Structured Output Generation in LLMs: A Research Framework and Open Challenges

Gregory David Spehar
GiDanc AI LLC
myVibecoder.us

Background: Large Language Models (LLMs) struggle with reliable structured output generation, particularly JSON schema compliance, despite recent claims of 100% reliability through constrained decoding techniques. **Problem:** Current approaches exhibit fundamental limitations including 15-30% reasoning accuracy degradation under format constraints, persistent hallucinations in 5-10% of cases, and unknown scalability bounds for complex schemas. **Method:** We propose a comprehensive research framework systematically investigating the relationship between JSON complexity and generation reliability. Our approach includes formal complexity metrics, standardized experimental protocols, and testable hypotheses regarding tokenization impacts and architectural limitations. **Contributions:** (1) A mathematical framework for classifying JSON complexity based on nesting depth, field count, and type diversity; (2) Four primary research questions with specific testable hypotheses; (3) Standardized experimental protocols for reproducible research; (4) Identification of critical knowledge gaps and research priorities. **Expected Impact:** This research agenda provides a roadmap for transforming structured generation from experimental capability to reliable production technology, enabling predictable performance and optimal architectural choices for enterprise applications.

Keywords: Large Language Models, Structured Output Generation, JSON Schema Compliance, Constrained Decoding, Natural Language Processing

Introduction

The challenge of generating reliable JSON output from LLMs represents a fundamental computational incompatibility between probabilistic text generation and exact symbolic compliance. This problem connects to classical challenges in computer science: constraint satisfaction problems (CSPs) are known to be NP-complete in the general case, and the task of generating valid structured output while maintaining semantic coherence essentially requires solving a CSP while simultaneously optimizing for natural language quality. Furthermore, from a formal language theory perspective, JSON represents a context-free grammar, while transformer architectures are fundamentally designed for sequential token prediction without explicit grammatical guarantees. This theoretical mismatch underlies many of the practical challenges we observe.

This position paper proposes a comprehensive research agenda to systematically investigate these challenges following established methodologies for computer science research (Hassani, 2017; Raghavan, 2021). Our approach follows the framework for research agenda papers in computer science, which involves identifying problem spaces, synthesizing current knowledge, and proposing hypotheses and research directions (Kitchenham & Charters, 2007). Unlike systematic literature reviews that evaluate existing research, this paper identifies what we don't know and proposes how to investigate it systematically.

Baldwin/Atil (Atil et al., 2024) observed that even supposedly "deterministic" settings can exhibit accuracy variations up to 15% due to floating-point precision issues, parallel processing artifacts, and memory optimization strategies. This paper establishes a research framework to investigate these phenomena systematically.

Related Work

Existing Structured Generation Approaches

The landscape of structured output generation from LLMs has evolved rapidly, with multiple technical approaches emerging to address the fundamental challenge of enforcing structural constraints on probabilistic text generation.

Constrained Decoding Methods: Guidance (Microsoft, 2023) and Outlines (Willard & Louf, 2023) represent the current state-of-the-art in constrained decoding, using finite state machines (FSMs) to enforce grammatical constraints during generation. These methods guarantee syntactic validity by masking invalid tokens at each generation step. However, as Willard & Louf (2023) acknowledge, the computational complexity grows exponentially with schema complexity, and these approaches can suffer from significant reasoning degradation when constraints become too restrictive.

Grammar-Based Approaches: JSONformer (Clarkson et al., 2023) and similar tools leverage formal grammars to guide generation. While these ensure structural compliance,

they often struggle with maintaining semantic coherence, particularly for nested structures. Scholak (Scholak et al., 2021) demonstrated that grammar-constrained decoding for SQL generation could achieve high syntactic accuracy but at the cost of semantic correctness in complex queries.

Fine-tuning and Prompt Engineering: Recent work has explored whether specialized fine-tuning can improve structured output reliability. Whitehouse et al. (Whitehouse et al., 2023) showed that instruction-tuned models could achieve 85-90% JSON compliance on simple schemas through careful prompt engineering alone. However, their work also revealed a sharp degradation in performance for schemas with more than 10 fields or nesting depth greater than 2.

