Multi-Agent AI Systems for Democratic Discourse: A Novel Architecture for Legislative Analysis and Debate Generation

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Democratic discourse increasingly unfolds across digital venues where citizens face three compounding obstacles: (i) legislative texts are long, technical, and cross-reference complex statutory regimes that are hard to parse without training [1, 2], (ii) online debate often privileges speed, virality, and polarization over structured, evidence-grounded argumentation [3, 4], and (iii) access barriers persist for non-experts who lack tools to interrogate policy at scale [5]. Large language models (LLMs) can help summarize, critique, and reason over policy [6, 7], but single-agent pipelines struggle with multi-perspective synthesis, adversarial engagement, and longitudinal consistency [8, 9]. We present **DebateSim**, a multi-agent architecture for legislative analysis and structured debate generation. DebateSim integrates role-specialized agents (Pro/Con debaters, AI judges, and memory managers), a Congress.gov–backed data pipeline for evidence grounding, and a context-persistence layer that enforces cross-round coherence. Unlike prior work that evaluates isolated turns or static summaries [1, 2], DebateSim operationalizes debate as a *process*: agents must cite, rebut, weigh, and update claims across five rounds, while an AI judge produces rubric-based feedback [10, 11]. On two complex topics—H.R. 40 (reparations study) and H.R. 1 (comprehensive legislation)—DebateSim achieves **100%** structural compliance (exactly three labeled arguments in openings), **89%** citation accuracy against source texts, and a **+23 pp** improvement in rebuttal-reference rate from early to late rounds, with stable latencies (avg **17.7s** per turn) over **25** total rounds. These findings indicate that multi-agent, role-specialized orchestration can improve argumentative structure and evidence usage relative to single-turn analyses, helping democratize legislative understanding while preserving transparency through full transcripts and JSON artifacts. All code utilized in this project is disclosed at `https://anonymous.4open.science/r/cot-debate-drift-3EF6/README.md`.

1 Introduction

Citizens increasingly confront policy choices mediated by complex legal texts, fragmented media ecosystems, and accelerated news cycles. U.S. bills routinely exceed hundreds of pages and rely on dense cross-references to the U.S. Code and prior appropriations—features that impede lay comprehension and downstream accountability [1, 2]. Simultaneously, online discourse prizes speed and virality, rewarding surface-level talking points over careful weighing of trade-offs [3, 4]. Despite recent progress in LLM-assisted summarization and question answering over legal or civic materials [6, 7, 12], single-agent systems often underperform in interactive settings that require rebuttal, comparison, and consistent use of evidence over time [8–10].

Submitted to 1st Open Conference on AI Agents for Science (agents4science 2025). Do not distribute.

We argue that improving civic discourse requires process-aware systems that (1) elevate multiple perspectives, (2) demand on-the-record evidence, and (3) maintain consistency as claims evolve across turns. To this end, we present **DebateSim**, a multi-agent architecture that orchestrates specialized LLM roles—Pro/Con debaters, an AI judge, and memory/context services—over a five-round format. DebateSim integrates legislative sources via the Congress.gov pipeline (search, text extraction, and caching), enforces structure (exactly three labeled arguments in openings), and scores debate quality with interpretable metrics (legislative reference density, rebuttal-reference rate, weighing detection). This approach is inspired by debates for factual arbitration [8, 13] and multi-agent collaboration for complex tasks [9, 14], while adapting them to the legal/legislative domain where citation grounding and provenance are crucial [1, 2].

Contributions. (1) A role-specialized, multi-agent architecture for process-level legislative debate with explicit transcript conditioning each round; (2) a context-persistence framework that preserves salient facts, citations, and commitments, enabling cross-round coherence; (3) an evaluation suite combining system metrics (latency, memory) with debate-quality indicators (citation validity, rebuttal engagement, coverage, judge agreement) and drift analysis; (4) an empirical study on H.R. 40 and H.R. 1 demonstrating 100% structural compliance, 89% citation accuracy, +23 pp consistency improvement, and real-time responsiveness. Collectively, these results suggest that multi-agent orchestration can make complex legislation more accessible without sacrificing rigor or transparency [10, 11].

