and strong biases even in tasks where many valid responses exist ([Saparina & Lapata, 2024](#); [West & Potts, 2025](#); [Zhang et al., 2025c](#); [Troshin et al., 2025](#)). Post-training approaches have been proposed to shift model outputs from peaked to more diffuse distributions ([Zhang et al., 2024](#); [Sorensen et al., 2025](#)), including concurrent work that trains models to generate diverse answers for questions admitting multiple correct ones ([Puri et al., 2026](#)). [Zhang et al. \(2025a\)](#) improve diversity by enumerating possible answers with explicit probability estimates via prompting. Our approach pursues diversity through a different lens: it targets semantic ambiguity, where interpretations lead to fundamentally different answers, and pairs each answer with an explicit interpretation rather than a probability estimate, making the source of diversity explicit and actionable.

3 Reasoning about User Intent

3.1 Problem Formulation

A question is ambiguous if it admits multiple distinct interpretations that lead to different answers. Given some input consisting of optional context c (e.g., a passage in conversa-

tional QA or a database schema in text-to-SQL generation; see Table 1) and a question q , the model must produce a set of interpretation-answer pairs $\{(i_1, a_1), \dots, (i_m, a_m)\}$. Each interpretation i_k explicitly disambiguates the question by specifying what is being asked, while a_k is the corresponding answer for interpretation i_k . For example, “Other state refers to the predecessor of Bohemia” in Table 1 is one possible interpretation of the ambiguous question “What other state ruled it?” which corresponds to the answer “Greater Moravia”.

We expect a maximum of five interpretation-answer pairs, which we find represents the majority of ambiguity cases in existing datasets (Section 4.1). In theory, questions could have more interpretations, but at that scale other approaches such as clarification questions may be more appropriate than enumeration. In practice, the questions are often grounded in specific contexts, which naturally constrain the space of plausible interpretations. For unambiguous cases, we expect a single interpretation-answer pair, with the interpretation being identical to the original question or a paraphrase that makes the intent even more explicit.

3.2 Model Training

Our model is trained on examples annotated with gold answers $\mathcal{A} = \{\hat{a}_1, \dots, \hat{a}_n\}$, where $n = 1$ for unambiguous and $n > 1$ for ambiguous questions. These annotations are easier to obtain compared to interpretations or clarification questions; for example, they emerge naturally when multiple annotators disagree on the answers they provide.

We instruct the model to first analyze the input for potential ambiguities, verbalizing its reasoning as a chain of thought, and then generate structured output consisting of interpretation-answer pairs. We design a task-agnostic prompt template that can be easily adapted to different tasks by only modifying the context format and task-specific instructions (e.g., answer style constraints). The prompt is given in Appendix A. We generate interpretation-answer pairs sequentially. Thanks to the autoregressive nature of language models, each subsequent interpretation is conditioned on previously generated interpretations and answers. This encourages diversity across interpretations (Zhang et al., 2025c; Troshin et al., 2025; Zhang et al., 2025a) as well as semantic alignment between interpretations and their corresponding answers. We confirm the diversity effect in Appendix H.2, where we compare against parallel sampling.

We train our model using DAPO (Decoupled Clip and Dynamic Sampling Policy Optimization Yu et al. 2025), a reinforcement learning algorithm based on GRPO (Group Relative Policy Optimization; Shao et al. 2024). DAPO extends GRPO with decoupled clipping parameters that enhance exploration and dynamic sampling that filters out prompts with uniform outcomes to maintain effective gradients (see Appendix B for the formulas).

We design **reward functions** to optimize different objectives for ambiguous versus unambiguous inputs. We extract the set of predicted answers $\mathcal{P}$ from the model’s output and compare it against the gold answer set $\mathcal{A}$, where all answers are distinct by construction. For each predicted answer $p \in \mathcal{P}$ and gold answer $\hat{a} \in \mathcal{A}$ we compute a **similarity** score $\text{sim}(p, \hat{a})$ where the specific type of similarity function is task dependent.

For tasks like text-to-SQL whose output can be objectively verified, $\text{sim}(p, \hat{a}) = 1$ if execution results are the same, and $\text{sim}(p, \hat{a}) = 0$ otherwise. For tasks like question answering whose outputs are harder to verify, we use soft similarity metrics. In experiments, we adopt F1 score based on word overlap, however, any other word similarity function could be used. Let $S_{ij} = \text{sim}(p_i, \hat{a}_j)$ denote a matching matrix, and $\mathcal{M}$ the set of one-to-one matches between predicted and gold answers.

We compute optimal assignments between $\mathcal{P}$ and $\mathcal{A}$ as a maximum weighted bipartite matching problem using the Hungarian algorithm ¹:

$$M^* = \operatorname{argmax}_{M \in \mathcal{M}} \sum_{(i,j) \in M} S_{ij}.$$

¹We use the Hungarian algorithm as a general-purpose solution. With a maximum of five interpretations, brute-force optimization is also tractable.

For verifiable tasks with binary similarity (e.g., text-to-SQL), M^* reduces to exact match.

For ambiguous questions ($|\mathcal{A}| > 1$), our reward function is **recall**:

$$R_{\text{recall}} = \frac{1}{|\mathcal{A}|} \sum_{(p_i, \hat{a}_j) \in M^*} \text{sim}(p_i, \hat{a}_j)$$

which measures how many distinct valid interpretations are covered by the model, rewarding more complete responses.

For unambiguous questions ($|\mathcal{A}| = 1$), our primary objective is **precision** which penalizes generating spurious interpretations:

$$R_{\text{precision}} = \frac{1}{|\mathcal{P}|} \sum_{(p_i, \hat{a}_j) \in M^*} \text{sim}(p_i, \hat{a}_j)$$

4 Experimental Setup

4.1 Datasets and Evaluation Metrics

The only requirement for training our model is a dataset with multiple valid answers per input. We evaluate on three datasets representing different tasks and evaluation scenarios. All datasets include contextual information (e.g., passages or databases), making ambiguity resolution challenging and realistic as models must validate their interpretations against the provided context. Appendix C describes the types of ambiguity present in the datasets.

Abg-CoQA (Guo et al., 2021) is a conversational question answering dataset with non-verifiable outputs. The input consists of a passage, one or two previous conversation turns, and a question that can be ambiguous, leading to different answers. The dataset provides clarification questions but does not include explicit interpretations. It covers different types of ambiguity, the majority being coreference, and contains 741 ambiguous and 5,727 unambiguous training examples and 123 ambiguous and 932 unambiguous test examples. Ambiguous questions have 2.21 answers on average, with a maximum of 4.

Guo et al. (2021) measure similarity between predicted and gold answers using word overlap (macro-averaged F1). To account for different wordings within the same interpretation, each prediction is compared against three gold answers, and the maximum score is taken; these scores are then averaged across gold answers. This metric favors coverage over precision because Abg-CoQA treats disambiguation and answer generation as separate tasks. For a more comprehensive evaluation, we also use an LLM-based judge (Qwen3 4B Instruct; Yang et al., 2025) to assess recall (the proportion of gold answers covered by the predictions), precision (the proportion of predictions that match a gold answer), and full coverage (the percentage of instances where all gold answers are recovered). For unambiguous questions, there is only one gold answer, so recall and full coverage are equivalent and we report recall in these cases. Details on our prompt and judge model selection are in Appendix D.

Ambrosia (Saparina & Lapata, 2024) is a text-to-SQL dataset, with verifiable outputs, where some user questions are ambiguous and can lead to different SQL queries and execution results. The dataset showcases multiple ambiguities including scope and attachment ambiguities and comes with gold interpretations for ambiguous questions. We compare query execution results and report recall, precision, and full coverage. We use the data split from Saparina & Lapata (2025), containing 1,026 ambiguous and 2,380 unambiguous training examples and 131 ambiguous and 309 unambiguous test examples. Ambiguous questions have 2.3 answers on average, with a maximum of 3.

AmbiQT (Bhaskar et al., 2023) is a text-to-SQL benchmark with ambiguous questions that we use to evaluate out-of-domain generalization, as models trained on Ambrosia are evaluated on AmbiQT without fine-tuning. Unlike Ambrosia’s synthetic databases, AmbiQT features real-world databases with more complex schemas. Nevertheless, generalization is possible because both datasets share vague ambiguity types. We use AmbiQT as a held-out test set: models trained on Ambrosia are evaluated on AmbiQT without fine-tuning. Following

Saparina & Lapata (2025), we use the same preprocessed splits (1,954 examples) and report execution match recall, precision, single and full coverage.

4.2 Training Details

We train Qwen3 4B Instruct (Yang et al., 2025) as our base model which offers a good balance between performance and efficiency; it is small enough for fast training and inference while maintaining competitive performance. We do not use reasoning models (e.g., Qwen3 4B Thinking), as preliminary experiments showed that they do not outperform instruction-tuned models on ambiguity resolution, while incurring much higher cost and latency.

Data Balancing Datasets with ambiguous inputs are often imbalanced, with many more unambiguous examples than ambiguous ones. However, ambiguous examples are more challenging and require stronger learning signals, especially with methods based on GRPO that have difficulty-level bias (Liu et al., 2025). We therefore rebalance the training data to maintain an ambiguous-to-unambiguous ratio of 3:1, by oversampling ambiguous examples. Ambiguous examples teach the model when to generate multiple interpretations, while unambiguous ones serve as regularization, preventing it from over-generating interpretations and maintaining high precision on straightforward inputs. During training, we sample 4 examples per batch, which combined with the 3:1 ratio, allocates more compute to harder ambiguous cases while still maintaining signal from unambiguous examples.

Reward Functions For Abg-CoQA, we compute rewards using word-overlap F1 as the similarity metric. While less reliable than LLM-as-a-judge for final evaluation, we found F1 provides effective training signal during reinforcement learning and is much faster to compute. For Ambrosia, we use execution match (comparing predicted and gold SQL query results) which provides an objective, verifiable reward signal. We train using DAPO (Yu et al., 2025) for 5 epochs with a learning rate of $5e-7$ and linear learning rate scheduling with 3% warmup. We generate 16 completions per prompt during training. Additional hyperparameters are in Appendix E.

4.3 Comparison Methods

We compare against approaches that isolate key aspects of ambiguity handling, namely prompting-based approaches, supervised fine-tuning, and state-of-the-art methods relying on interaction or multi-stage generation (implementation details are in Appendix F):

IntentPrompt We provide models with explicit instructions to reason about user intent, detect ambiguity and output multiple interpretation-answer pairs, using the same prompt template as our training approach (see Appendix A). This baseline isolates whether ambiguity handling can be achieved purely through prompting, without additional training. We evaluate across two model scales: Qwen3 4B and 235B (22B active parameters), with both standard instruction-tuned (Instruct) and reasoning (Thinking) variants.

SFT We fine-tune models to directly predict multiple interpretation-answer pairs. On Abg-CoQA we predict answers only as the dataset lacks gold interpretations. We use IntentPrompt but exclude reasoning instructions due to the absence of gold chain-of-thought annotations. For fine-tuning, we use Qwen3 4B Instruct, to match our model’s scale.

State-of-the-Art We compare against two state-of-the-art methods. ACT (Chen et al., 2025) is a preference optimization approach that trains models to ask clarification questions for ambiguous requests, requiring multiple stages to resolve ambiguity. We compare on Abg-CoQA against their best system, which uses Zephyr 7B β (Tunstall et al., 2023) as the policy model and Gemini Ultra 1.0 (Team et al., 2025) for user simulation. *DisambigParse* (Saparina & Lapata, 2025) is a two-stage approach for handling ambiguity in semantic parsing that first generates interpretations (with Llama3.1 8B Instruct; Team et al. 2024) and then uses them to guide SQL generation (with Qwen2.5 Coder 32B Instruct; Hui et al. 2024).

Method	Model	Ambiguous				Unambiguous		
		Overlap F1	Judge Recall	Judge Precision	Full Coverage	Overlap F1	Judge Recall	Judge Precision
IntentPrompt	4B Instruct	44.4	62.3	42.7	45.5	59.2	77.7	40.6
	4B Thinking	18.2	46.2	45.3	31.7	32.5	60.6	44.6
	235B Instruct	55.2	63.1	47.0	45.5	68.9	80.8	42.1
	235B Thinking	47.8	66.2	49.7	50.4	59.6	82.1	47.2
SFT*	4B Instruct	40.1	48.1	70.3	20.3	72.2	77.6	70.6
IntentRL	4B Instruct	72.9	78.1	53.5	61.0	84.4	90.9	40.7

Table 2: Evaluation on **Abg-CoQA** using Word-Overlap F1 and LLM-as-a-Judge metrics (Recall, Precision, Full Coverage) for ambiguous and unambiguous questions. SFT* outputs answers only as gold interpretations are unavailable. Best results per metric are in **bold**.

