Instruction Ambiguity in LLM Agents: An Analysis of Internal Representations and Text Outputs

2330003 Naohiro Kaide

Large Language Models (LLMs) are increasingly transitioning from static roles in information retrieval and question answering to dynamic agents capable of intervening in real-world environments. In these agentic scenarios, the ability to accurately interpret human instructions is critical. However, user instructions in practical settings are frequently imperfect; they may lack necessary details or contradict the current state of the environment. Such scenarios result in "instruction ambiguity", which can be categorized into two primary types: ambiguity arising from insufficient information (multiplicity of interpretation) and ambiguity arising from grounding failures (contradictions with the environment). For an agent operating in a physical or simulated world, acting upon an arbitrary interpretation of an ambiguous command poses significant risks, potentially leading to irreversible errors or accidents. Consequently, the ability to detect when an instruction is ambiguous and requires clarification—rather than execution—is a fundamental requirement for safe agent deployment.

Existing research on this problem has predominantly concentrated on engineering solutions to improve the output text, such as developing high-precision ambiguity detectors using supervised learning or optimizing prompt strategies to encourage clarification questions. While these approaches address the surface-level behavior of the model, they treat the LLM as a black box. There remains a significant gap in understanding the internal mechanisms governing how LLMs process and represent instruction ambiguity within their hidden states. It is unclear whether the models fail to detect ambiguity because they lack the internal representation of it, or if they possess the relevant information but fail to express it due to generation biases. To ensure the reliability of LLM-based agents, it is necessary to investigate these internal processing mechanisms.

This study addresses this gap by analyzing the internal states of instruction-tuned LLMs, specifically the Qwen3 (4B, 8B, 14B) and Gemma-3 (4B, 12B) series. We employed the Interactive Grounded Language Understanding (IGLU) 2022 dataset, which focuses on a 3D block arrangement task. This environment serves as a proxy for real-world tasks where an agent must interpret natural language instructions relative to a visual grid world. The dataset was restructured to allow for the simultaneous evaluation of task execution success (for clear instructions) and ambiguity detection (for unclear instructions). The experimental design involved feeding the models with inputs consisting of task definitions, environmental states (grid configurations),

dialogue history, and user instructions. We then compared the models' text outputs against the information encoded in their internal representations.

To analyze the internal representations, we utilized linear probing, a technique from mechanistic interpretability. This involved training simple logistic regression classifiers on the activation vectors of the models' intermediate layers to predict whether a given instruction was clear or ambiguous. We examined the internal states at four distinct positions in the inference process: the end of the prompt ('promp_end'), the start of the planning phase ('plan_start'), the end of the thinking process when Chain-of-Thought is used ('thinking_end'), and the end of the final output ('output_end'). This internal analysis was contrasted with the models' behavioral performance, measured by the Exact Match (EM) score for task execution and the Macro F1 score for ambiguity detection based on the generated text.

We investigated the impact of three specific prompt engineering factors on both internal and external performance: the presence of "hints" (explicit instructions to report ambiguity), the use of Chain-of-Thought (CoT) reasoning, and Few-shot prompting (including examples of valid and ambiguous instructions). The analysis of the text outputs revealed that while CoT and Few-shot prompting consistently improved the task success rate for clear instructions, they did not necessarily enhance the models' ability to reject ambiguous instructions via text. In conditions without explicit ambiguity hints, most models exhibited poor detection performance, often performing worse than random prediction. This behavior suggests a strong "sycophancy bias", where the model prioritizes complying with the user's request and generating an executable plan, even when the instruction is fundamentally flawed or executable in multiple mutually exclusive ways.

The linear probing results provided a contrasting perspective. We found that information regarding instruction ambiguity is encoded in a linearly separable manner within the models' intermediate to deeper layers. Even in conditions where the text output failed to identify ambiguity (resulting in low F1 scores), the linear probes trained on the internal states achieved significantly higher accuracy. This discrepancy demonstrates that the models internally distinguish between clear and ambiguous instructions to a greater extent than their generated text implies. The information exists within the high-dimensional vector space of the model but is lost or suppressed during the transformation into the final token sequence, likely overridden by the objective to generate valid command syntax.

Furthermore, our analysis of the Chain-of-Thought mechanism yielded a counter-intuitive finding. While CoT is widely recognized for improving complex reasoning capabilities, our results showed that it does not improve the quality of the internal representation of ambiguity. In fact, for the Qwen3-

14B model, while CoT drastically improved the execution of clear tasks, the internal classification accuracy for ambiguity remained static or slightly declined compared to standard prompting. This suggests that the reasoning steps generated during CoT primarily serve the purpose of task decomposition and plan formulation (execution capability) rather than the critical evaluation of the instruction's validity (recognition capability). The internal representation of ambiguity appears to be formed relatively early in the processing of the input and is not significantly refined by the subsequent generation of reasoning steps.

Comparing the layer-wise performance, we observed that ambiguity information typically peaks in the middle-to-late layers. In most settings, this information degrades in the final layers as the model prepares to emit specific tokens, supporting the hypothesis that the generation objective overshadows the detection signal. However, an exception was observed in the Qwen3-14B model when explicit hints were provided; in this specific case, the ambiguity information was preserved effectively up to the final output layer, suggesting that specific prompting strategies combined with sufficient model scale can help align the internal representation with the external output.

This work represents the first analysis of agent instruction ambiguity at the internal representation level. These findings provide a necessary foundation for detailed future studies of ambiguity in LLM-based agents.