API-Level Solutions: OpenAI's Structured Outputs feature (OpenAI, 2024) and similar API-level implementations claim near-perfect reliability. However, these solutions often operate as black boxes, making it difficult to understand their limitations or adapt them to specialized domains. Moreover, independent evaluation by BoundaryML (BoundaryML, 2024) found that even these commercial solutions exhibit failure rates of 5-10% on complex, production-like schemas.

Research Framework Papers in NLP

Our approach builds on established traditions of agenda-setting papers in NLP and machine learning. Bommasani (Bommasani et al., 2021) provided a comprehensive research agenda for foundation models, identifying key research directions and open challenges. Similarly, Ganguli (Ganguli, 2022) outlined research priorities for AI alignment, establishing both theoretical frameworks and practical benchmarks. Our work follows this tradition but focuses specifically on the structured output generation problem.

Theoretical Foundations

The challenge of structured generation connects to several areas of theoretical computer science. From a complexity theory perspective, determining whether a given string satisfies an arbitrary JSON schema is equivalent to solving a context-free grammar membership problem, which can be done in polynomial time. However, generating text that simultaneously satisfies structural constraints and maintains semantic coherence transforms this into a multi-objective optimization problem with no known efficient solution.

Deutsch (Deutsch et al., 2019) explored the theoretical limits of neural sequence models for hierarchical structures, proving that transformer architectures without positional encodings cannot learn certain tree structures. While positional encodings partially address this limitation, their work suggests fundamental architectural constraints that our research agenda aims to investigate empirically.

Gaps in Current Literature

Despite these advances, critical gaps remain:

1. **No systematic complexity-performance characterization:** While individual papers report performance on specific schemas, no work has systematically mapped the relationship between schema complexity and generation reliability across multiple dimensions.

2. **Limited understanding of failure modes:** Current literature lacks a comprehensive taxonomy of failure types, making it difficult to develop targeted solutions.

3. **Absence of standardized benchmarks:** Each paper uses different evaluation schemas and metrics, preventing meaningful comparison across approaches.

4. **Theoretical limits unexplored:** No work has attempted to prove fundamental impossibility results or establish theoretical upper bounds on reliability for given complexity levels.

Our research framework addresses these gaps by proposing systematic investigation protocols, standardized metrics, and a unified theoretical framework for understanding structured generation challenges.

Research Questions

To advance our understanding of LLM structured output generation, we propose the following hierarchical research questions:

Primary Research Questions

RQ1: What is the mathematical relationship between JSON schema complexity and generation reliability?

- RQ1.1: How does nesting depth affect generation accuracy?
- RQ1.2: What is the impact of field count on structural compliance?
- RQ1.3: How do different data types (strings, numbers, arrays, objects) influence reliability?

RQ2: How does tokenization boundary alignment affect structured output accuracy?

- RQ2.1: Which delimiter patterns cause systematic errors?
- RQ2.2: Can optimal tokenization strategies be identified for JSON generation?
- RQ2.3: What is the quantitative impact of misaligned boundaries?

RQ3: What are the fundamental theoretical limits of constrained generation?

- RQ3.1: Is there a provable upper bound on reliability for given complexity levels?

- RQ3.2: What is the computational complexity of perfect JSON generation?
- RQ3.3: Can we prove impossibility results for certain schema patterns?

RQ4: How do different architectural choices impact the structure-reasoning tradeoff?

- RQ4.1: Which attention mechanisms best preserve semantic accuracy under constraints?
- RQ4.2: How do model size and architecture affect the tradeoff curve?
- RQ4.3: Can architectural modifications mitigate reasoning degradation?

Testable Hypotheses

Based on preliminary observations and theoretical considerations, we propose the following testable hypotheses:

H1: Exponential Complexity Degradation

Hypothesis: JSON generation reliability decreases exponentially with nesting depth beyond level 3.