Method	Model	Ambiguous			Unambiguous	
		Recall	Precision	Full Coverage	Recall	Precision
IntentPrompt	4B Instruct	22.5	30.8	3.8	22.0	18.8
	4B Thinking	16.9	23.2	3.1	25.2	19.7
	235B Instruct	46.2	48.9	18.3	74.1	57.8
	235B Thinking	37.3	37.8	12.2	53.7	35.9
SFT	4B Instruct	68.5	75.6	58.8	77.7	74.7
IntentRL	4B Instruct	82.4	77.5	74.1	88.7	74.7

Table 3: Recall, Precision, and Full Coverage on the ambiguous and unambiguous subsets of **Ambrosia**. Best results per metric are in **bold**.

5 Results

5.1 Comparison with Prompting and Fine-tuning

Conversational Question Answering Table 2 summarizes our results on Abg-CoQA using word overlap F1 and LLM-as-a-Judge as evaluation metrics. We observe that larger models consistently outperform smaller ones, which suggests that our approach would benefit from stronger backbone models. Interestingly, IntentPrompt baselines achieve higher recall than precision, indicating that they over-detect ambiguity. For instance, the 235B Thinking model achieves 66.2% recall but only 49.7% precision on ambiguous questions; it generates multiple answers, some of which are spurious.

Supervised fine-tuning (SFT*, answers only since Abg-CoQA lacks gold interpretations) reveals a critical limitation: despite our efforts to rebalance the training data and make unambiguous examples less frequent than ambiguous ones, the model overfits to unambiguous cases. This results in high precision (70.3% for ambiguous, 70.6% for unambiguous) at the cost of recall (48.1%) and full coverage (20.3%) for ambiguous questions. The model learns to commit to a single answer rather than enumerate multiple options.

Our RL-based model (IntentRL) achieves the best results for ambiguous questions: 72.9% word-overlap F1, 78.1% recall, and 61.0% full coverage. For unambiguous questions, we achieve 84.4% word-overlap F1 and 90.9% recall, demonstrating that the model successfully recognizes when questions have a single intent. Lower precision (53.5% for ambiguous, 40.7% for unambiguous) reflects a bias toward generating additional interpretations as the model sometimes treats unambiguous questions as ambiguous and produces alternatives. We achieve the best precision on ambiguous questions among all interpretation-answer approaches (53.5% vs. 42.7–49.7% for IntentPrompt), showing that our method improves precision even though we do not explicitly optimize for it in the reward function, and the base models initially exhibit even lower precision than recall. High recall implies that when the model generates multiple interpretations, the correct ones are often included. In safety-critical applications, over-generating plausible interpretations is preferable to silently committing to one, potentially incorrect interpretation.

Text-to-SQL Parsing Table 3 presents our results on Ambrosia. Again, larger models outperform smaller ones, while reasoning models show no consistent improvement over instruct models (e.g., the 4B Thinking model achieves 16.9% recall vs. 22.5% for 4B Instruct on ambiguous questions). Unlike conversational QA, models show higher precision than recall on ambiguous questions, which means that they often generate one answer but miss valid interpretations. On unambiguous questions, the pattern reverses, indicating models struggle to calibrate when ambiguity is genuinely present.

IntentRL substantially outperforms all baselines, achieving 82.4% recall, 77.5% precision, and 74.1% full coverage on ambiguous questions, while maintaining strong performance on unambiguous questions (88.7% recall, 74.7% precision). High precision is particularly impressive since we do not explicitly optimise for it during training. There are several reasons for this. Firstly, the base model already shows more balanced or even slightly higher precision than recall (e.g., 4B Instruct has 30.8% precision vs. 22.5% recall on ambiguous questions), providing a strong starting point. Secondly, text-to-SQL parsing allows for verifiable, execution-based rewards, making it easier to learn valid interpretations compared to tasks with soft similarity-based rewards, such as conversational QA. Finally, IntentRL exhibits balanced performance on ambiguous *and* unambiguous questions. It correctly identifies when questions require multiple answers (high recall on ambiguous) and when a single answer suffices (74.7% precision on unambiguous).

Appendix G provides additional results including generalization across model families. Appendix H explores alternative ambiguity-handling strategies (answer-only generation, parallel sampling, and clarification questions), finding that none outperforms our approach, though IntentRL integrates effectively with answer-only and clarification pipelines.

5.2 Comparison with State-of-the-art Methods

Conversational QA In Table 4, we compare IntentRL with ACT (Chen et al., 2025), a state-of-the-art method that learns to disambiguate through multi-turn clarification. We report embedding-based semantic similarity (Reimers & Gurevych, 2019), macro-averaged across the test set, for direct comparison with their work (they use the term trajectory similarity). Despite generating interpretations and corresponding answers in a single turn, IntentRL achieves a similarity of 83.0%, significantly outperforming ACT’s 75.1%. This suggests that enumerating multiple interpretations (and their answers) upfront can be more effective than multi-turn clarification, allowing users to identify their intended meaning immediately without additional exchanges.

Method	Similarity
ACT	75.1
IntentRL	83.0

Table 4: Embedding-based similarity on **Abg-CoQA**.

Text-to-SQL In Table 5, we compare IntentRL with DisambigParse (Saparin & Lapata, 2025), a two-stage method that first predicts interpretations and then generates SQL. In addition to our metrics, we include Single Coverage (percentage of questions where the model correctly identifies *at least* one valid interpretation) to match their evaluation setup.

We outperform DisambigParse across all metrics, achieving much higher precision (77.5% vs. 26.3%) and full coverage (74.1% vs. 30.5%), while learning from answers alone in a single stage, unlike two-stage DisambigParse, which requires gold interpretations for training.

Method	Recall	Prec	Single	Full
DisambigParse	59.7	26.3	87.8	30.5
IntentRL	82.4	77.5	92.4	74.1

Table 5: DisambigParse and IntentRL results on **Ambrosia** (ambiguous questions).

5.3 Out-of-Domain Generalization

Following Saparin & Lapata (2025), we evaluate on AmbiQT (Bhaskar et al., 2023), another text-to-SQL benchmark with ambiguous questions which serves as out-of-domain evaluation for trainable methods. Although SFT achieves strong in-domain performance on Ambrosia, it fails to generalise to AmbiQT, achieving only 9.1% full coverage, learning dataset-specific patterns like the structure of databases and common query templates

Method	Model	Recall	Precision	Full Coverage
IntentPrompt	4B Instruct	20.5	27.2	9.3
	4B Thinking	20.7	25.1	12.2
	235B Instruct	60.5	63.2	38.2
	235B Thinking	51.8	43.2	38.1
SFT	4B Instruct	32.9	51.4	9.1
IntentRL	4B Instruct	66.9	58.2	49.1

Table 6: Out-of-domain evaluation on **AmbiQT** (SFT and IntentRL trained on Ambrosia).

Dataset	Alignment	Agreement	κ	AC1
Abg-CoQA	90.0	81.0	29.7	83.8
Ambrosia	91.7	84.0	27.2	86.5

Table 7: Human evaluation: alignment accuracy, inter-annotator agreement, Fleiss’ κ and Gwet’s AC1.

rather than the underlying task of identifying ambiguity. IntentRL generalises much better, identifying all valid interpretations for nearly half of the questions *without training* on this dataset (66.9% recall and 49.1% full coverage), demonstrating that IntentRL learns general reasoning skills about ambiguity that transfer across domains, databases, and queries.

5.4 Human Evaluation of Interpretations

Automatic metrics evaluate answer quality, but do not measure whether the interpretations genuinely explain their answers. Abg-CoQA lacks gold interpretations, and while Ambrosia does, we did not use them during training, so our model’s interpretations may differ in style and wording. We thus assess the quality of the predicted interpretations via a human study. We sampled 30 ambiguous examples from each dataset (80 interpretation-answer pairs in Abg-CoQA and 92 in Ambrosia), as interpretations are most critical when multiple plausible answers exist. For each pair, three annotators judged whether the interpretation is compatible with its answer (i.e., whether it supports or explains it). Annotators were given the full context, the ambiguous question, and instructed to evaluate alignment even in cases where the answer was incorrect; an interpretable wrong answer is more useful than an unexplained one, since it allows users to quickly identify and reject it (see Appendix I).

Table 7 reports average alignment accuracy across all annotations and agreement (percentage of pairs where all three annotators agree). Our model achieves 90.0% Alignment on Abg-CoQA and 91.7% on Ambrosia with high inter-annotator consensus (81.0–84.0%). Fleiss’ κ is low, reflecting the well-known kappa paradox (Byrt et al., 1993; Cicchetti & Feinstein, 1990): over 90% of judgments are positive, so chance agreement is near 87% and deflates κ even when observed agreement is high. We therefore also report Gwet’s AC1 (Gwet, 2008), which is robust to skewed distributions and indicates strong agreement. These results provide a more complete picture of model performance, demonstrating that it generates meaningful interpretations that explain the provided answers. Aligned interpretations open up the model’s decision-making, allowing users to select their intended meaning.

5.5 Analysis of Model Output

To understand model failures, we analyze the 30 examples from Section 5.4. On Abg-CoQA, we identify three types of error: (1) the model ignores important context from previous dialogue turns; (2) it sometimes produces valid interpretations but then gives factually incorrect answers, such as referencing a different person than the one implied in the interpretation; and (3) it predicts generic interpretations like “The question is ambiguous”, sometimes paired with the answer “Unknown” which appears in some training examples as a valid gold answer, but the model has learned to produce more often than necessary. On Ambrosia, interpretations are sometimes vague, paraphrasing rather than disambiguating

the question. The model occasionally produces correct interpretations but generates queries that do not follow them (e.g., adding ID fields not mentioned in the question) and sometimes produces non-executable queries, while noting the error in SQL comments (e.g., “this column does not exist”). Repeated interpretations also occur, though these can be filtered via execution.

Next, we analyze the reasoning traces of IntentPrompt and IntentRL both quantitatively and qualitatively. Figure 1 shows average reasoning length versus full coverage on ambiguous examples in Abg-CoQA and Ambrosia. Thinking models frequently overthink on Abg-CoQA, getting stuck on the same interpretation rather than considering alternatives, and sometimes miss interpretations entirely on Ambrosia. Instruct models with IntentPrompt are more concise but are outperformed by IntentRL. Our models achieve the best balance of efficiency and coverage; they explicitly consider ambiguity and alternative interpretations before committing to an answer. Example traces are in Appendix K.

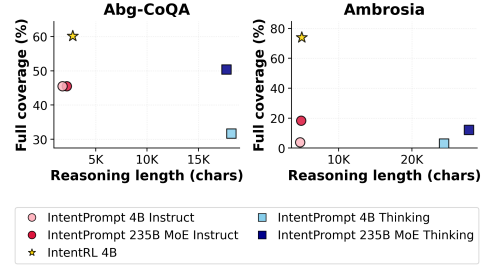


Figure 1: Reasoning length (as characters) vs. coverage (ambiguous subsets).

6 Conclusion

We introduced IntentRL, a reinforcement learning approach that trains models to reason explicitly about user intent by generating structured interpretation-answer pairs for ambiguous inputs. Using a dual reward objective (recall for ambiguous inputs to maximise coverage, precision for unambiguous ones to suppress spurious alternatives) our method requires only answer annotations as supervision, with no need for interpretation labels or clarification questions. Across conversational question-answering and text-to-SQL parsing, IntentRL substantially outperforms prompting and fine-tuning baselines, achieving strong gains in full coverage while maintaining competitive precision. Human evaluation confirms that predicted interpretations are highly aligned with their answers (over 90%) despite receiving no direct supervision on this objective. Crucially, IntentRL generalises well to out-of-domain data where supervised fine-tuning fails, suggesting the model learns general reasoning about intent rather than data-specific patterns.

