Structured Outputs in Prompt Engineering: Enhancing LLM Adaptability on Counterintuitive Instructions

Jingjing Ye
yejj177@gmail.com
Independent Researcher

Song Bai
song.bai6174@gmail.com
Independent Researcher

Zhenyang Li
zli2022@alumni.usc.edu
Independent Researcher

Zheqi Shen
zheqi@shen.zone
Independent Researcher

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language processing tasks, yet they often exhibit cognitive inertia, rigidly adhering to ingrained training conventions even when prompted to deviate. This paper investigates the efficacy of structured output techniques in prompt engineering to mitigate such inertia and improve instruction-following on counterintuitive tasks. We argue that using the structured input and output with our framework yields significant performance gains, studied on the Inversed IFEval dataset across varying prompts and domains. This work contributes to the growing field of prompt engineering research by demonstrating structured outputs as a robust method for enhancing LLM logical reasoning.

1 Introduction

The recent advancements of Large Language Models (LLMs) have revolutionized artificial intelligence, enabling sophisticated applications in natural language understanding, generation, and reasoning. However, a persistent challenge is their tendency toward cognitive inertia, a phenomenon where models persist in following learned patterns from pre-training and fine-tuning, resisting deviations even under explicit instructions. This inertia manifests in scenarios requiring unlearning or counterintuitive behavior, such as generating flawed outputs intentionally or ignoring standard formatting conventions. The logic twist inside might be easy for elementary school students, but is proven difficult for LLM models, a factor critical in scientific reasoning.

Prompt engineering emerges as a non-invasive method to guide LLMs without retraining, encompassing techniques like zero-shot (Kojima et al., 2022; Li, 2023), few-shot (Dang et al., 2022), and chain-of-thought (CoT) prompting (Lyu et al., 2023; Zhang et al., 2024). Among these, structured outputs, which enforce responses in predefined formats such as JSON, XML or phased structures, offer verifiability and consistency, while reducing hallucinations and improving reliability. Recent advancements, including OpenAI's Structured Outputs feature, underscore their practical utility in production environments.

To evaluate these techniques on counterintuitive tasks, we employ the Inverse IFEval dataset (Zhang et al., 2025), an extension of the IFEval benchmark that inverts verifiable instructions to probe unlearning capabilities. The dataset includes challenges like Question Correction (answering incorrectly on purpose), Intentional Textual Flaws (introducing errors), Mid-turn Instruction Modification, and others, spanning diverse domains and languages.

Our contributions are: 1. We develop a framework that utilizes structured outputs to improve LLM responses to counterintuitive instructions; 2. We evaluate three structured prompts with varying output formats and determine that the list-structured approach performs best; 3. We investigate variants of the list-structured method and study the performance impact of explicit prioritization; 4. We test our approach on the Inverse IFEval (Zhang et al., 2025) benchmark and demonstrate that our list-structured prompting framework largely outperforms baselines, providing insights for more adaptable and logical AI systems.

2 Related Works

2.1 Prompt Engineering Techniques

Prompt engineering has evolved from basic input crafting to sophisticated strategies for eliciting optimal LLM responses. Surveys categorize techniques into zero/few-shot prompting (Dang et al., 2022; Kojima et al., 2022; Li, 2023), CoT (Lyu et al., 2023; Zhang et al., 2024), ToT (Yao et al., 2023; Mo and Xin, 2024; Ranaldi et al., 2024), and self-consistency methods (Zhou et al., 2025; Tauben-

Proceedings of the Third Workshop for Artificial Intelligence for Scientific Publications, pages 115–120

feld et al., 2025; Nowak, 2025). CoT, for instance, encourages step-by-step reasoning, while ToT explores multiple paths for complex problem-solving. Structured prompting extends these by imposing formats, such as role-playing or output schemas, to enhance control and parseability.

2.2 Cognitive Inertia and Unlearning in LLMs

LLMs exhibit human-like cognitive effects, including priming, anchoring (Lou and Sun, 2024), and irrational biases (Echterhoff et al., 2024; Tang and Kejriwal, 2024) in decision-making tasks. Cognitive inertia, a form of resistance to change, is particularly evident in instruction-following scenarios where models default to "helpful" behaviors despite contrary prompts. LLMs exhibit cognitive inertia, reflecting a persistent adherence to patterns learned during self-supervised pre-training. (Resnik, 2025) observes that biases in LLMs are not merely a result of training data, but are intrinsically embedded within the model architecture and optimization objectives. Optimizing for next-token prediction causes models to internalize statistical regularities, including societal biases, without distinguishing between high-probability patterns and harmful conventions. Humans, in contrast, can flexibly adjust behavior via metacognition, reasoning, and contextual judgment, enabling them to follow counterintuitive instructions. LLMs, lacking autonomous reasoning or self-correction, struggle to overcome entrenched patterns even when fine-tuned or aligned through RLHF. Cognitive inertia thus arises from the interaction of pre-training habits, modeling constraints, and limited post-hoc flexibility, leading models to reproduce established patterns rather than adapt to out-of-distribution tasks. One potential approach to mitigate this issue is to reconstitute LLMs' internal representations as structured representations, encoding entities, relations, logical structure, and distinctions between meaning, normativity, and factuality, thereby enhancing the model's flexibility in adapting to novel or counterintuitive instructions.