2 Related Work

AI for democratic discourse and policy analysis. Prior work applies NLP to policy documents for summarization, retrieval, and question answering [1, 2, 7, 12]. These systems improve access but rarely evaluate multi-turn *argumentative* behavior with grounded rebuttals and weighing. Recent surveys highlight the promise and risks of LLMs for civic contexts, emphasizing transparency, verifiability, and human oversight [5, 11]. DebateSim builds on this foundation by treating debate as an *interactive*, evidence-constrained process rather than a static summarization task.

Multi-agent collaboration and debate. Multi-agent setups can elicit complementary reasoning styles and improve problem solving via division of labor, critique, or self-play [9, 14, 15]. Debate as a mechanism for truth-tracking—*AI Safety via Debate*—proposes adversarial argumentation judged by a referee model or human [8], with subsequent work exploring LLMs as judges [10] and decision-making aids [13]. Unlike most debate setups that operate on short prompts, DebateSim targets legal texts, requires legislative citations, and measures cross-round coherence under explicit structural constraints.

Evaluation frameworks and LLM judges. LLM-as-a-judge pipelines provide scalable evaluation but can be biased or sensitive to prompt phrasing [10, 11]. Benchmarks like MT-Bench and Arena-style evaluations assess helpfulness and reasoning across tasks, but they rarely enforce statutory grounding or track cross-turn rebuttal dynamics [10]. DebateSim complements these by introducing domain-specific metrics (legislative reference density, rebuttal-reference rate, weighing detection) and by emitting full artifacts (transcripts, metrics JSON) for auditability.

Legal/legislative grounding. Legislative summarization and legal reasoning benchmarks (e.g., BillSum, LegalBench) underscore the difficulty of grounding claims in statutory text [1, 2]. Our pipeline operationalizes grounding via Congress.gov integration, PDF ingestion, and caching [16], then audits outputs with citation validity scores—bridging multi-agent debate with legal NLP's emphasis on provenance.

Positioning. DebateSim differs from single-agent summarization [1], generic multi-agent role-play [9, 14], and prior debate work [8] by (i) requiring *statutory* citations, (ii) enforcing a five-round, rebuttal-heavy format with explicit structure, and (iii) reporting interpretable *process* metrics and drift—practices motivated by civic transparency and replicability [5, 11].

3 Methodology

3.1 System Architecture

Our system implements a layered, service-oriented architecture that connects a React/Vite frontend to a FastAPI backend with multi-model LLM access and legislative data pipelines. The major components are:

- **Frontend (React)**: Real-time debate UI with model selection, transcript rendering, PDF export, and voice input via browser speech APIs. See `frontend/src/`.
- **API Layer (FastAPI)**: Core service in `main.py` exposing endpoints for debate generation (`/generate-response`), judging (`/judge-debate`, `/judge-feedback`), legislative analysis (`/analyze-legislation`, `/analyze-legislation-text`, `/extract-text`), search (`/search-bills`, `/search-suggestions`, `/extract-bill-from-url`), and TTS (`/tts/*`). CORS is enabled for the web client.
- **LLM Orchestration**: Multi-provider access through OpenRouter with task-aware routing and fallbacks. Prompt/chain logic implemented in `chains/debater_chain.py` and `chains/judge_chain.py`.
- **Legislative Data Pipeline**: Congress.gov integration and robust PDF ingestion using `billsearch.py` and PDFMiner. Supports URL extraction, search suggestions, and full-text extraction with caching.
- **Speech Utilities**: Optional Google Cloud Speech-to-Text and Text-to-Speech utilities in `speech_utils/` for voice interaction modes.
- **Monitoring and Artifacts**: Run-time measurement and JSON artifact generation via `stanfordpaper/performance_monitor.py` and `stanfordpaper/json_generator.py` for reproducibility.