Beyond performance gains, our experiments help explain why the approach works. We find that ambiguity is not inherently captured in a model’s output distribution: neither parallel sampling nor alternative prompting formats recover the diverse interpretations that ambiguous inputs admit, and only targeted training does. This indicates that the gains come from learning to reason about user intent rather than from the interpretation-answer format itself. Crucially, IntentRL also generalises to out-of-domain data where supervised fine-tuning collapses, suggesting that it learns transferable reasoning about intent rather than dataset-specific patterns. Where our model favours recall over precision, it does so by surfacing multiple plausible interpretations instead of silently committing to a single one. We argue this behaviour is desirable for safety-sensitive requests.

In future work, we plan to use interpretations to detect and correct errors without regenerating the full response, and to integrate reasoning about intent into broader post-training alongside domains such as mathematics and code. Controlled decoding could further improve efficiency and accuracy, e.g., by ensuring that interpretations are mutually exclusive. Finally, adapting to individual user preferences would make interpretations more personalized and relevant.

Acknowledgments

We thank the anonymous reviewers and area chair for their feedback. This work was supported by the Edinburgh International Data Facility (EIDF) and the Data-Driven Innovation Programme at the University of Edinburgh. Access to EIDF was facilitated through the

University of Edinburgh’s Generative AI Laboratory GAIL Fellow scheme. Lapata gratefully acknowledges the support of the UK Engineering and Physical Sciences Research Council (grant EP/W002876/1).

References

- Reinald Kim Amplayo, Kellie Webster, Michael Collins, Dipanjan Das, and Shashi Narayan. Query refinement prompts for closed-book long-form QA. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7997–8012, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.444. URL <https://aclanthology.org/2023.acl-long.444/>.
- Jonathan Berant, Maximillian Chen, Adam Fisch, Reza Aghajani, Fantine Huot, Mirella Lapata, and Jacob Eisenstein. Learning steerable clarification policies with collaborative self-play. *arXiv preprint arXiv:2512.04068*, 2025.
- Adithya Bhaskar, Tushar Tomar, Ashutosh Sathe, and Sunita Sarawagi. Benchmarking and improving text-to-SQL generation under ambiguity. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7053–7074, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.436. URL <https://aclanthology.org/2023.emnlp-main.436/>.
- Ted Byrt, Janet Bishop, and John B. Carlin. Bias, prevalence and kappa. *Journal of Clinical Epidemiology*, 46(5):423–429, 1993. doi: 10.1016/0895-4356(93)90018-V.
- Maximillian Chen, Ruoxi Sun, Sercan Ö. Arik, and Tomas Pfister. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. In *The 13th International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=SIE6VFps9x>.
- Jio Choi, Mohit Bansal, and Elias Stengel-Eskin. Language models identify ambiguities and exploit loopholes. *arXiv preprint arXiv:2508.19546*, 2025.
- Domenic V. Cicchetti and Alvan R. Feinstein. High agreement but low kappa: II. resolving the paradoxes. *Journal of Clinical Epidemiology*, 43(6):551–558, 1990. doi: 10.1016/0895-4356(90)90159-M.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.
- Yang Deng, Wenqiang Lei, Wenxuan Zhang, Wai Lam, and Tat-Seng Chua. PACIFIC: towards proactive conversational question answering over tabular and textual data in finance. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*, 2022.
- Zhongjun Ding, Yin Lin, and Tianjing Zeng. Ambisql: Interactive ambiguity detection and resolution for text-to-sql. *arXiv preprint arXiv:2508.15276*, 2025.
- Mingwen Dong, Nischal Ashok Kumar, Yiqun Hu, Anuj Chauhan, Chung-Wei Hang, Shuaichen Chang, Lin Pan, Wuwei Lan, Henghui Zhu, Jiarong Jiang, Patrick Ng, and Zhiguo Wang. PRACTIQ: A practical conversational text-to-SQL dataset with ambiguous and unanswerable queries. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.),

- Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 255–273, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.13. URL <https://aclanthology.org/2025.naacl-long.13/>.
- Avrilia Floratou, Fotis Psallidas, Fuheng Zhao, Shaleen Deep, Gunther Hagleither, Wangda Tan, Joyce Cahoon, Rana Alotaibi, Jordan Henkel, Abhik Singla, Alex Van Grootel, Brandon Chow, Kai Deng, Katherine Lin, Marcos Campos, K. Venkatesh Emani, Vivek Pandit, Victor Shnayder, Wenjing Wang, and Carlo Curino. NL2SQL is a solved problem... not! In *14th Conference on Innovative Data Systems Research, CIDR*, 2024. URL <https://www.cidrdb.org/cidr2024/papers/p74-floratou.pdf>.
- Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. Are we done with mmlu?, 2024.
- Meiqi Guo, Mingda Zhang, Siva Reddy, and Malihe Alikhani. Abg-coQA: Clarifying ambiguity in conversational question answering. In *3rd Conference on Automated Knowledge Base Construction*, 2021. URL <https://openreview.net/forum?id=SLDZ1o8FsJU>.
- Kilem L. Gwet. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48, 2008. doi: 10.1348/000711006X126600.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.
- Binyuan Hui, Jian Yang, Zeyu Cui, Jiayi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. Qwen2.5-coder technical report, 2024. URL <https://arxiv.org/abs/2409.12186>.
- Nan Huo, Xiaohan Xu, Jinyang Li, Per Jacobsson, Shipai Lin, Bowen Qin, Binyuan Hui, Xiaolong Li, Ge Qu, Shuzheng Si, et al. Bird-interact: Re-imagining text-to-sql evaluation for large language models via lens of dynamic interactions. *arXiv preprint arXiv:2510.05318*, 2025.
- Yeonjun In, Sungchul Kim, Ryan A. Rossi, Mehrab Tanjim, Tong Yu, Ritwik Sinha, and Chanyoung Park. Diversify-verify-adapt: Efficient and robust retrieval-augmented ambiguous question answering. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1212–1233, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.56. URL <https://aclanthology.org/2025.naacl-long.56/>.
- Kasia Kobalczyk, Nicolás Astorga, Tennison Liu, and Mihaela van der Schaar. Active task disambiguation with LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=JAMxRSXLfz>.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl.a.00276. URL <https://aclanthology.org/Q19-1026/>.

- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.
- Haau-Sing (Xiaocheng) Li, Mohsen Mesgar, André Martins, and Iryna Gurevych. Python code generation by asking clarification questions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14287–14306, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.799. URL <https://aclanthology.org/2023.acl-long.799/>.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding rl-zero-like training: A critical perspective. In *Second Conference on Language Modeling*, 2025. URL <https://openreview.net/forum?id=5PAF7PAY2Y>.
- Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. AmbigQA: Answering ambiguous open-domain questions. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5783–5797, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.466. URL <https://aclanthology.org/2020.emnlp-main.466/>.
- Sewon Min, Kenton Lee, Ming-Wei Chang, Kristina Toutanova, and Hannaneh Hajishirzi. Joint passage ranking for diverse multi-answer retrieval. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6997–7008, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.560. URL <https://aclanthology.org/2021.emnlp-main.560/>.
- Fangwen Mu, Lin Shi, Song Wang, Zhuohao Yu, Binqun Zhang, Chenxue Wang, Shichao Liu, and Qing Wang. Clarifygpt: Empowering llm-based code generation with intention clarification, 2024. URL <https://arxiv.org/pdf/2310.10996>.
- Jiaxin Pei, Aparna Ananthasubramaniam, Xingyao Wang, Naitian Zhou, Apostolos Dede-loudis, Jackson Sargent, and David Jurgens. POTATO: The portable text annotation tool. In Wanxiang Che and Ekaterina Shutova (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 327–337, Abu Dhabi, UAE, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-demos.33. URL <https://aclanthology.org/2022.emnlp-demos.33/>.
- Steven T Piantadosi, Harry Tily, and Edward Gibson. The communicative function of ambiguity in language. *Cognition*, 122(3):280–291, 2012.
- Mohammadreza Pourreza and Davood Rafiei. Evaluating cross-domain text-to-SQL models and benchmarks. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1601–1611, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.99. URL <https://aclanthology.org/2023.emnlp-main.99/>.
- Isha Puri, Mehul Damani, Idan Shenfeld, Marzyeh Ghassemi, Jacob Andreas, and Yoon Kim. Reaching beyond the mode: RL for distributional reasoning in language models. *arXiv preprint arXiv:2603.24844*, 2026.
- Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20*, pp. 3505–3506, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3406703. URL <https://doi.org/10.1145/3394486.3406703>.

- Nathan Rasmussen and William Schuler. A corpus of encyclopedia articles with logical forms. In Nicoletta Calzolari, Frédéric B  chet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, H  l  ne Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 1051–1060, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL <https://aclanthology.org/2020.lrec-1.132/>.
- Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL <https://aclanthology.org/D19-1410/>.
- Irina Saparina and Mirella Lapata. Ambrosia: A benchmark for parsing ambiguous questions into database queries. *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/a4c942a8405cc910f0a833d28d2573cc-Abstract-Datasets_and_Benchmarks_Track.html.
- Irina Saparina and Mirella Lapata. Disambiguate first, parse later: Generating interpretations for ambiguity resolution in semantic parsing. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 16825–16839, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.863. URL <https://aclanthology.org/2025.findings-acl.863/>.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL <https://arxiv.org/abs/2402.03300>.
- Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell Gordon, Niloofar Miresghallah, Christopher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. Position: a roadmap to pluralistic alignment. In *Proceedings of the 41st International Conference on Machine Learning, ICML’24*. JMLR.org, 2024.
- Taylor Sorensen, Benjamin Newman, Jared Moore, Chan Park, Jillian Fisher, Niloofar Miresghallah, Liwei Jiang, and Yejin Choi. Spectrum tuning: Post-training for distributional coverage and in-context steerability, 2025. URL <https://arxiv.org/abs/2510.06084>.
- Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. ASQA: Factoid questions meet long-form answers. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 8273–8288, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.566. URL <https://aclanthology.org/2022.emnlp-main.566/>.
- Elias Stengel-Eskin, Kyle Rawlins, and Benjamin Van Durme. Zero and few-shot semantic parsing with ambiguous inputs. In *The 12th International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=qL9gogRepu>.
- Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. Head-to-tail: How knowledgeable are large language models (LLMs)? A.K.A. will LLMs replace knowledge graphs? In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 311–325, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.18. URL <https://aclanthology.org/2024.naacl-long.18/>.

- Mehrab Tanjim, Yeonjun In, Xiang Chen, Victor Bursztyn, Ryan A. Rossi, Sungchul Kim, Guang-Jie Ren, Vaishnavi Muppala, Shun Jiang, Yongsung Kim, and Chanyoung Park. Disambiguation in conversational question answering in the era of LLMs and agents: A survey. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9537–9550, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.482. URL <https://aclanthology.org/2025.emnlp-main.482/>.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, et al. Gemini: A family of highly capable multimodal models, 2025. URL <https://arxiv.org/abs/2312.11805>.
- Llama 3 Team, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Bap tiste Roziere, Bethany Biron, Binh Tang, et al. The llama 3 herd of models. *ArXiv*, abs/2407.21783, 2024. URL <https://api.semanticscholar.org/CorpusID: 271571434>.
- Sergey Troshin, Irina Saparina, Antske Fokkens, and Vlad Niculae. Asking a language model for diverse responses. *arXiv preprint arXiv:2509.17570*, 2025.
- Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment, 2023. URL <https://arxiv.org/abs/2310.16944>.
- Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. <https://github.com/huggingface/trl>, 2020.
- Bing Wang, Yan Gao, Zhoujun Li, and Jian-Guang Lou. Know what I don’t know: Handling ambiguous and unknown questions for text-to-SQL. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 5701–5714, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.352. URL <https://aclanthology.org/2023.findings-acl.352/>.
- Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.
- Peter West and Christopher Potts. Base models beat aligned models at randomness and creativity. In *Second Conference on Language Modeling*, 2025. URL <https://openreview.net/forum?id=vqN8uom4A1>.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

- Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R Tomz, Christopher D Manning, and Weiyan Shi. Verbalized sampling: How to mitigate mode collapse and unlock llm diversity. *arXiv preprint arXiv:2510.01171*, 2025a.
- Michael JQ Zhang, W. Bradley Knox, and Eunsol Choi. Modeling future conversation turns to teach LLMs to ask clarifying questions. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL <https://openreview.net/forum?id=cwuSAR7EKd>.
- Yiming Zhang, Avi Schwarzschild, Nicholas Carlini, J Zico Kolter, and Daphne Ippolito. Forcing diffuse distributions out of language models. In *First Conference on Language Modeling*, 2024. URL <https://openreview.net/forum?id=9JY1QLVFPZ>.
- Yiming Zhang, Harshita Diddee, Susan Holm, Hanchen Liu, Xinyue Liu, Vinay Samuel, Barry Wang, and Daphne Ippolito. Noveltybench: Evaluating creativity and diversity in language models. In *Second Conference on Language Modeling*, 2025c. URL <https://openreview.net/forum?id=XZm1ekzERf>.
- George Kingsley Zipf. *Human Behaviour and the Principle of Least Effort*. Addison-Wesley Press, 1949.