Unlearning benchmarks like TOFU, MUSE, WMDP, and RWKU assess models' ability to forget specific knowledge while retaining general capabilities. However, critiques highlight flaws in these benchmarks, such as over-optimistic evaluations due to separate testing of forget/retain queries.

2.3 Benchmarks for Instruction Following

Inverse IFEval (Zhang et al., 2025) is a new benchmark for testing counterintuitive adherence. It inverts the paradigm in IFEval (Zhou et al., 2023) that evaluates verifiable instructions. Constructed via human-in-the-loop processes, the inverse IFEval reveals that larger, instruction-tuned models paradoxically struggle more with deviations. Gaps persist in integrating structured prompting into such benchmarks, which our work addresses by proposing a verifiable framework.

3 Methodology

Our methodology centers on developing and testing a structured output framework designed to enhance LLMs' ability to follow counterintuitive instructions from the Inverse IFEval dataset. This framework decomposes the instruction-following process into four explicit phases: Instruction Parsing, Requirement Checklist, Structured Response and Self-Check. We explore multiple variants of this framework to identify the most effective implementation.

3.1 Dataset and Task Description

We evaluate our approach on the Inverse IFEval dataset, a challenging benchmark with 1012 high-quality samples designed to test LLMs' ability to follow counterintuitive instructions that contradict their training patterns. The dataset covers eight distinct instruction types: (1) Instructional Induction, (2) Mid-turn Instruction Modification, (3) Counterfactual Answering, (4) Counter-Conventional Formatting, (5) Question Correction, (6) Deliberately Incorrect Answers, (7) Intentional Textual Flaws, and (8) Code without Comments. These types span diverse domains and require models to override ingrained behaviors such as being helpful, following conventions, and producing polished outputs.

The dataset includes both English and Chinese subsets, enabling cross-lingual evaluation. For our experiments, we use stratified sampling to select 40-48 representative samples, ensuring balanced coverage across all eight instruction types. This sample size balances computational feasibility with statistical reliability while maintaining type diversity for robust evaluation.

3.2 Variants on Structured Approach

```
Before responding to this instruction, first analyze and
      structure it into clear components:

**Original Instruction**: {instruction}

**Step 1 - Parse the Instruction:**
Break down the instruction into these components:
- **Condition**: Any context, assumptions, or conditional
      statements
- **Questions**: The core tasks or questions being asked
- **Requirements**: Specific formatting, style, or content
      constraints
- **Distribution**: Length, structure, or organizational
      requirements

**Step 2 - Structured Analysis:**
Condition: [Extract any context or conditions]
Questions: [Identify the main task]
Requirements: [List all specific constraints]
Distribution: [Note any length/structure requirements]

**Step 3 - Systematic Response:**
Now provide your response, ensuring you address each
      component systematically:
```

Structured Prompt 1: Basic Structured Approach

We investigate varying structured prompts and adapt them to counterintuitive instructions. We begin with three initial variants: (1) a basic structured approach, Prompt 1, which applies the four phases in a simple textual format without additional enhancements, (2) JSON-structured prompting, Prompt 2, which organizes the components into machine-readable JSON fields (e.g., {"context": "...", "tasks": "..."}) for improved parseability and verifiability, and (3) checklist-based prompting, Prompt 3, which uses enumerated lists to break down requirements, promoting systematic adherence. These variants allow us to assess the impact of different structuring mechanisms on model performance.