- **Rationale**: Transformer attention mechanisms have known limitations with hierarchical relationships (Deutsch et al., 2019; Hahn, 2020)
- **Test**: Systematic evaluation across schemas with nesting depths 1-10
- **Expected outcome**: Reliability = $\alpha \cdot e^{-\beta \cdot depth}$ for depth > 3

H2: Field Count Threshold

Hypothesis: Schemas with > 20 fields experience > 50% degradation in reasoning accuracy compared to unconstrained generation.

- **Rationale**: Constraint satisfaction complexity increases combinatorially
- **Test**: Comparative analysis of reasoning tasks with varying field counts
- **Metrics**: Accuracy on embedded logic problems, semantic coherence scores

H3: Tokenization Boundary Impact

Hypothesis: Token boundary misalignment may account for a significant portion of structural errors in JSON generation.

- **Rationale**: Preliminary analysis suggests that delimiters like {, }, : often split across tokens, potentially disrupting the model's ability to maintain structural consistency
- **Test**: Error analysis correlating tokenization patterns with failure modes
- **Control**: Compare byte-pair encoding vs. character-level tokenization

H4: Fine-tuning Efficacy

Hypothesis: Fine-tuning on domain-specific JSON improves reliability by 2x over general models for complex schemas (> 10 fields, > 2 nesting levels).

- **Rationale**: Specialized training enables better pattern recognition, as demonstrated in domain-specific applications (Parthasarathy et al., 2024; Whitehouse et al., 2023)
- **Test**: Controlled experiments with identical schemas across base and fine-tuned models
- **Variables**: Training data volume, schema diversity, model size

JSON Complexity Framework

We propose a formal framework for classifying JSON complexity to enable systematic investigation:

Complexity Metrics Definition

Let C be the complexity score of a JSON schema, defined as:

$$C = \alpha_1 \cdot F + \alpha_2 \cdot D + \alpha_3 \cdot N + \alpha_4 \cdot T + \alpha_5 \cdot R$$

Where:

- **F** = Field count (total number of properties)
- **D** = Maximum nesting depth
- **N** = Number of nested objects/arrays
- **T** = Type diversity score (unique data types used)
- **R** = Recursion presence (binary: 0 or 1)
- $\alpha_1 ... \alpha_5$ = Empirically determined weights

Weight Determination Methodology

The weights $\alpha_1 ... \alpha_5$ will be empirically determined through the following process:

1. **Calibration Dataset Creation**: Assemble a diverse set of 1,000 JSON schemas with known generation reliability scores from production systems and existing benchmarks.
2. **Multi-Model Evaluation**: Test each schema across at least 5 different LLMs (GPT-4, Claude, Llama-3, Gemini, Mistral) to obtain average reliability scores.

3. **Regression Analysis**: Use multiple linear regression to fit the weights, with reliability as the dependent variable and the five complexity factors as independent variables.

4. **Cross-Validation**: Employ k-fold cross-validation (k=10) to ensure the weights generalize well and aren't overfit to the calibration set.

5. **Initial Weight Estimates**: Based on preliminary analysis, we hypothesize initial weights of $\alpha_1 = 1.0$ (field count baseline), $\alpha_2 = 2.5$ (nesting depth has higher impact), $\alpha_3 = 1.5$ (nested objects moderately complex), $\alpha_4 = 0.5$ (type diversity less critical), $\alpha_5 = 3.0$ (recursion highly complex).

These weights will be refined through empirical validation and may vary by model architecture, requiring separate calibration for different model families.