A IntentPrompt

Below we provide the prompt template for reasoning about user intent with explicit interpretations, used in IntentPrompt baseline and IntentRL training:

IntentPrompt Template

You receive a context and a question. First analyse the question to identify different possible interpretations, then provide the corresponding answer for each interpretation.

If the question is unambiguous, provide a single clear interpretation and answer. If the question contains ambiguities, underspecifications, or could be understood in multiple ways, provide multiple plausible interpretations (maximum of 5) and corresponding answers.

An interpretation is a self-contained, precise reformulation of the question that clarifies scope, entities, and intent so that the question could be answered unambiguously. Different interpretations should lead to meaningfully different answers.

Context:
{context}

Question to interpret and answer:
{question}

General Instructions:

- Think through the question step-by-step
- Consider different ways the question could be understood
- Look for potential ambiguities in terminology, scope, or intent
- Consider how the provided context might influence interpretation

Task-Specific Instructions:
{instructions}

Output Format:

Provide your analysis and reasoning, then conclude with interpretations and answers. Follow the format below:

...analysis and reasoning...

<answer>

****Interpretation 1:**** [State first interpretation]

[Answer for interpretation 1]

[Continue for all interpretations (up to 5) if the question is ambiguous...]

<answer>

Task-specific instructions for question answering and text-to-SQL parsing are shown below:

Answer Generation Instructions

- Provide direct, concise answers based on the given context. Keep your answers short and to the point.
- Do not include any other text or explanation in the lines that contain the answers.

SQL Generation Instructions

- Make sure you only output the information that is asked in the question. If the question asks for a specific column, make sure to only include that column in the SELECT clause, nothing more.
- The generated query(ies) should return all of the information asked in the question without any missing or extra information.
- Use a fenced code block with "sql" annotation, e.g. ````sql ... ````

B DAPO Algorithm

DAPO (Yu et al., 2025) extends GRPO with several key modifications designed for long-form reasoning tasks. It removes the KL penalty term (as divergence from the initial model is acceptable during reasoning training) and introduces the following changes:

Decoupled Clipping DAPO uses asymmetric clipping parameters ϵ_{low} and ϵ_{high} to enhance exploration while maintaining stability.

Token-Level Loss Instead of sample-level averaging, DAPO weights tokens equally across all responses, giving longer sequences more influence proportional to their length.

Dynamic Sampling DAPO filters out prompts where all sampled responses are either correct or incorrect (accuracy equal to 1 or 0), ensuring effective gradients throughout training.

Overlong Reward Shaping DAPO applies a soft length penalty that gradually reduces rewards for responses exceeding a maximum length threshold L_{max} , rather than hard truncation. This discourages excessively long outputs while preserving useful gradient signal.

Let G denote the number of responses sampled per prompt. Each response o_i receives a binary correctness reward:

$$R_i = \text{is_equivalent}(a, o_i) \in \{0, 1\}$$

which is then shaped via the overlong reward penalty:

$$R_i^{\text{shaped}} = R_i \cdot \max\left(1 - \alpha \cdot \frac{\max(|o_i| - L_{\text{max}}, 0)}{L_{\text{max}}}, \beta_{\text{min}}\right)$$

where L_{max} is the maximum desired response length, α controls the penalty slope, and β_{min} is a floor preventing the reward from being fully zeroed out.

The DAPO objective is defined over a filtered subset $D' \subseteq D$ that excludes prompts with trivial outcomes:

$$D' = \left\{ (q, a) \in D \mid 0 < \sum_{i=1}^G \mathbf{1}[\text{is_equivalent}(a, o_i)] < G \right\}$$

The training objective is then:

$$J_{\text{DAPO}}(\theta) = \mathbb{E}_{(q, a) \sim D', \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot | q)} \left[\frac{1}{\sum_{i=1}^G |o_i|} \sum_{i=1}^G \sum_{t=1}^{|o_i|} \min(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}}) \hat{A}_{i,t}) \right]$$

where the probability ratio and normalized advantage are given by:

$$r_{i,t}(\theta) = \frac{\pi_{\theta}(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}$$

$$\hat{A}_{i,t} = \frac{R_i^{\text{shaped}} - \text{mean}(\{R_j^{\text{shaped}}\}_{j=1}^G)}{\text{std}(\{R_j^{\text{shaped}}\}_{j=1}^G)}$$

C Types of Ambiguity

Each dataset we use has its own ambiguity taxonomy. Below, we briefly describe the types of ambiguity as defined by the dataset authors. For more details, we refer readers to the original papers.

Abg-CoQA Guo et al. (2021) defines four ambiguity types:

Coreference resolution occurs when pronouns or referring expressions can map to multiple entities in the context. For example, given a passage mentioning a husband and a son, the question “What is his name?” could refer to either.

Time-dependency occurs when temporal context is underspecified, allowing questions to refer to different time periods. For example, in a story describing an accident, the question “What was everyone doing?” can refer to actions either before or after the accident. The example in Table 1 illustrates a similar case where “What other state ruled it?” can refer to different historical periods.

Answer types are ambiguous with respect to the intended level of granularity. For example, when asked “What pupils’ were reading?”, the answer could be either novels a specific book.

Event ambiguity occurs when a story contains multiple parallel events or storylines, and questions underspecify which event they are referring to. For example, when a story mentions both a police investigation and CNN’s reporting on the same incident, a question about whether the suspect was identified by name could refer to either storyline (the police identify but CNN does not).

Ambrosia (Saparin & Lapata, 2024) defines three ambiguity types:

Scope ambiguity occurs when it is unclear which elements a quantifier (e.g., “every”, “each”) refers to. For example, the question “What activities does each gym offer?” has a collective interpretation (a common class for all gyms in the database) and distributive one (each gym is considered separately, a list of all classes and gyms).

Attachment ambiguity occurs when it is unclear where a modifier attaches in a sentence. For example, “Show screenwriters and editors on a work-for-hire” can mean either both roles are on work-for-hire contracts (high attachment to the verb) or only editors are (low attachment to “editors”). Table 1 shows another example where a budget constraint may attach to one or both genres.

Vagueness occurs when context creates uncertainty about which database entities are being referenced. For example, a question about “required experience” of the candidate is vague when the database contains both minimum and preferred years columns.

AmbiQT (Bhaskar et al., 2023) defines four ambiguity types:

Column ambiguity occurs when multiple columns satisfy the same semantic description. For example, the question “List the ids of all students” is ambiguous when a database contains both columns named `admission_number` and `roll_number`.

Table ambiguity occurs when multiple tables represent similar entities. For example, “How many singers do we have?” could query either an `artist` or `performer` table.

Join ambiguity occurs when it is unclear whether information should be retrieved from a single table or from joining multiple tables. For example, asking for a person’s name and address is ambiguous when address is in the same table as name and `postal_address` is in a separate `person_details` table.

Precomputed aggregates arise when databases store pre-aggregated values alongside raw data. For example, “Find the average weight for each pet type” could either compute the aggregate from data or select from pre-computed `avg_weight`.

D LLM-as-a-Judge Setup

We use an LLM judge because it can recognize semantic equivalence despite wording differences unlike rule-based metrics (exact match, word-overlap F1). For example, in Table 1 “The early Polish state” vs. “Poland” would score 0% F1, despite having identical meaning. This is crucial for evaluating zero-shot and RL-trained models, which often

generate answers in different styles than gold annotations. We adapt the prompt from Sun et al. (2024), who found 98% agreement between human judgments and LLM-based metrics.

We chose Qwen3 4B Instruct (Yang et al., 2025) as the judge model because it is fast and cost-effective for large-scale evaluation, and provides reliable judgments for our task. In preliminary experiments, we compared our judge with the larger Qwen3 235B A22B Instruct, across multiple evaluation sets and found 1.4% difference in judgment accuracy on average. In most cases, the smaller model was more conservative, marking particularly non-obvious answer reformulations as incorrect.

E Training Hyperparameters

We train our models using DAPO implemented with the TRL library (von Werra et al., 2020) with hyperparameters adapted from TRL defaults. We use the Qwen3-4B-Instruct-2507 model and train for 5 epochs with AdamW (fused implementation) with learning rate $5e-7$, linear scheduler with 3% warmup, and gradient clipping at norm 1.0. The per-device batch size is 16. During training, we generate 16 completions per prompt with maximum completion length of 2,500 tokens.

We use vLLM (Kwon et al., 2023), Flash Attention 2 (Dao, 2024) and DeepSpeed (Rasley et al., 2020) for efficient generation during training. All experiments were conducted on 5 NVIDIA H100 80GB GPUs: 1 GPU dedicated to the vLLM server for generation and 4 GPUs for training. For evaluation, we use a single GPU, and choose the last checkpoint. We use default sampling parameters (recommended by the developers) for Qwen3 4B Instruct: temperature = 0.7, top-k = 20, top-p = 0.8, and maximum length is 30k tokens. This limit applies uniformly across all methods for fair comparison, and is generous enough to accommodate the long traces of the Thinking baselines (Figure 1). Our model, however, rarely generates outputs longer than the 2,500-token completion limit used during training.

Code, data, and models are available at <https://github.com/saparina/intentRL>.

F Comparison Details

We use the updated versions of the Qwen3 model family: Qwen3-4B-Instruct-2507, Qwen3-4B-Thinking-2507, Qwen3-235B-A22B-Instruct-2507, Qwen3-235B-A22B-Thinking-2507.

We train SFT baselines using the TRL library with default hyperparameters. We modified the prompts shown in Appendix A to exclude chain-of-thought instructions (e.g., “Provide your analysis and reasoning”) since we do not have gold chain-of-thought reasoning. Following the same procedure as our method, we train for 5 epochs, balance the training data, and select the last checkpoint for evaluation. We use the same model (Qwen3 4B Instruct) with the same LoRA configuration. Training uses the AdamW optimizer (fused implementation) with a learning rate $2e-5$, linear scheduler with no warmup, and gradient clipping at norm 1.0. The per-device batch size is 8 with no gradient accumulation. We use gradient checkpointing for memory efficiency and set the maximum prompt length to 8192 tokens. All SFT experiments were conducted on 1 NVIDIA H100 80GB GPU.

For both prompting and fine-tuned methods, we use the same maximum length as our method: 30k tokens. We also use the default sampling parameters recommended by the model developers for all models.

G Additional Results

Generalization to Different Model Families To evaluate whether our findings generalize beyond Qwen models, we experiment with instruction-tuned Gemma3 4B and Llama3.1 8B (Table 8). Across models, IntentRL outperforms both IntentPrompt and SFT baselines. Gains are consistent across model architectures and sizes, with improvements of 8–12% in recall and 5–9% in full coverage over SFT. Prompting alone struggles across all models

Model	Method	Ambiguous			Unambiguous	
		Recall	Precision	Full Coverage	Recall	Precision
Qwen3 4B Instruct	IntentPrompt	22.5	30.8	3.8	22.0	18.8
	SFT	68.5	75.6	58.8	77.7	74.7
	IntentRL	82.4	77.5	74.1	88.7	74.7
Gemma3 4B IT	IntentPrompt	21.8	26.0	2.3	42.1	25.6
	SFT	56.3	61.2	50.4	65.0	63.4
	IntentRL	78.8	72.8	69.5	80.9	71.9
Llama3.1 8B Instruct	IntentPrompt	17.4	16.4	2.3	35.6	18.1
	SFT	70.5	70.9	64.9	75.7	74.2
	IntentRL	79.4	74.4	71.8	87.4	78.8

Table 8: Model family comparison on Ambrosia (Recall, Precision, and Full Coverage). Best results per model are in **bold**.