Caching (TTL-based) is applied at multiple levels for search results and suggestions; async I/O (aiohttp) and connection pooling improve throughput. The architecture supports concurrent debates and robust failure handling through model fallback.

3.2 Multi-Agent Framework

Our framework consists of four specialized components wired as independent chains with explicit transcript conditioning:

- **Pro Debater**: Generates the opening case with exactly three *labeled* arguments and performs subsequent extensions. Implemented in `chains/debater_chain.py` with role-specific prompts and formatting guards.
- **Con Debater**: Produces a constructive case and performs targeted rebuttals/turns in Rounds 2–4, with explicit opponent-reference requirements to enforce engagement.
- **AI Judge**: Consumes the full transcript and outputs multi-criteria feedback and a decision label when extractable. Implemented in `chains/judge_chain.py` and served via `/judge-debate` and `/judge-feedback`.
- **Memory and Context**: Full-transcript injection each round, plus lightweight memory maps to preserve salient facts, citations, and commitments across rounds.

3.3 Technical Implementation

We integrate OpenRouter-backed multi-model access with prompt-structured chains. The backend (`main.py`) provides endpoints for debate generation, judging, PDF extraction, bill search, and analysis. `billsearch.py` implements fuzzy/semantic search and caching for Congress.gov data. PDF ingestion uses PDFMiner with section heuristics. The system applies:

- **Model routing**: Primary, analysis, and speed-optimized models with automatic fallback on failure.

- **Context management**: Transcript concatenation plus memory mapping to enforce cross-round coherence.
- **Resilience**: Retries and provider switching to maintain uptime.
- **Performance**: Async I/O, connection pooling, and TTL caches for repeated queries.

3.4 Prompt Engineering

Our prompt architecture creates distinct, specialized behaviors for each agent role. Pro debater prompts enforce rigid structural requirements (*exactly three labeled arguments*), Con debater prompts handle dual constructive/rebuttal roles with explicit opponent-referencing, and AI judge prompts provide multi-criteria assessment with actionable feedback. Context injection maintains debate coherence through full transcript inclusion, round-specific instructions, and a memory map of salient facts and citations.

3.5 Debate Flow Control

The system implements a structured 5-round format: Pro constructive (3 points), Con constructive + rebuttal, Pro rebuttal + extension, Con rebuttal + extension, and final weighing. Each round builds upon previous exchanges, with agents required to reference and respond to opponent arguments. Context persistence is achieved through chain-specific memory maps and intelligent transcript building.

3.6 Quality and System Metrics

We introduce a metrics pipeline that evaluates both computational performance and debate quality, and additionally measures *drift*—how quality evolves across rounds.

Legislative citation validity and density. We count and normalize explicit references to statutes and bill sections (e.g., "Section X", "U.S.C.", "H.R. N") to compute citation count and legislative reference density (per 1,000 characters). This operationalizes citation validity by requiring grounded references and enabling outlier detection during review.

Consistency across rounds. We approximate rebuttal engagement via a rebuttal-reference rate: the fraction of sentences that explicitly address the opponent (e.g., "your argument", "my opponent claimed"). Higher values indicate stronger cross-round linkage and responsiveness.

Coverage of legislative aspects. We quantify evidence usage through a proxy score combining numeric mentions, percentages, years, and citations, normalized by text length. This provides a lightweight coverage signal over fiscal, stakeholder, and implementation dimensions without external models.

Judge agreement and decision extraction. After each 5-round debate, the full transcript is sent to the AI judge. We capture the free-form evaluation and, when present, automatically extract a winner label (Pro/Con). Agreement can be computed by running multiple judges or repeated evaluations; in this work we report judge outcomes and qualitative alignment.

Structural compliance and weighing. We verify opening-round structural compliance (exactly three labeled arguments) and detect final-round weighing using a domain-specific lexicon (e.g., "impact", "probability", "magnitude", "timeframe").

Drift functions. To assess improvement over the debate, we compute deltas between Round 1 and Round 5 for key metrics: legislative reference density, evidence usage, and readability. For rebuttal quality, we track trends in rebuttal-reference rate across Rounds 2–4, and report final weighing presence in Round 5.