For the checklist variant, we further investigate two sub-types: an equal checklist Prompt 3, where all requirements are treated uniformly without explicit prioritization, and a priority checklist Prompt 3, where items are categorized as CRITICAL (essential for compliance), IMPORTANT (affecting quality), or SECONDARY (enhancing completeness). This prioritization is intended to guide the model in focusing on high-impact elements first, potentially reducing cognitive inertia by emphasizing core constraints.

````
Parse this instruction into structured components, then
      respond:

**Instruction**: {instruction}

**Step 1: JSON Structure Analysis**
Parse the instruction into this JSON format:
```json
{{
  "context": "any background or situational information",
  "tasks": "the core tasks or questions",
  "format_requirements": ["list", "of", "formatting",
      "constraints"],
  "content_requirements": ["list", "of", "content",
      "constraints"],
  "length_requirements": "any length or size constraints",
  "style_requirements": "any tone or style requirements"
}}
```

**Step 2: Component-by-Component Response**
Now respond to the instruction, explicitly addressing each
      JSON component:

**Context addressed**: [How you handle the context]
**Task completion**: [Your core response]
**Format compliance**: [How you meet format requirements]
**Content compliance**: [How you meet content requirements]
**Length compliance**: [How you meet length requirements]
**Style compliance**: [How you meet style requirements]

**Final Response**:
````

Structured Prompt 2: JSON-Structured Prompting

```
You will respond to this instruction using a systematic
      parsing approach:

**Instruction to Analyze**: {instruction}

**Phase 1: Instruction Parsing**
Parse the instruction and identify:
- Context/Conditions: What situation or context is
      established?
- Core Tasks: What are the main things being asked?
- Format Requirements: Any specific formatting constraints?
- Content Requirements: What must be included/excluded?
- Length/Structure: Any size or organizational requirements?

**Phase 2: Requirement Checklist**
List each requirement as a checkable item (with priority):
- Requirement 1: [First constraint]
- Requirement 2: [Second constraint]
- Requirement 3: [Third constraint]
[Add more as needed]

**Phase 3: Structured Response**
Provide your response while explicitly addressing each
      requirement:

[Your response here]

**Phase 4: Self-Check**
Verify your response against each requirement (with priority):
- Requirement 1: Y/N [Brief check]
- Requirement 2: Y/N [Brief check]
- Requirement 3: Y/N [Brief check]
```

Structured Prompt 3: Checklist-Based Prompting

The baseline condition presents the original Inverse IFEval instructions directly to the models without any modifications, and serves as a control to measure the added value of our structured approaches. We evaluate these methods across five diverse LLMs: DeepSeek-Chat, Qwen, Gemini-2.5 Pro, o1-preview, and Claude-3.5-Sonnet, selected for their varying sizes and architectures to ensure generalizability. All models are accessed via the OpenRouter API with consistent generation parameters (temperature=1.0, max_tokens=4096, top_p=1.0, frequency_penalty=0, presence_penalty=0) to facilitate fair comparisons.

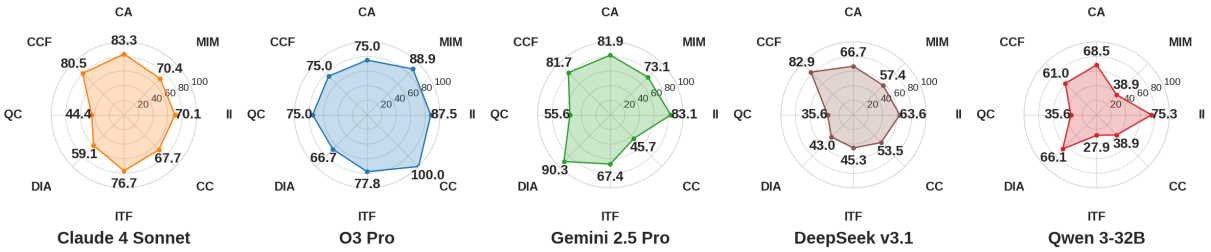

Figure 1: Performance breakdown by instruction type for five models using structured checklist approach. Each radar chart displays accuracy percentages across eight instruction types: II (Instructional Induction), MIM (Mid-turn Instruction Modification), CA (Counterfactual Answering), CCF (Counter-Conventional Formatting), QC (Question Correction), DIA (Deliberately Incorrect Answers), ITF (Intentional Textual Flaws), and CC (Code without Comments).

4 Experiments

Our experiments follow a sequential design to iteratively refine and validate the structured framework. We first test the three initial variants (basic, JSON, and checklist) on a subset of 32 samples from the Inverse IFEval dataset across the selected models.

4.1 Experiment Setup

Evaluation employs an LLM-as-a-Judge paradigm using Claude-4.5-Sonnet (temperature=0) for impartial, binary scoring (1 for semantic match with the reference answer, 0 otherwise). We use a subset of 32 samples for initial variant comparisons and the full set for final assessments. Statistical analysis includes paired t-tests ($\alpha = 0.05$) and Cohen's d for effect sizes, ensuring robust interpretation of results.