Method	Ambiguous				Unambiguous		
	Overlap F1	Judge Recall	Judge Precision	Full Coverage	Overlap F1	Judge Recall	Judge Precision
IntentRL	72.9	78.1	53.5	61.0	84.4	90.9	40.7
w/o Balancing	38.1	45.8	81.3	15.5	77.9	82.1	82.1
Down-sampling	62.3	75.4	43.7	59.4	86.2	90.8	37.4

(a) Ablation of IntentRL on Abg-CoQA.

Method	Ambiguous			Unambiguous	
	Recall	Precision	Full Coverage	Recall	Precision
IntentRL	82.4	77.5	74.1	88.7	74.7
w/o Balancing	49.4	85.0	9.2	84.5	81.7

(b) Ablation of IntentRL on Ambrosia.

Table 9: Ablation of IntentRL: *w/o Balancing* refers to IntentRL trained without training data balancing and *Down-sampling* refers to balancing by removing unambiguous examples rather than oversampling ambiguous ones (see Section 4.2)

(13.9–22.5% recall), demonstrating that explicit training for ambiguity resolution is necessary regardless of base model choice. Qwen3 4B Instruct achieves the strongest performance overall, consistent with its reported advantages (Yang et al., 2025), though all models benefit from our training approach. Performance differences across models suggest that stronger base models yield better results, while consistent improvements demonstrate that our approach is model-agnostic and represents a general methodology for handling ambiguity.

Data Balancing Ablations We ablate the effect of training data balancing (Section 4.2). As shown in Tables 9a and 9b, removing the 3:1 ambiguous-to-unambiguous ratio causes full coverage drops from 61.0% to 15.5% on Abg-CoQA and from 74.1% to 9.2% on Ambrosia, while precision increases from 53.5%–77.5% to over 80% on both benchmarks. Without balancing, the training set is dominated by unambiguous examples, which are easier, and the model already performs well on them by producing a single answer. DAPO reinforces this behavior since the reward signal from these straightforward examples is strong and consistent, while ambiguous examples are rare and harder to learn from.

We also compare against down-sampling unambiguous examples to the same ratio on Abg-CoQA. Up-sampling is better across all metrics on ambiguous questions, with a particularly large gap in word-overlap F1 (72.9 vs. 62.3), while performance on unambiguous questions is comparable. Balancing is thus an important design choice in our setup rather than a

	GSM8K	MMLU	MMLU-Pro	MMLU-Redux
Qwen3 4B Instruct (reported) ²	—	—	69.6	84.2
Qwen3 4B Instruct (reproduced)	92.2	80.0	70.7	84.8
IntentRL (Abg-CoQA)	92.8	80.8	71.0	84.9
IntentRL (Ambrosia)	92.4	80.2	70.4	85.0

Table 10: Accuracy of Qwen3 4B Instruct and IntentRL variants trained on Abg-CoQA and Ambrosia on mathematical reasoning (GSM8K) and general knowledge tasks (MMLU, MMLU-Pro, MMLU-Redux).

Method	Recall	Precision	Single Coverage	Full Coverage
DisambigParse	58.5	35.3	88.0	30.0
IntentRL	66.9	58.2	84.7	49.1

Table 11: Comparison between DisambigParse (Saparina & Lapata, 2025) and IntentRL on AmbiQT when trained on Ambrosia. We report Recall, Precision, Single and Full Coverage.

minor hyperparameter. We view the underlying issue as an instance of the difficulty bias in GRPO-style training (Liu et al., 2025); oversampling is one remedy, but alternative sampling schemes, curricula, or modifications to the RL objective may work as well. We leave this to future work.

SFT with Silver Interpretations On Abg-CoQA, SFT predicts answers only, as the dataset lacks gold interpretations. To verify this does not disadvantage the baseline, we generated silver interpretations by prompting Qwen3 4B Instruct to convert each gold clarification question into an explicit interpretation, and trained SFT on the resulting pairs. Results are slightly below answer-only SFT on ambiguous questions (39.9 vs. 40.1 word-overlap F1, 17.9 vs. 20.3 full coverage), consistent with Ambrosia, where the answer-only SFT variant also outperforms the variant trained with gold interpretations (Table 14b). Interpretations vary more in wording and are therefore harder to learn, so adding them does not improve SFT.

General Knowledge Evaluation To verify that IntentRL does not degrade general capabilities, we evaluate on standard reasoning and knowledge benchmarks: GSM8K (Cobbe et al., 2021), MMLU (Hendrycks et al., 2021), MMLU-Pro (Wang et al., 2024), and MMLU-Redux (Gema et al., 2024). All evaluations use 5-shot chain-of-thought prompts following Yang et al. (2025). Note that we use the updated Qwen3 model (Qwen3-4B-Instruct-2507) rather than the version evaluated by Yang et al. (2025). As shown in Table 10, IntentRL models match the base Qwen3 4B Instruct across all tasks, confirming that training for ambiguity resolution does not come at the cost of general performance. This is an encouraging signal that IntentRL could be integrated into broader post-training regimes alongside other reasoning domains such as mathematics or code, likely with larger datasets.

Comparison with State-of-the-Art on Out-of-Domain Data We extend the comparison with DisambigParse (Saparina & Lapata, 2025), the current state of the art for ambiguous text-to-SQL, to the out-of-domain setting. Saparina & Lapata (2025) highlight out-of-domain generalization as a key strength of their approach. Table 11 reports results on AmbiQT for both methods when trained on Ambrosia. IntentRL outperforms DisambigParse across most metrics, with substantial gains in precision (58.2% vs. 35.3%) and full coverage (49.1% vs. 30.0%), demonstrating that the advantages of IntentRL hold even in the setting most favourable to the previous state of the art.

Performance of Closed-Source Frontier Models To contextualise our results against the strongest available systems, we evaluate two closed-source frontier models: GPT-5.4 ³

²<https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507>

³<https://openai.com/index/introducing-gpt-5-4/>

Method	Ambiguous			Unambiguous	
	Recall	Precision	Full Coverage	Recall	Precision
Gemini 2.5 Pro	49.5	51.8	14.5	75.1	51.8
GPT-5.4 (low)	45.6	53.2	8.4	78.0	58.8
GPT-5.4 (medium)	49.8	49.1	14.5	83.8	57.5
GPT-5.4 (high)	50.3	45.7	16.0	82.2	53.8
IntentRL (4B)	82.4	77.5	74.1	88.7	74.7

Table 12: Comparison with closed-source frontier models on Ambrosia. GPT-5.4 reasoning effort levels are shown in parentheses. Best results per metric are in **bold**.

Method	Ambiguous F1	Unambiguous F1
IntentPrompt	6.3	9.7
SFT	11.2	16.1
IntentRL	13.6	16.5

Table 13: F1 on ambiguous and unambiguous subsets of **AmbigQA** (closed-book). All methods use Qwen3 4B Instruct. Best results per metric are in **bold**.

(gpt-5.4-2026-03-05) and Gemini 2.5 Pro ⁴. Both are proprietary systems whose exact architectures and parameter counts are undisclosed, though they are widely understood to be substantially larger than our 4B parameter model. We use the same IntentPrompt template (Appendix A) for both models. For GPT-5.4, we evaluate three reasoning effort levels (low, medium, high) to assess whether additional compute at inference time improves ambiguity handling. For all models, we set the maximum generation length to 30k tokens, matching the budget used for IntentRL and all other methods in our evaluation. Due to the cost of running closed-source API models at scale, we focus this comparison on Ambrosia, which provides verifiable evaluation through SQL execution and represents the more challenging setting for ambiguity resolution.

Table 12 reports results on the ambiguous and unambiguous subsets of Ambrosia. IntentRL considerably outperforms both closed-source models across all metrics, achieving 74.1% full coverage compared to 16.0% for GPT-5.4 at high effort and 14.5% for Gemini 2.5 Pro. Recall follows a similar pattern: 82.4% for IntentRL versus 50.3% for GPT-5.4 and 49.5% for Gemini 2.5 Pro. Increasing reasoning effort from low to high results in only modest improvements for GPT-5.4 (8.4% to 16.0% full coverage), indicating that additional inference-time computation alone does not address the underlying challenge. On unambiguous questions, IntentRL also maintains a clear advantage in both recall (88.7% vs. 82.2–83.8% for GPT-5.4) and precision (74.7% vs. 53.8–58.8%). These results reinforce our central finding: handling ambiguity effectively requires targeted training rather than model scale or inference-time compute, particularly for structured output tasks where generating multiple valid alternatives demands systematic reasoning about the input.

Generalization to Open-Domain Question Answering We additionally report exploratory results on AmbigQA (Min et al., 2020), an open-domain benchmark with multiple valid answers per question. Unlike the datasets in our main experiments, AmbigQA provides no grounding context, so we evaluate in a closed-book setting where the model must rely on parametric knowledge. We follow the same training setup as Section 4.2 and report F1, the standard AmbigQA metric. As shown in Table 13, IntentRL outperforms both IntentPrompt and SFT on ambiguous and unambiguous subsets. However, absolute F1 is low across all methods. Since our reward operates at the answer level, the training signal in a closed-book setup is confounded by parametric knowledge: a model that surfaces every valid interpretation still scores poorly if it lacks the underlying facts, which is orthogonal to the ambiguity reasoning we target. AmbigQA also poses practical challenges for our setup,

⁴<https://blog.google/innovation-and-ai/models-and-research/google-deepmind/gemini-model-thinking-updates-march-2025/>

as roughly 7% of questions admit more than five interpretations (up to 41), an interpretation space too large for our method.

H Alternative Approaches to Handling Ambiguity

Handling ambiguous questions is a challenging task, in part because there is no single format or interface that is universally optimal for users, applications, and models. Below we consider three alternative approaches to ambiguity handling: generating only answers without explicit interpretations, sampling answers in parallel, and generating clarification questions. We compare these approaches with our proposed interpretation-answer format and show how it can be integrated into each: it naturally generalizes to the answer-only format (which can also be viewed as an ablation of our approach), and it helps produce better clarifications and answers, demonstrating its utility in multi-stage pipelines. Finally, we discuss the trade-offs between these formats.

H.1 Answer-Only Generation

We consider approaches that generate multiple plausible answers $\{a_1, \dots, a_m\}$ without explicit interpretations. For the conversational QA example in Table 1, such approaches produce “Greater Moravia”, “The early Polish state” and “The Holy Roman Empire” without disambiguating which interpretation each answer reflects. This can be viewed as an ablation of our approach: we train IntentRL to reason about user intent and generate answers only, without producing interpretations.

We consider the same comparisons with prompting alone and SFT. Note that in this setting, training does not require gold interpretations, only answers; this is the variant used for Abg-CoQA in Table 2. We use the same prompt as for interpretation-answer pairs with minimal changes (removing the request for interpretations). Next we provide the prompt:

IntentPrompt Template, Answer-only

You receive a context and a question. First analyse the question to identify different possible interpretations, then provide the corresponding answer for each interpretation.

If the question is unambiguous, provide a single answer. If the question contains ambiguities, underspecifications, or could be understood in multiple ways, provide multiple plausible answers (maximum of 5).

Context:
{context}

Question:
{question}

General Instructions:
 - Think through the question step-by-step
 - Consider different ways the question could be understood
 - Look for potential ambiguities in terminology, scope, or intent
 - Consider how the provided context might influence interpretation

Task-Specific Instructions:
{instructions}

Output Format:
Provide your analysis and reasoning, then conclude with answers. Follow the format below:

...analysis and reasoning...
 <answer>
 [State first answer]

[Continue for all answers (up to 5) if the question is ambiguous...]
 <answer>

Method	Model	Ambiguous				Unambiguous		
		Overlap F1	Judge Recall	Judge Precision	Full Coverage	Overlap F1	Judge Recall	Judge Precision
Answer-only								
IntentPrompt	4B Instruct	48.5	58.5	42.6	35.8	56.8	74.8	40.2
	4B Thinking	31.3	60.4	52.6	37.4	53.4	72.7	50.2
	235B Instruct	47.8	62.7	52.2	39.8	64.7	76.6	44.7
	235B Thinking	44.6	70.4	51.9	55.3	66.1	85.5	51.5
SFT	4B Instruct	40.1	48.1	70.3	20.3	72.2	77.6	70.6
IntentRL	4B Instruct	73.8	78.9	45.5	61.8	85.0	90.6	35.0
Interpretation-Answer Pairs								
IntentPrompt	4B Instruct	44.4	62.3	42.7	45.5	59.2	77.7	40.6
	4B Thinking	18.2	46.2	45.3	31.7	32.5	60.6	44.6
	235B Instruct	55.2	63.1	47.0	45.5	68.9	80.8	42.1
	235B Thinking	47.8	66.2	49.7	50.4	59.6	82.1	47.2
IntentRL	4B Instruct	72.9	78.1	53.5	61.0	84.4	90.9	40.7

(a) Comparison of answer-only and interpretation-answer formats on Abg-CoQA.