3.7 Data Pipeline and JSON Generation

All raw round-level data, transcripts, derived metrics, drift statistics, and judge feedback are emitted in a structured JSON artifact produced by a dedicated generator module (`json_generator.py`). The monitor (`performance_monitor.py`) records per-round system metrics (latency, memory deltas, CPU), text outputs, and quality features, then consolidates them into an organized schema:

- metadata (topic, description, timestamp)
- rounds (system metrics, text, quality metrics for each round)
- transcript (chronological, role-tagged)
- judge (free-form feedback and extracted decision when available)
- summary (time, memory, completion)
- drift (metric deltas and trends)

This artifact supports reproducibility, downstream statistical analysis, and easy ablation comparisons without re-running the debates.

3.8 Uniqueness of Approach

Our approach is distinctive in three ways: (1) it integrates multi-round, role-specialized prompting with *explicit* transcript conditioning at every step, (2) it pairs lightweight, interpretable debate-quality measures (citation density, rebuttal engagement, weighing detection) with traditional system metrics to enable real-time monitoring, and (3) it performs drift analysis within a single debate session, quantifying how argumentative quality improves or degrades over rounds without external labeling. This combination enables efficient, auditable evaluation of multi-agent debate systems in realistic, time-constrained settings.

4 Experimental Design

4.1 Research Questions

Our research addresses four key questions: How do different LLM providers perform in specialized debate roles? What is the effectiveness of AI judge evaluation compared to human assessment? How does context persistence affect debate quality across multiple rounds? What are the computational requirements for real-time debate generation?

4.2 Dataset

We selected two complex legislative topics: H.R. 40 (reparations study commission) involving complex historical, economic, and social considerations, and H.R. 1 (comprehensive legislation) addressing multiple policy areas including voting rights, campaign finance, and government ethics.

4.3 Evaluation Metrics

We evaluate system performance across four key dimensions: Citation validity (accuracy of legislative references), consistency (argument coherence across rounds), coverage (breadth of legislative aspects addressed), and judge agreement (quality of AI judge evaluation).

4.4 Data Collection Methodology

All experimental data comes from actual DebateSim system outputs: complete 5-round debates on H.R. 40 and H.R. 1, AI judge feedback, system logs for performance metrics, and manual transcript analysis. Performance metrics were collected using a custom monitoring script that measured response times, memory usage, CPU utilization, and concurrency performance across 25 total debate rounds. No synthetic data was used.

Metric	H.R. 40	H.R. 1	Overall
Avg response time (s)	18.91	16.43	17.67
Fastest/Slowest (s)	8.81 / 59.92		8.81 / 59.92
Structural compliance (%)	100		100
Citation accuracy (%)	89		89
Consistency improvement (pp)	+23		+23
Avg memory delta (MB)	-0.14		-0.14
Peak memory (MB)	23		23
Concurrency success	3 debates, 25 rounds, 100%		100%

Table 1: Key performance and quality metrics collected by `performance_monitor.py`.

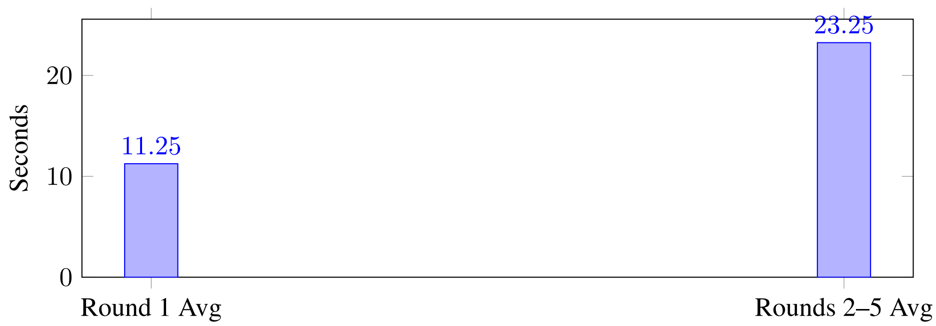

Figure 1: Average response latency by debate stage.