The performance comparison, summarized in Table 1, reveals that the checklist-based approach consistently outperforms the basic and JSON variants, achieving higher average accuracy. This suggests that the enumerated, human-readable format of checklists better mitigates cognitive inertia by enforcing explicit requirement tracking.

Method	Accuracy (%)
Basic Structure	43.8
JSON Structure	54.2
Checklist Structure (Equal)	60.4

Table 1: Comparison Between Varying Output Formats on DeepSeek V3.1

Specifically, the checklist method applies equal prioritization for better accuracy. As revealed in Table 2, we compare the equal checklist against a priority checklist under the same experimental setup and found that prioritization degrades performance. This is contrary to our expectations and may indicate that explicit hierarchies introduce unnecessary complexity, causing models to overfocus on specific categories and overlook holistic compliance.

Model	Priority	Equal
Claude 4 Sonnet	68.8	**77.1**
Gemini 2.5 Pro	67.5	**80.0**

Table 2: Impact of Priority System on Structured Prompting Performance

To better understand the performance characteristics across different instruction types, we analyze the breakdown of results for our equal checklist approach across the eight categories in the Inverse IFEval dataset. Figure 1 presents radar charts showing how different models handle various counterintuitive instruction types.

The results reveal that all models struggle significantly with "Question Correction", highlighting systematic challenges in this category across the board. While O3 Pro performs strongly across most categories, demonstrating high accuracies such as 87.5% in "Instructional Induction" and 100% in "Code without Comments", Claude 4 Sonnet excels in "Counterfactual Answering" (83.3%) and "Counter-Conventional Formatting" (80.5%), though its performance drops to 44.4% in "Question Correction". Gemini 2.5 Pro demonstrates high accuracy in "Deliberately Incorrect Answers" (90.3%), but struggles notably with "Code without Comments" (45.7%) and "Question Correction" (55.6%). Meanwhile, both DeepSeek v3.1 and Qwen 3-32B consistently face challenges, particularly in "Question Correction" (35.6% for both) and "Intentional Textual Flaws" (45.3% for DeepSeek v3.1 and 27.9% for Qwen 3-32B), underscoring common areas of difficulty among these models.

The eight categories are meticulously designed to probe nuanced aspects of instruction comprehen-

sion and execution, ranging from straightforward adherence to complex inferential tasks. Figure 2 presents stripe charts, where each model's performance is denoted by a unique marker, allowing for an immediate and intuitive comparison of their respective accuracy scores on each task category.

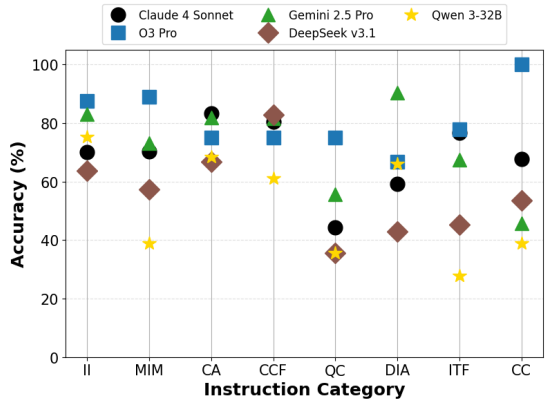

Figure 2: Comparison of Model Performance Across Eight Instruction Categories: II, MIM, CA, CCF, QC, DIA, ITF, CC

Finally, we benchmark our best variant, the equal checklist, against the baseline across all models and the full Inverse IFEval evaluation set. Figure 3 demonstrates substantial improvements, with statistically significant gains $p < 0.05$ and large effect sizes, confirming that structured prompting effectively enhances adaptability on counterintuitive tasks.

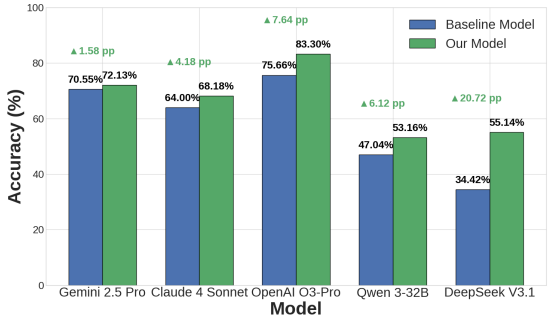

Figure 3: Comparison of Baseline vs. Our Structured Checklist Approach Performance. The chart shows improvements for all models: Gemini 2.5 Pro (+1.58pp), Claude 4 Sonnet (+4.18pp), O3 Pro (+7.64pp), Qwen 3-32B (+6.12pp) and DeepSeek V3.1 (+20.72pp).