Method	Model	Ambiguous			Unambiguous	
		Recall	Precision	Full Coverage	Recall	Precision
Answer-only						
IntentPrompt	4B Instruct	23.8	48.1	2.3	58.3	57.2
	4B Thinking	27.4	49.1	2.3	58.3	56.1
	235B Instruct	35.2	53.3	8.4	69.3	64.1
	235B Thinking	28.2	32.8	5.3	61.8	48.4
SFT	4B Instruct	72.7	75.1	66.4	78.0	73.4
IntentRL	4B Instruct	82.4	78.3	71.8	86.1	75.3
Interpretation-Answer Pairs						
IntentPrompt	4B Instruct	22.5	30.8	3.8	22.0	18.8
	4B Thinking	16.9	23.2	3.1	25.2	19.7
	235B Instruct	46.2	48.9	18.3	74.1	57.8
	235B Thinking	37.3	37.8	12.2	53.7	35.9
SFT	4B Instruct	68.5	75.6	58.8	77.7	74.7
IntentRL	4B Instruct	82.4	77.5	74.1	88.7	74.7

(b) Comparison of answer-only and interpretation-answer formats on Ambrosia.

Table 14: Comparison of answer-only and interpretation-answer formats on Abg-CoQA and Ambrosia. Best results per metric are in **bold** (marked separately per format).

Results Tables 14a and 14b show results for answer-only generation alongside interpretation-answer pairs on Abg-CoQA and Ambrosia. Answer-only IntentRL is competitive with the interpretation-answer variant, showing comparable recall and full coverage on Abg-CoQA (78.9 vs. 78.1 and 61.8 vs. 61.0, respectively), though the interpretation-answer format achieves higher full coverage on Ambrosia (74.1 vs. 71.8). This confirms that learning to reason about user intent using RL, not just the output format, is the primary driver of improvement. Both IntentRL variants outperform prompting and SFT in answer-only mode. The main advantage of the interpretation-answer format is interpretability: it provides explicit justifications for each answer, making predictions more transparent, while requiring the same supervision as the answer-only version. Providing interpretations is particularly beneficial in downstream tasks like text-to-SQL parsing, where the answer (an SQL query) can be difficult to understand without them.

H.2 Parallel Sampling

Both our interpretation-answer pair generation and the answer-only generation considered above follow the same general approach: alternatives are generated sequentially within a single model call. The main advantage is that the model is conditioned on previously generated alternatives when producing the next one, which encourages it to explore distinct interpretations.

An alternative strategy is to sample multiple independent responses from the model and aggregate them. If a question is ambiguous and the model recognises this, it should be less certain in its predictions, meaning alternatives should have comparable probabilities and thus appear across different samples. We refer to this approach as *parallel sampling*. In this setting, the model is prompted with standard task-specific instructions (for conversational question answering and text-to-SQL parsing) with no mention of ambiguity or multiple outputs, which we refer to as the *task prompt*. We provide these prompts below:

Conversational Question Answering Task Prompt	Text-to-SQL Task Prompt
You receive a context and a question. Context: {context} Question to answer: {question} Answer Generation Instructions: {instructions} Output Format: Follow the format below: <answer> [State your answer] <answer>	You receive a context and a question. Context: SQLite database schema: {context} This schema describes the database’s structure, including tables, columns, primary keys, foreign keys, and any relevant relationships or constraints. Question: {question} SQL Generation Instructions: {instructions} Output Format: Follow the format below: <answer> [State your SQL query] <answer>

The model predicts a single answer per call, we then sample n independent responses and aggregate all predicted answers. As the model is not expected to verbalise interpretations, we compare this approach to the answer-only variant of IntentPrompt. We also evaluate a variant without chain-of-thought reasoning (IntentPrompt w/o CoT), where the model is not prompted to provide explicit reasoning before generating answers. We consider two versions of parallel sampling: Sample-5 matches IntentPrompt’s budget of 5 answers, while Sample-10 doubles it.

Results Table 15a and Table 15b provide results on Abg-CoQA and Ambrosia for different answer-only prompt variations. All models are based on Qwen3 4B Instruct. With the task prompt, the model is not expected to predict more than a single answer; we also verified that it never responds with a clarification question (which could be a reasonable response to an ambiguous request) nor acknowledges ambiguity in any other way. Parallel sampling provides modest gains over a single response on Abg-CoQA and no gains on Ambrosia, and underperforms against IntentPrompt even with twice the computational budget (Sample-10). This indicates that the model consistently collapses to the same interpretation. In other words, **ambiguity is not inherently captured in the model’s output distribution: our proposed approach learns to reason about intent and consider diverse interpretations rather than merely learning to format what parallel sampling could also recover.**

Our findings are consistent with prior work showing that parallel sampling often fails to produce diverse outputs, as models tend to be biased towards certain completions,

Method	# answers	Ambiguous				Unambiguous		
		Overlap F1	Judge Recall	Judge Precision	Full Coverage	Overlap F1	Judge Recall	Judge Precision
Task prompt	1	30.3	35.0	62.6	10.6	51.0	66.5	67.6
+Sample-5	5	40.6	41.9	62.0	17.9	55.5	72.4	67.7
+Sample-10	10	41.6	43.8	62.6	18.7	56.9	74.0	67.8
IntentPrompt w/o CoT	5	29.3	41.5	43.0	14.6	48.8	70.2	52.0
IntentPrompt	5	48.5	58.5	42.6	35.8	56.8	74.8	40.2

(a) Variations of *answer-only* prompts on Abg-CoQA.

Method	# answers	Ambiguous			Unambiguous	
		Recall	Precision	Full Coverage	Recall	Precision
Task prompt	1	21.2	49.6	0.0	56.3	56.3
+Sample-5	5	24.2	50.5	0.0	65.4	60.2
+Sample-10	10	24.2	50.5	0.0	66.7	60.2
IntentPrompt w/o CoT	5	16.0	36.5	0.8	42.4	42.4
IntentPrompt	5	23.8	48.1	2.3	58.3	57.2

(b) Variations of *answer-only* prompts on Ambrosia.

Table 15: Variations of *answer-only* prompts. **Task Prompt** refers to prompting the model with standard task-specific instructions with no mention of ambiguity. **+Sample- n** refers to sampling n independent responses from the Task Prompt. **IntentPrompt w/o CoT** refers to the prompt where we do not prompt the model to provide explicit reasoning. All models are based on Qwen3 4B Instruct.

downweighting alternatives (Zhang et al., 2025c; Troshin et al., 2025; Zhang et al., 2025a). Moreover, parallel samples are not conditioned on each other, so there is no mechanism to encourage exploration of alternative interpretations. Saparina & Lapata (2024) similarly considered n completions from beam search as alternatives but found no advantage; our experiments extend their analysis with sampling (which is more diverse than beam search), and a more capable model (Qwen3 4B Instruct), yet reach the same conclusion.

We note that our experiments use standard sampling parameters (temperature 0.7, top-k 20, top-p 0.8, as recommended by the model developers). More sophisticated test-time search strategies over the output space could potentially improve diversity, but this remains an open direction: to our knowledge, there are no established results demonstrating that such approaches successfully recover diverse interpretations of ambiguous inputs.

H.3 Clarification Question Generation

Clarification question generation is a well-studied alternative approach for handling ambiguity: the model identifies ambiguous requests, generates a clarification question to help disambiguate, and returns it to the user for a response. Given the user’s clarification (or after multiple rounds of interaction), the model then responds to the original request.

Our approach can be incorporated into such interactive pipelines in multiple ways. One natural option is to present interpretations (and possible answers) to the user and ask them to select one. This pairs well with clarification questions of the form “Which of the following did you mean?” and can be viewed as a variant of the interactive approach where the user is presented with a multiple-choice selection rather than providing an open-ended response. While open-ended clarification in theory offers flexibility, it introduces practical challenges. For example, in Table 1, the user would need to respond “No, I mean what state ruled it in the 14th century” to obtain the Holy Roman Empire answer but the user may not know enough to answer precisely (e.g., when the user does not have access to the source passage). Alternatively, the user might simply respond “no”, which narrows the interpretation space without fully disambiguating, potentially requiring multiple rounds. These challenges make



Figure 2: Open-ended clarification pipeline: (1) the model reasons about intent and provides possible interpretations and answers; (2) the model generates an open-ended clarification question; (3) the user simulator provides all possible responses to the clarification question; (4) for each response, the model generates a final answer.

clarification question generation an interesting but difficult problem to model and evaluate, especially without access to real users.

Following previous work, we consider user simulation with LLMs (Chen et al., 2025). We implement a multi-stage open-ended clarification pipeline (Figure 2):

1. **Reasoning about Intent.** Given the question and context, the model reasons about possible interpretations and provides them along with candidate answers. This stage uses the approach proposed in this work. We consider three options: using IntentRL, using the original Qwen3 4B Instruct, or skipping this step entirely.
2. **Clarification question generation.** The model generates an open-ended question whose answer would disambiguate the original ambiguous question. When interpretations and answers are available from the first stage, the model can rely on them; however, generating an effective clarification question remains non-trivial even with correct interpretations (e.g., Kobalczyk et al. (2025) define informative questions through information gain). We provide the model with one in-context example, although none of the models (including IntentRL) are specifically trained for clarification question generation. We consider both IntentRL and the original Qwen3 4B Instruct for this stage.
3. **User simulation.** We prompt a model to generate all possible user responses to the clarification question, simulating different users. This allows comparison with our approach. The quality of the user simulator is critical: if it misses plausible options or provides responses that fail to disambiguate, downstream performance suffers. We therefore use the largest model in the Qwen family, Qwen3 235B A22B Instruct and provide one in-context example.
4. **Final answer generation.** For each simulated user response, we generate a final answer (again with one in-context example) using the same model as in the clarification stage (either IntentRL or the original Qwen3 4B Instruct). We then aggregate all answers across simulated responses and compute the metrics described in Section 4.1.

We acknowledge that this pipeline is simplified: we do not train models for any particular stage (e.g., clarification question generation), and we do not include unambiguous questions, as that would require the model to first identify whether clarification is needed. We therefore experiment only with the ambiguous subsets. We provide prompts below.

Clarification Question Generation Instructions

You receive a context, a question, and possible interpretations of the question and their corresponding answers. You need to provide a clarification question that disambiguates the question.

Example:

{example}

Context:

{context}

Question:

{question}

Possible interpretations and answers:
`{interpretations_and_answers}`

General Instructions:

- Ask a clarification question that will help to disambiguate the question.
- Try to identify the single axis of ambiguity (scope, aggregation, filtering condition, entity reference, etc.).
- Ask a clarification question about that axis so the user's reply naturally selects one interpretation.
- Try to ask a concise clarification question that will be easy to understand and answer.

Output Format:

Provide clarification question. Follow the format below:

```
<answer>
[Clarification question]
</answer>
```

User Simulation Instructions

You receive a context, a question, possible interpretations of the question and their corresponding answers and a clarification question. Provide all possible answers to the clarification question.

Example:
`{example}`

Context:
`{context}`

Ambiguous question:
`{question}`

Possible interpretations and answers:
`{interpretations_and_answers}`

Clarification question:
`{clarification_question}`

General Instructions:

- Focus on covering all possible answers to the clarification question. They should cover all interpretations of the question.
- The answers should be concise and to the point. They should be easy to understand and provide a clear disambiguation of the question.

Output Format:

Provide all possible answers to the clarification question. Follow the format below:

```
<answer>
Answer 1: [Answer 1 to the clarification
question]
Answer 2: [Answer 2 to the clarification
question]
...
Answer N: [Answer N to the clarification
question]
</answer>
```

Answer Generation Instructions

You receive a context, a question, possible interpretations of the question and their corresponding answers, a clarification question, and the user's answer to that clarification question. Use the user's clarification answer to narrow down the interpretations and provide the final answer.