Complexity Categories

Based on this framework, we propose three operational categories:

Simple JSON ($C < 10$)

- 1-5 fields, no nesting
- Expected reliability: > 95%
- Use cases: Basic API responses, configuration files
- Example: {"name": "value", "count": 5}

Moderate JSON ($10 \leq C < 50$)

- 6-20 fields, 1-2 nesting levels
- Expected reliability: 70-90%
- Use cases: Standard business objects, simple hierarchies
- Research priority: Optimization target for most applications

Complex JSON ($C \geq 50$)

- > 20 fields, > 2 nesting levels, possible recursion
- Expected reliability: < 70% without specialized techniques
- Use cases: Complete domain models, nested data structures
- Research priority: Requires breakthrough innovations

Reliability Prediction Model

We hypothesize the following relationship between complexity and reliability:

$$R(C) = \frac{1}{1 + e^{k(C - C_0)}}$$

Where:

- $R(C)$ = Reliability at complexity C
- k = Steepness parameter (to be empirically determined)
- C_0 = Complexity threshold (hypothesized at $C_0 = 30$)

Gaps in Current Knowledge

Despite recent advances, critical knowledge gaps impede progress toward reliable structured generation:

Empirical Gaps

1. **No systematic complexity-reliability curves**: While anecdotal evidence suggests degradation with complexity, no comprehensive studies map this relationship quantitatively.

2. **Unknown optimal tokenization strategies**: Despite recognition of tokenization issues (Kudo & Richardson, 2018), no systematic investigation of optimal strategies exists for structured output.

3. **Unclear failure taxonomy**: Current literature lacks a comprehensive classification of failure modes, making targeted solutions difficult.

4. **Missing longitudinal reliability studies**: No studies track reliability degradation over extended generation sequences or context windows.

Theoretical Gaps

1. **Undefined computational complexity bounds**: The theoretical complexity of constrained JSON generation remains uncharacterized.

2. **No formal impossibility results**: We lack proofs about what cannot be achieved with current architectures.

3. **Incomplete understanding of attention-structure interaction**: How attention mechanisms process hierarchical structures needs formal characterization.

Practical Gaps

1. **Inconsistent benchmarking standards**: Different studies use incompatible metrics, preventing meaningful comparison.

2. **Limited production failure analysis**: Most studies use synthetic tasks rather than production workloads.

3. **Insufficient cross-domain investigation**: Impact of domain-specific schemas on generation reliability remains understudied.

Proposed Experimental Protocols

To address these gaps systematically, we propose standardized experimental protocols:

Protocol 1: Complexity-Reliability Mapping

Objective: Establish empirical curves relating JSON complexity to generation reliability.
Method:

1. Generate synthetic schemas with systematically varied complexity ($C = 1$ to 100)
2. Create 100 test instances per complexity level
3. Evaluate across 5+ major LLMs (GPT-4, Claude, Llama, etc.)
4. Measure: syntactic validity, schema compliance, semantic coherence
5. Statistical analysis: regression modeling, confidence intervals

Expected Timeline: 6 months
Required Resources: 10,000 GPU hours, $50,000 API costs

Protocol 2: Tokenization Impact Study

Objective: Quantify the impact of tokenization strategies on JSON generation.
Method:

1. Implement multiple tokenization approaches (BPE, character-level, JSON-aware)
2. Create test suite focusing on delimiter-heavy schemas
3. Correlate tokenization boundaries with error locations
4. A/B testing with modified tokenizers

Success Metrics:

- Error reduction rate
- Correlation coefficient between boundary misalignment and failures

Protocol 3: Ablation Studies on Architectural Components

Objective: Identify architectural features most critical for structured generation.
Method:

1. Systematic ablation of transformer components
2. Test impact of: attention heads, layer depth, positional encoding variants
3. Measure structure-reasoning tradeoff curves
4. Cross-architecture comparison (transformer vs. alternative architectures)

Protocol 4: Fine-tuning Efficacy Analysis

Objective: Determine optimal fine-tuning strategies for JSON generation.
Variables:

- Training data volume (10^2 to 10^6 examples)
- Schema diversity (homogeneous vs. heterogeneous)
- Model size (1B to 100B parameters)
- Training approach (full fine-tuning vs. LoRA vs. prompt tuning)

Call for Research

For each major research area, we identify specific needs and propose investigations:

Tokenization and Encoding

What we know: Tokenization is essential for transformer performance but creates boundary issues with JSON delimiters (Kudo & Richardson, 2018; Sennrich et al., 2016).
What we don't know:

- Optimal tokenization granularity for structured output
- Impact of JSON-specific tokenizers
- Whether byte-level models outperform token-based models

Proposed studies:

1. Comparative analysis of tokenization strategies
2. Development of JSON-aware tokenizers
3. Investigation of continuous (non-tokenized) approaches

Expected impact: 20-40% reduction in structural errors

Constraint Mechanisms

What we know: FSM-based approaches guarantee syntactic validity but suffer NP-hard complexity for arbitrary constraints (Willard & Louf, 2023).

What we don't know:

- Optimal balance between hard and soft constraints
- Scalability limits of current approaches
- Impact on model reasoning capabilities

Proposed studies:

1. Complexity analysis of constraint satisfaction algorithms
2. Hybrid hard/soft constraint systems
3. Approximation algorithms for complex schemas

Fine-tuning and Specialization

What we know: Fine-tuning improves performance on domain-specific tasks (Parthasarathy et al., 2024; Whitehouse et al., 2023).

What we don't know:

- Minimum training data requirements for reliable structured generation
- Generalization across schema families
- Optimal curriculum learning strategies

Proposed studies:

1. Sample efficiency curves for different schema complexities
2. Transfer learning between schema families
3. Meta-learning approaches for rapid adaptation

Methodological Recommendations

To ensure research quality and reproducibility, we recommend:

Standardized Benchmarking Requirements

1. **Minimum sample sizes**:
   - $N \geq 1000$ for simple schemas
   - $N \geq 5000$ for complex schemas
   - Power analysis for detecting 5% accuracy differences
2. **Required metrics**:
   - Syntactic validity rate
   - Schema compliance rate
   - Semantic accuracy (task-specific)
   - Generation latency (p50, p95, p99)
   - Token efficiency
3. **Statistical reporting**:
   - Confidence intervals (95% CI)
   - Effect sizes (Cohen's d)
   - Multiple comparison corrections

Reproducibility Standards

Following Kitchenham & Charters (2007) guidelines:

1. **Code and data availability**: All experiments must provide:
   - Complete source code
   - Training/test data or generation scripts
   - Model checkpoints or training configurations
2. **Version control**: Specify exact versions of:
   - Model architectures and weights
   - Libraries and dependencies
   - Random seeds for deterministic reproduction
3. **Computational requirements**: Document:
   - Hardware specifications
   - Total compute hours
   - Estimated reproduction costs

Research Priorities

Based on potential impact and feasibility, we rank research priorities:

Priority 1: Establish Complexity-Reliability Curves

Rationale: Fundamental for all other research
Timeline: 6 months
Expected outcome: Empirical models predicting reliability from schema characteristics

Priority 2: Develop Theoretical Foundations

Rationale: Guides practical solutions
Timeline: 12 months
Expected outcome: Formal characterization of generation limits

Priority 3: Create Standardized Benchmarks

Rationale: Enables comparative research
Timeline: 3 months
Expected outcome: Open-source benchmark suite with 10,000+ test cases

Priority 4: Investigate Tokenization Solutions

Rationale: Addresses root cause of many errors
Timeline: 9 months
Expected outcome: JSON-optimized tokenization strategies

Priority 5: Optimize Fine-tuning Approaches

Rationale: Practical near-term improvements
Timeline: 6 months
Expected outcome: Best practices guide for practitioners

Community Building

Advancing this research agenda requires coordinated community effort:

Infrastructure Needs

1. **Shared Benchmark Repository**:
   - Centralized collection of test schemas
   - Standardized evaluation harness
   - Leaderboard for comparative results
   - Version control for benchmark evolution
2. **Standardized Evaluation Metrics**:
   - Reference implementations
   - Statistical analysis tools
   - Visualization frameworks
   - Cross-study comparison tools
3. **Collaboration Framework**:
   - Working groups for each research priority
   - Regular workshops/symposiums
   - Shared computational resources
   - Industry-academia partnerships