5 Conclusion

This study showcases the power of structured output techniques in prompt engineering to boost LLMs' handling of counterintuitive instructions on the Inverse IFEval dataset. Our zero-shot framework—decomposing tasks into instruction parsing, checklists, structured responses, and self-checks—effectively counters cognitive inertia without any fine-tuning or training data. The equal checklist variant delivers a 10.06% average accuracy gain over baselines across models, with significant statistical improvements $p < 0.05$ and large effect sizes, underscoring zero-shot prompting's role in enhancing adaptability.

Our zero-shot approach advances AI robustness by providing a lightweight, verifiable method that outperforms standard prompting, ideal for safety-critical scenarios like ethical decisions or dynamic settings. Future work could extend this to multimodal domains or combine it with reinforcement learning for amplified flexibility.

6 Limitations

Despite these advancements, our work presents several limitations that warrant future consideration.

6.1 Resource and Scope Constraints

First, due to the substantial computational cost associated with proprietary models, particularly O3 Pro, we were unable to run the full benchmark on this specific model. Consequently, the evaluation for this model relies on a smaller sample size, while all other models were tested on the complete set of 500 samples.

6.2 Evaluation Methodology Limitations

Second, our reliance on the LLM-as-a-Judge paradigm introduces potential biases. While this methodology (using Claude-4.5-Sonnet) is recognized for its scalability and inter-rater consistency, the evaluation outcomes may inherently inherit the biases or stylistic preferences of the judge model itself. Furthermore, the use of binary scoring (correct/incorrect) overlooks instances of partial correctness or nuanced, but incomplete, responses, which limits the granularity of our error analysis.

6.3 Future Work

These constraints suggest clear avenues for future refinement. Potential directions include: (1) integrating a hybrid human-AI evaluation framework to validate and cross-reference the automated assessment, and (2) pursuing full benchmark testing across all models as resource constraints are alleviated.

References

Hai Dang, Lukas Mecke, Florian Lehmann, Sven Goller, and Daniel Buschek. 2022. How to prompt? opportunities and challenges of zero-and few-shot learning for human-ai interaction in creative applications of generative models. *arXiv preprint arXiv:2209.01390*.

Jessica Echterhoff, Yao Liu, Abeer Alessa, Julian McAuley, and Zexue He. 2024. Cognitive bias in decision-making with llms. *arXiv preprint arXiv:2403.00811*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Yinheng Li. 2023. A practical survey on zero-shot prompt design for in-context learning. *arXiv preprint arXiv:2309.13205*.

Jiaxu Lou and Yifan Sun. 2024. Anchoring bias in large language models: An experimental study. *arXiv preprint arXiv:2412.06593*.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. In *The 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (IJCNLP-AACL 2023)*.

Shentong Mo and Miao Xin. 2024. Tree of uncertain thoughts reasoning for large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12742–12746. IEEE.

Robert Nowak. 2025. Estimating the self-consistency of llms. *arXiv preprint arXiv:2509.19489*.

Leonardo Ranaldi, Giulia Pucci, Federico Ranaldi, Elena Sofia Ruzzetti, and Fabio Massimo Zanzotto. 2024. A tree-of-thoughts to broaden multi-step reasoning across languages. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1229–1241.

Philip Resnik. 2025. Large language models are biased because they are large language models. *arXiv preprint arXiv:2406.13138*.

Zhisheng Tang and Mayank Kejriwal. 2024. Humanlike cognitive patterns as emergent phenomena in large language models. *arXiv preprint arXiv:2412.15501*.

Amir Taubenfeld, Tom Sheffer, Eran Ofek, Amir Feder, Ariel Goldstein, Zorik Gekhman, and Gal Yona. 2025. Confidence improves self-consistency in llms. *arXiv preprint arXiv:2502.06233*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Qinyan Zhang, Xinping Lei, Ruijie Miao, Yu Fu, Haojie Fan, Le Chang, Jiafan Hou, Dingling Zhang, Zhongfei Hou, Ziqiang Yang, and 1 others. 2025. Inverse ifeval: Can llms unlearn stubborn training conventions to follow real instructions? *arXiv preprint arXiv:2509.04292*.

Xuan Zhang, Chao Du, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. 2024. Chain of preference optimization: Improving chain-of-thought reasoning in llms. *Advances in Neural Information Processing Systems*, 37:333–356.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Zhi Zhou, Tan Yuhao, Zenan Li, Yuan Yao, Lan-Zhe Guo, Xiaoxing Ma, and Yu-Feng Li. 2025. Bridging internal probability and self-consistency for effective and efficient llm reasoning. *arXiv preprint arXiv:2502.00511*.