Example:
`{example}`

Context:
`{context}`

Ambiguous question:
`{question}`

Possible interpretations and answers:
`{interpretations_and_answers}`

Clarification question:
`{clarification_question}`

User's answer to clarification:
`{clarification_answer}`

General Instructions:

- If the user's clarification answer matches one of the possible interpretations, provide the corresponding pair of interpretation and answer.
- If none of the possible interpretations match the user's clarification answer, write a new interpretation and answer that matches the user's clarification.

Output Format:

Provide the final interpretation and answer that matches the user's clarification. Follow the format below:

```
<answer>
Interpretation: [State interpretation]
[Answer]
</answer>
```

Results Table 16 compares our Interpretation-Answer Pairs approach against the open-ended Clarification Pipeline across both datasets.

Reasoning about Intent	Clarification & Answer Generation	Overlap F1	Judge Recall	Judge Precision	Full Coverage
<i>Interpretation-Answer Pairs</i>					
Qwen3 4B Instruct	—	44.4	62.3	42.7	45.5
IntentRL	—	72.9	78.1	53.5	61.0
<i>Open-ended Clarification Pipeline</i>					
—	Qwen3 4B Instruct	42.6	57.3	38.7	36.6
Qwen3 4B Instruct	Qwen3 4B Instruct	50.6	61.2	46.5	41.5
IntentRL	Qwen3 4B Instruct	66.5	72.3	55.6	52.9
IntentRL	IntentRL	66.9	73.1	56.4	55.3

(a) Comparison with clarification pipeline on Abg-CoQA (ambiguous questions).

Reasoning about Intent	Clarification & Answer Generation	Recall	Precision	Full Coverage
<i>Interpretation-Answer Pairs</i>				
Qwen3 4B Instruct	—	22.5	30.8	3.8
IntentRL	—	82.4	77.5	74.1
<i>Open-ended Clarification Pipeline</i>				
—	Qwen 4B Instruct	36.3	38.8	8.4
Qwen3 4B Instruct	Qwen3 4B Instruct	31.8	33.2	6.1
IntentRL	Qwen3 4B Instruct	74.6	71.7	59.5
IntentRL	IntentRL	76.2	75.0	63.4

(b) Comparison with clarification pipeline on Ambrosia (ambiguous questions).

Table 16: We compare our Interpretation-Answer Pairs approach, which eliminates the need for open-ended clarification, against a traditional Clarification Pipeline. For both settings, we ablate the choice of model (Qwen3 4B Instruct and IntentRL) across pipeline stages. For the Clarification Pipeline, we additionally ablate the inclusion of the reasoning stage (which provides interpretations).

The top section of each subtable shows the direct Interpretation-Answer Pairs setting, which can also be viewed as the first stage of the pipeline in isolation. In the Clarification Pipeline results, we observe several patterns. First, generating clarification questions and answers without any prior reasoning about intent (first row) performs poorly, confirming that identifying possible interpretations is a valuable step to effective clarification. Second, incorporating interpretations from the base Qwen3 4B Instruct model helps on Abg-CoQA, where the initial interpretation quality is reasonable, but hurts on Ambrosia, where the base model’s interpretations are too noisy to be useful. Third, using high-quality interpretations from IntentRL results in clear gains in both datasets. Using IntentRL for clarification and answer generation as well provides a further improvement, as IntentRL has also learned to predict answers more accurately; however, the dominant factor is the quality of the interpretations rather than the answer generation model.

Notably, even the best Clarification Pipeline configuration underperforms the direct Interpretation-Answer Pairs approach with IntentRL. We manually examined typical failure cases and identified three main sources of error propagation: (1) the clarification question targets the wrong axis of ambiguity or fails to disambiguate effectively, (2) the user simulator does not cover all plausible responses, and (3) both user simulation and answer generation can collapse to the same interpretation, missing valid alternatives.

These results suggest that open-ended clarification remains a hard problem that does not outperform our approach while requiring substantially more computation. Nonetheless, high-quality interpretations consistently improve clarification performance, indicating that our method is complementary to interactive pipelines. Furthermore, our proposed approach could in principle be adapted to train clarification question generation directly: the model

would reason about ambiguity but produce clarification questions instead of interpretation-answer pairs, with user simulation and answer generation incorporated into the reward computation. We leave this direction for future work. More broadly, we believe the optimal strategy in practice would be a hybrid approach: presenting interpretations as multiple-choice options while allowing the user to provide an open-ended response if the listed interpretations do not cover their intended meaning.

H.4 Discussion of Ambiguity Handling Strategies

The experiments presented in this section lead to three main conclusions about how our approach compares to alternative strategies for handling ambiguity.

Switching paradigms does not yield strong zero-shot results. A natural question is whether ambiguity handling can be achieved simply by adopting a different output format, that is, whether existing models already handle ambiguity well under the right prompting paradigm. Our results show this is not the case. Across all three alternative formats (answer-only generation, parallel sampling, and clarification question generation), zero-shot performance with prompting alone remains significantly below that of IntentRL. These findings confirm that IntentRL does not merely train models to produce a particular output format when they already exhibit strong ambiguity awareness. Rather, the training itself is necessary: it teaches models to reason about user intent and systematically explore diverse interpretations.

No alternative format provides a clear advantage; each involves trade-offs. We find that none of the alternative paradigms offers a uniformly better solution than interpretation-answer pairs. Answer-only generation achieves competitive results, but sacrifices transparency: without interpretations, predictions lack justification, which is particularly problematic in downstream applications such as text-to-SQL where the generated query may be difficult to understand on its own. Parallel sampling avoids sequential conditioning but consistently collapses to the same interpretation, failing to recover the diversity that our approach achieves through explicit reasoning about alternatives. Open-ended clarification questions offer user control but introduce greater complexity: they require multiple interaction rounds, depend on the quality of user simulation, and suffer from error propagation across pipeline stages. Even the best clarification pipeline configuration underperforms direct interpretation-answer generation with IntentRL (Table 16), with failures arising from poorly targeted clarification questions, incomplete user simulation, and answer collapse. A key limitation is the constrained number of interpretations (five in our work). For extremely vague questions with large interpretation spaces, clarification questions may be more appropriate. However, we believe the optimal strategy likely involves a hybrid: presenting interpretations as multiple-choice options while allowing users to provide open-ended responses when the listed options do not capture their intent.

Our approach integrates effectively with alternative formats. Despite the trade-offs of each paradigm in isolation, our approach integrates well with both the answer-only and clarification formats. The interpretation-answer format naturally generalises to the answer-only setting, which can be viewed as an ablation where interpretations are simply omitted; both variants benefit from IntentRL training. More notably, high-quality interpretations from IntentRL consistently improve the clarification pipeline: incorporating IntentRL’s reasoning about intent into the first stage of the pipeline leads to clear gains on both Abg-CoQA and Ambrosia, regardless of which model is used for subsequent clarification and answer generation (Table 16). This demonstrates that learning to reason about user intent is complementary to interactive approaches rather than competing with them. In principle, our training framework could be extended to directly optimise clarification question generation, with user simulation and answer quality incorporated into the reward signal.

More broadly, an important open question is which response format is actually preferred by users, and whether the optimal format varies across tasks, domains, and interaction contexts. For instance, interpretation-answer pairs may be most useful in technical settings where transparency matters (e.g., text-to-SQL), while clarification questions may be preferred in

conversational scenarios, and answer-only generation may suffice when brevity is prioritised. Our experiments evaluate these formats through automatic metrics, but user preferences may not align with metric-level performance. Conducting user studies to understand how people interact with different ambiguity handling formats, and whether the preferred format depends on factors such as expertise, task complexity, or time constraints, is an interesting direction for future work.

I Human Annotation Details

For the interpretation-answer alignment evaluation, we recruited annotators through Prolific⁵ with the following criteria. We required annotators to have English as their primary language (and SQL knowledge for Ambrosia) and at least a 95% approval rate on the platform. Annotators were not allowed to retake the task. To reduce cognitive load and make the task more efficient, we grouped interpretation-answer pairs by their source example. Each annotator evaluated 3 examples, with each example containing multiple interpretation-answer pairs from the same context. We paid annotators £9 per hour, following Prolific’s fair wage guidelines. The average completion time was 15 minutes. We used the POTATO annotation tool (Pei et al., 2022).

J Limitations

Our work has several limitations that suggest directions for future research.

Dataset coverage While we demonstrate positive out-of-domain generalisation for text-to-SQL, we lack suitable out-of-domain conversational question answering datasets. Our approach requires multiple valid answers per input, which excludes datasets like PACIFIC (Deng et al., 2022), where each ambiguous turn is paired with a *single* clarification question and its corresponding answer in the following turn. We also do not consider open-domain multi-answer datasets such as AmbigQA (Min et al., 2020) and ASQA (Stelmakh et al., 2022), because state-of-the-art systems for these datasets typically rely on retrieval (e.g., In et al. (2025)). This introduces orthogonal challenges beyond our focus on contextualised ambiguity resolution. Appendix G reports exploratory closed-book results on AmbigQA, where IntentRL outperforms both baselines, though absolute scores conflate method gains with parametric knowledge. Integrating IntentRL with retrieval-based approaches is a promising direction for future work.

Backbone model requirements Like many RL-based methods, our approach requires a sufficiently strong backbone model. Weak models may fail to produce valid completions that receive non-zero rewards during training, preventing effective learning. However, the model does not need to be good at ambiguity handling prior to training: on Ambrosia, the base model achieves only 3.8% full coverage with prompting, while IntentRL improves this to 74.1%, demonstrating that the ability to reason about ambiguity can be learned through training even when it is largely absent initially.

Inference cost Our model produces all interpretations and answers in *one pass*, without a user in the loop, unlike clarification-based methods which require an extra round of interaction to resolve the ambiguity, or two-stage pipelines that run a separate model sequentially. This does not imply fewer tokens: non-reasoning methods can be more token-efficient, and reasoning about intent may not be warranted for requests that are clearly unambiguous.

Response format evaluation We acknowledge that our interpretation-answer format is not universally optimal. It is constrained to a fixed number of interpretations (five in our work), which may be insufficient for highly vague questions with large interpretation spaces, particularly open-ended questions that lack the grounding context our datasets provide. At

⁵<https://www.prolific.com/>

the same time, it offers practical benefits: it provides explicit justifications for each answer, requires no additional annotation beyond multiple answers, and produces structured output that is easy to parse for downstream applications. We discuss further trade-offs between alternative formats in Appendix H.4. We acknowledge that our comparison with alternative ambiguity handling formats is based entirely on automatic metrics. We do not conduct human studies to evaluate which format users actually prefer, or whether preferences vary across tasks, domains, and user expertise. As discussed in Appendix H.4, understanding user preferences for different response formats remains an open question and an important direction for future work.

K Model Outputs

Tables 17 and 18 show full examples corresponding to those presented in Table 1, including model outputs with reasoning traces and interpretation-answer pairs. In the conversational QA example (Table 17), the model identifies all states that ruled Silesia in chronological order and generates five interpretations covering different time periods. The reasoning trace is notably repetitive, revisiting the same list of states multiple times before arriving at the final answer, illustrating a pattern we observe frequently in Abg-CoQA outputs (see Section 5.5). In the text-to-SQL example (Table 18), the model correctly identifies the attachment ambiguity in “horror films and thriller films with a budget of 20 million” and generates SQL queries with different operator precedence. However, it also illustrates the redundancy issue discussed in Section 5.5: interpretations 1, 3, 4, and 5 express the same logical intent with only syntactic SQL variations, while the core ambiguity (whether the budget constraint applies to both genres or only thrillers) is captured by the difference between queries with and without parentheses in the WHERE clause. In practice, such redundant interpretations can be filtered out by comparing execution results.

Silesia is a region of Central Europe located mostly in Poland, with small parts in the Czech Republic and Germany... The first known states to hold power there were probably those of **Greater Moravia** at the end of the 9th century and **Bohemia** early in the 10th century. In the 10th century, Silesia was incorporated into the **early Polish state**, and after its division in the 12th century became a **Piast duchy**. In the 14th century, it became a constituent part of the Bohemian Crown Lands under the **Holy Roman Empire**, which passed to the **Austrian Habsburg Monarchy** in 1526.