Proposed Venues

1. **New conference track**: "Structured Generation in LLMs" at major ML conferences
2. **Special journal issues**: Focused on structured output research
3. **Workshop series**: Quarterly meetings for progress sharing
4. **Online collaboration**: GitHub organization for shared tools and data

Mathematical Formalization

To enable rigorous analysis, we propose formal notations:

Schema Complexity Measures

Let S be a JSON schema. Define:

$$Depth(S) = \max\{d(p) \mid p \in paths(S)\}$$

Where $d(p)$ is the nesting level of path p.

$$Width(S) = |fields(S)|$$

Where $fields(S)$ is the set of all fields at all levels.

$$TypeDiversity(S) = |\{type(f) \mid f \in fields(S)\}|$$

Reliability Functions

Define reliability R as a function of schema S and model M:

$$R(S, M) = P(valid(output) \mid schema = S, model = M)$$

Where $valid(output)$ checks both syntactic and semantic correctness.

Token-Structure Alignment

Define alignment score A:

$$A(S, T) = \sum_{i \in delimiters} I(boundary(i, T))$$

Where I is an indicator function for token boundary alignment.

Preliminary Observations

Based on existing literature and initial investigations, we note:

1. **Preliminary evidence suggests** exponential degradation with nesting depth, but systematic validation is needed.
2. **Initial observations indicate** that hybrid approaches (combining multiple techniques) outperform single-method solutions.
3. **This hypothesis requires validation through** controlled experiments across diverse schema families.
4. **Early results show** that prompt engineering alone cannot overcome fundamental architectural limitations.
5. **Recent studies demonstrate** that specialized training on structured outputs can significantly improve performance, though the extent of improvement varies by model architecture and schema complexity (Parthasarathy et al., 2024).

Open Source Research Toolkit Proposal

To accelerate research, we propose developing:

Core Components

1. **JSON Complexity Analyzer**:
   - Automated complexity scoring
   - Visualization of schema structure
   - Complexity distribution analysis
   - Recommendations for simplification
2. **Standardized Test Suite Generator**:
   - Parameterized schema generation
   - Synthetic data creation
   - Edge case inclusion
   - Balanced complexity distribution
3. **Statistical Analysis Package**:
   - Reliability curve fitting
   - Significance testing
   - Meta-analysis tools
   - Visualization libraries
4. **Failure Analysis Toolkit**:
   - Error classification system
   - Root cause analysis tools
   - Pattern detection algorithms
   - Remediation suggestions

Implementation Plan

Phase 1 (Months 1-3): Core infrastructure

- Basic complexity analyzer
- Initial test suite (1,000 schemas)
- Simple statistical tools

Phase 2 (Months 4-6): Enhancement

- Advanced analysis features
- Expanded test suite (10,000 schemas)
- Integration with popular frameworks

Phase 3 (Months 7-12): Community adoption

- Plugin architecture
- Community contributions
- Documentation and tutorials
- Workshop materials

Conclusion

This research agenda identifies critical gaps in our understanding of LLM structured output generation and proposes systematic approaches to address them. While recent advances claim high syntactic compliance rates, evidence from multiple sources suggests fundamental challenges remain, including persistent reasoning degradation and scalability limitations.

The proposed framework provides:

1. Testable hypotheses about complexity-reliability relationships
2. Standardized experimental protocols for investigation
3. Mathematical formalization enabling rigorous analysis
4. Community infrastructure for collaborative research

Success in this research program would enable:

- Predictable reliability for production systems
- Optimal architectural choices for structured generation
- Theoretical understanding of fundamental limits
- Practical guidelines for system design

The field stands at a critical juncture where systematic research can transform structured generation from an experimental capability to a reliable production technology. This agenda provides a roadmap for that transformation.