— What are the other two countries it is part of? — the Czech Republic and Germany.
— When did Bohemia rule there? — the 10th century
— **What other state ruled it?**

Reasoning Trace

Analysis and reasoning:

The question “What other state ruled it?” is ambiguous because “it” refers to Silesia, which was ruled by multiple states over time. The previous question asked when Bohemia ruled there, and the context states that Bohemia ruled Silesia “early in the 10th century.” The question now asks for “other states” that ruled Silesia. The context mentions several states that ruled Silesia in sequence:

- Greater Moravia (end of 9th century)
- Bohemia (early 10th century)
- Early Polish state (10th century)
- Piast duchy (after division in 12th century)
- Bohemian Crown Lands under the Holy Roman Empire (14th century)
- Austrian Habsburg Monarchy (in 1526)

The question “What other state ruled it?” implies that Bohemia is already known as a ruling state, and the user is asking for another state that ruled Silesia. Possible interpretations:

1. “It” refers to Silesia, and “other state” means other than Bohemia. The context states that Bohemia ruled Silesia “early in the 10th century,” and after that, Silesia was incorporated into the early Polish state. So, the early Polish state is a candidate. Also, later, it was ruled by the Austrian Habsburg Monarchy.
2. “It” could refer to a different entity. However, the previous context clearly refers to “it” as Silesia.
3. The phrasing “what other state” implies a plural or single additional state. The context states that Bohemia ruled Silesia early in the 10th century, and after that, it became a constituent part of the Bohemian Crown Lands under the Holy Roman Empire, which passed to the Austrian Habsburg Monarchy in 1526.

Therefore, the states that ruled Silesia after Bohemia include: Early Polish state, Piast duchy, Bohemian Crown Lands under the Holy Roman Empire, Austrian Habsburg Monarchy.

Key possible “other states” ruled Silesia: The early Polish state, The Piast duchy, The Holy Roman Empire (via Bohemian Crown Lands), The Austrian Habsburg Monarchy.

Since the question is ambiguous, possible interpretations: “What other state ruled it?” could mean: What other state succeeded Bohemia? Answer: early Polish state. What other state ruled it after Bohemia? Answer: the early Polish state. What other state ruled it in the 14th century? Answer: the Holy Roman Empire. What other state ruled it after Bohemia? Answer: Austrian Habsburg Monarchy. What other state ruled it? (plausible answer: the early Polish state).

From the context: “In the 10th century, Silesia was incorporated into the early Polish state, and after its division in the 12th century became a Piast duchy. In the 14th century, it became a constituent part of the Bohemian Crown Lands under the Holy Roman Empire, which passed to the Austrian Habsburg Monarchy in 1526.”

Thus, possible “other states” that ruled it include: early Polish state, Piast duchy, Holy Roman Empire (as ruler of Bohemian Crown Lands), Austrian Habsburg Monarchy.

Possible interpretations of the question: 1. “What other state ruled it?” after Bohemia — early Polish state. 2. What other state ruled it after Bohemia? — the early Polish state. 3. What other state ruled it? (alternative to Bohemia) — the early Polish state. 4. What other state ruled it? (after Bohemia) — the Piast duchy. 5. What other state ruled it? (in the 14th century) — the Holy Roman Empire.

Given that the previous question was “when did Bohemia rule there?” and the answer was “the 10th century,” the current question is asking for another state that ruled Silesia. The most direct answer is the early Polish state, as it is the first state mentioned after Bohemia.

Other possible states: The early Polish state, The Piast duchy, The Holy Roman Empire, The Austrian Habsburg Monarchy.

Final interpretations: “What other state ruled it?” — refers to the state that ruled Silesia before Bohemia: Greater Moravia. “What other state ruled it?” — refers to the state that ruled Silesia after Bohemia: early Polish state. “What other state ruled it?” — refers to the state that ruled it in the 14th century: Holy Roman Empire. “What other state ruled it?” — refers to the state that ruled it after Bohemian Crown Lands: Austrian Habsburg Monarchy. “What other state ruled it?” — refers to the Piast duchy.

The safest and most clear interpretation is that the question is ambiguous and “it” refers to Silesia, and the user is asking for another state that ruled it besides Bohemia. The context says Bohemia ruled early in the 10th century, and before that, Greater Moravia and Bohemia were early rulers. The state that ruled it after Bohemia was the early Polish state.

Thus, possible answers: early Polish state, Piast duchy, Holy Roman Empire, Austrian Habsburg Monarchy, Greater Moravia.

But the context says: “The first known states to hold power there were probably those of Greater Moravia at the end of the 9th century and Bohemia early in the 10th century.” So, Greater Moravia ruled before Bohemia.

Therefore, the states that ruled Silesia in sequence: 1. Greater Moravia, 2. Bohemia, 3. Early Polish state, 4. Piast duchy, 5. Bohemian Crown Lands under Holy Roman Empire, 6. Austrian Habsburg Monarchy.

“Other state” — possible answers: Greater Moravia, early Polish state, Piast duchy, Holy Roman Empire, Austrian Habsburg Monarchy.

Final interpretations: The question is ambiguous: “it” refers to Silesia, and “what other state ruled it?” could refer to any state besides Bohemia. Possible states: Greater Moravia, early Polish state, Piast duchy, Holy Roman Empire, Austrian Habsburg Monarchy.

Since the context directly states: “In the 10th century, Silesia was incorporated into the early Polish state, and after its division in the 12th century became a Piast duchy,” these are clear.

Thus, possible interpretations: 1. What other state ruled it (besides Bohemia) before the 10th century? — Greater Moravia. 2. What other state ruled it after Bohemia? — early Polish state. 3. What other state ruled it in the 12th century? — Piast duchy. 4. What other state ruled it in the 14th century? — Holy Roman Empire. 5. What other state ruled it after the Bohemian Crown Lands? — Austrian Habsburg Monarchy.

Given that the question is ambiguous, and based on the context, the intended meaning is likely to find the state that ruled Silesia directly after Bohemia, which is the early Polish state.

However, the context states that Silesia was incorporated into the early Polish state in the 10th century. So, the answer “early Polish state” is valid.

Thus, possible interpretations and answers: “What other state ruled it?” — besides Bohemia — Greater Moravia. “What other state ruled it?” — besides Bohemia — early Polish state. “What other state ruled it?” — besides Bohemia — Piast duchy. “What other state ruled it?” — besides Bohemia — Holy Roman Empire. “What other state ruled it?” — besides Bohemia — Austrian Habsburg Monarchy.

Final selection (maximum 5 interpretations):

Interpretation–Answer Pairs

Interpretation 1: The question refers to the state that ruled Silesia before Bohemia.
Greater Moravia

Interpretation 2: The question refers to the state that ruled Silesia after Bohemia in the 10th century.
early Polish state

Interpretation 3: The question refers to the state that ruled Silesia after becoming a Piast duchy.
Piast duchy

Interpretation 4: The question refers to the Holy Roman Empire, which ruled Silesia in the 14th century.
Holy Roman Empire

Interpretation 5: The question refers to the state that ruled Silesia after the Bohemian Crown Lands.
Austrian Habsburg Monarchy

Table 17: Model output for an ambiguous conversational QA example from Abg-CoQA.

```

CREATE TABLE Directors(
  id INTEGER PRIMARY KEY, name TEXT,
  birth_year INTEGER, nationality TEXT);
CREATE TABLE Film_Director(
  film_id INTEGER, director_id INTEGER,
  PRIMARY KEY(film_id, director_id));
CREATE TABLE Film_Genre(
  film_id INTEGER, genre_id INTEGER,
  PRIMARY KEY(film_id, genre_id));
CREATE TABLE Films(
  id INTEGER PRIMARY KEY, title TEXT,
  release_date DATE, director TEXT, budget REAL);
CREATE TABLE Genres(
  id INTEGER PRIMARY KEY, genre TEXT);

```

Show horror films and thriller films with a budget of 20 million. Show them in one table.

Reasoning Trace

The question is:

“Show horror films and thriller films with a budget of 20 million. Show them in one table.”

Step-by-step analysis:

1. Identify the films to be shown: The question asks for “horror films and thriller films”. In the context, “Horror” and “Thriller” are genres stored in the Genres table. Films are linked to genres via the Film_Genre table, which connects film_id to genre_id. So, we must find films that belong to either the Horror (genre_id = 1) or Thriller (genre_id = 2) genre.

2. Filter by budget: “With a budget of 20 million” — this clearly refers to films where the budget = 20000000.0. The Films table has a budget column of type REAL. So, we are to filter films that have both: Genre = Horror or Thriller, Budget = 20,000,000.0

3. What does “Show them” mean? The phrase “Show horror films and thriller films” could mean: List the titles of the films that are either horror or thriller and have a budget of 20 million. Show only the titles (as the films themselves). The next part: “Show them in one table” is ambiguous: It might mean: “Return the results in a single result set” (i.e., combine both horror and thriller films with the budget condition into one output). It might mean: “Show the films (titles) in a single table” — which could imply a query that returns titles from the Films table, filtered appropriately. Alternatively, it could be interpreted as: “Bring together (join) the horror and thriller films that have the budget into a single table (i.e., a query that returns one list of titles)”. Another possible interpretation: “Create a view or a result set that contains horror and thriller films with a budget of 20 million” — which again leads to filtering films by both genre and budget.

4. What exactly is being requested to be shown? The phrase “Show horror films and thriller films” could mean: Show titles of films that are horror or thriller — i.e., union both. Or, show films that are both horror and thriller (intersection), but that seems less likely given “and” in the context of genres (as a list of films that belong to either). However, “horror films and thriller films” typically means films that are in either of those genres — i.e., union. Also, the phrase “with a budget of 20 million” applies to both — so we are filtering the horror and thriller films *that also have a budget of 20 million*.

5. “Show them in one table”: This could mean: Return the film titles in a single result set (a table with one column, the title). It may imply that the result should be treated as a single table (i.e., a simple SELECT query returning titles). Alternatively, it could be asking to join or combine the horror and thriller films with budget 20M into a single output — which is simply returned in one query.

6. Scope of the data: The only data we can extract from the schema is the titles of films that: Are in genre Horror (id = 1) or Thriller (id = 2), Have a budget of 20,000,000.0. We must retrieve only the film titles, as “show horror films” likely means showing the titles of those films.

7. Ambiguity in “films” — should it be all films or only those that are both horror and thriller? “Horror films and thriller films” can be read as: Films that are horror films **and** thriller films (intersection) → films that belong to both genres. Films that are horror films **or** thriller films (union) → films in either genre. In everyday language, “horror films and thriller films” often means the collection of films in either category — i.e., union. However, in database queries, “and” can sometimes denote intersection. So we must consider both possibilities.

8. Ambiguity in “show them in one table”: Could mean: return the titles in a single column (a one-column table). Could mean: create a view or a combined listing (still just one query). It might simply mean: return the results as a single result set (which is standard in SQL).

Interpretation–Answer Pairs

Interpretation 1: Show the titles of horror films and thriller films that have a budget of 20 million, displaying them in a single table (i.e., a list of film titles).

```
SELECT f.title FROM Films f JOIN Film_Genre fg ON f.id = fg.film_id WHERE fg.genre_id = 1 OR fg.genre_id = 2 AND f.budget = 20000000.0;
```

Interpretation 2: Show the titles of films that are both horror and thriller (i.e., films in both genres) and have a budget of 20 million, in one table.

```
SELECT f.title FROM Films f JOIN Film_Genre fg ON f.id = fg.film_id WHERE fg.genre_id = 1 AND fg.genre_id = 2 AND f.budget = 20000000.0;
```

Interpretation 3: Show the titles of horror and thriller films with a budget of 20 million, combined in one result set (i.e., union of horror and thriller films with that budget).

```
SELECT f.title FROM Films f JOIN Film_Genre fg ON f.id = fg.film_id WHERE (fg.genre_id = 1 OR fg.genre_id = 2) AND f.budget = 20000000.0;
```

Interpretation 4: Show the titles of films that are either horror or thriller and have a budget of 20 million, in a single table (i.e., a combined list).

```
SELECT f.title FROM Films f JOIN Film_Genre fg ON f.id = fg.film_id JOIN Genres g ON fg.genre_id = g.id WHERE g.genre = 'Horror' OR g.genre = 'Thriller' AND f.budget = 20000000.0;
```

Interpretation 5: Show the titles of horror and thriller films with a budget of 20 million, in one table, using the film titles only.

```
SELECT f.title FROM Films f JOIN Film_Genre fg ON f.id = fg.film_id WHERE fg.genre_id IN (1, 2) AND f.budget = 20000000.0;
```

Table 18: Model output for an ambiguous text-to-SQL example from Ambrosia.