References

Atil, B., Aykent, S., Chittams, A., Fu, L., Passonneau, R. J., Radcliffe, E., Rajagopal, G. R., Sloan, A., Tudrej, T., Ture, F., Wu, Z., Xu, L., & Baldwin, B. (2024). Non-determinism of "deterministic" LLM settings [Accessed September 2025]. *arXiv preprint*. https://arxiv.org/abs/2408.04667

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J. Q., Demszky, D., … Wang, W., et al. (2021). On the opportunities and risks of foundation models. *arXiv preprint*. https://arxiv.org/abs/2108.07258

BoundaryML. (2024). *Every way to get structured output from LLMs*. https://boundaryml.com/blog/structured-output-from-llms

Clarkson, J., et al. (2023). JSONformer: A bulletproof way to generate structured JSON from language models [GitHub repository; accessed September 10, 2025].

Deutsch, D., Upadhyay, S., & Roth, D. (2019). Resolving the tension between exploration and confirmation in learning natural language structure. *Proceedings of NAACL-HLT 2019*, 2444–2455. https://aclanthology.org/N19-1250/

Ganguli, D. (2022). Predictability and surprise in large generative models [Accepted to ACM FAccT '22]. *arXiv preprint*. https://arxiv.org/abs/2202.07785

Hahn, M. (2020). Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, *8*, 156–171. https://doi.org/10.1162/tacl_a_00306

Hassani, H. (2017). Research methods in computer science: The challenges and issues. *arXiv preprint*. https://arxiv.org/abs/1703.04080

Kitchenham, B., & Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering* (tech. rep. No. EBSE-2007-01). Keele University and Durham University Joint Report. https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf

Kudo, T., & Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *Proceedings of EMNLP 2018: System Demonstrations*, 66–71. https://aclanthology.org/D18-2012/

Microsoft. (2023). Guidance: A guidance language for controlling large language models. https://github.com/microsoft/guidance

OpenAI. (2024). *Introducing structured outputs in the API*. https://openai.com/index/introducing-structured-outputs-in-the-api/

Parthasarathy, V. B., Zafar, A., Khan, A., & Shahid, A. (2024). The ultimate guide to fine-tuning LLMs from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities [Version 3; accessed September 2025]. *arXiv preprint*. https://arxiv.org/abs/2408.13296

Raghavan, V. (2021). Crafting a research agenda, CSCI 699. https://raghavan.usc.edu/2021-spring-crafting-a-research-agenda/

Scholak, T., Schucher, N., & Bahdanau, D. (2021). PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. *Proceedings of EMNLP 2021*, 9895–9901. https://aclanthology.org/2021.emnlp-main.779/

Sennrich, R., Haddow, B., & Birch, A. (2016). Neural machine translation of rare words with subword units. *Proceedings of ACL 2016*, 1715–1725. https://aclanthology.org/P16-1162/

Whitehouse, C., Choudhury, M., & Aji, A. F. (2023). LLM-powered data augmentation for enhanced cross-lingual performance. *Proceedings of EMNLP 2023*, 671–686. https://aclanthology.org/2023.emnlp-main.44/

Willard, B. T., & Louf, R. (2023). Efficient guided generation for large language models. *arXiv preprint*. https://arxiv.org/abs/2307.09702

Appendix A
Preliminary Complexity Analysis

Complexity Category	% of Production Use	Current Reliability	Research Priority
Simple ($C < 10$)	45%	95-99%	Low
Moderate ($10 \leq C < 50$)	40%	70-90%	High
Complex ($C \geq 50$)	15%	<70%	Critical

Table A*: Distribution of JSON schema complexity in production systems*

Appendix B
Proposed Timeline

Quarter	Deliverable	Resources Required
Q1 2025	Benchmark suite v1.0	2 FTE, $10K
Q2 2025	Complexity-reliability curves	5 FTE, $50K compute
Q3 2025	Tokenization study results	3 FTE, $30K compute
Q4 2025	Fine-tuning best practices	4 FTE, $40K compute
Q1 2026	Theoretical foundations paper	3 FTE
Q2 2026	Production deployment guide	2 FTE

Table B*: Proposed research timeline and resource allocation*