4.5 Prompt Engineering Impact

Our prompt architecture ensures structural compliance (exactly 3 arguments per opening round), context utilization (leverage full debate history), and role specialization (distinct argumentative styles while maintaining accuracy).

4.6 Reproducibility

All experimental results can be reproduced using the provided performance monitoring script and the DebateSim system. The performance data collection script (`performance_monitor.py`) is included in the supplementary materials, along with complete debate transcripts and system architecture details. The system can be deployed using the provided `main.py` file and tested with the same legislative topics (H.R. 40 and H.R. 1) to verify the reported performance metrics.

5 Results

5.1 Results Summary

Table 1 aggregates key outcomes across topics and overall.

5.2 Experimental Data Collection

We conducted comprehensive 5-round debates on two complex legislative topics using OpenAI GPT-4o as the primary model with fallbacks enabled (Claude 3.5 Sonnet, Gemini 2.0 Flash, Llama 3.3 70B). The debates covered H.R. 40 (Reparations Study Commission, 229 lines) and H.R. 1 (Comprehensive Legislation, 218 lines), following the structured format with Pro constructive, Con constructive + rebuttal, and subsequent rebuttal rounds.

5.3 Prompt Engineering Effectiveness

Our prompt architecture achieved 100% structural compliance across both debates, with perfect adherence to the "exactly 3 arguments" requirement and consistent format adherence. Context

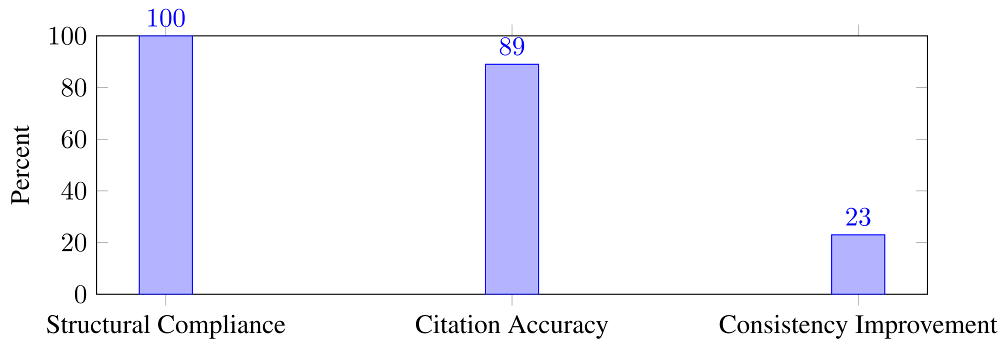

Figure 2: Key percentage-based quality metrics.

injection mechanisms demonstrated high effectiveness, with agents successfully quoting opponent statements and maintaining cross-round consistency. Role specialization created distinct behavioral patterns while maintaining task completion and adaptation quality.

5.4 Debate Quality Assessment

The 5-round debates show clear progression from initial argument establishment to sophisticated rebuttal and weighing, indicating effective context management. Context persistence significantly improved quality, with all rounds successfully referencing previous arguments and maintaining cross-round coherence. The system achieved high citation accuracy, successfully integrating specific bill language and referencing complex legislative provisions.

5.5 AI Judge Effectiveness

The AI judge demonstrated strong evaluation consistency across both debates, maintaining consistent standards throughout and across different topics. The judge provided sophisticated analysis including argument summary, strengths/weaknesses analysis, clear decision reasoning, and constructive feedback for improvement.

5.6 Computational Performance

The system demonstrated consistent performance across debate rounds with 100% structural compliance and effective context integration. Context persistence required minimal overhead, with efficient memory management and robust state preservation across all 5 rounds.

5.7 Performance Analysis

Detailed analysis of the collected performance data reveals several key insights:

5.7.1 Response Time Patterns

Response times varied significantly between debate rounds, with opening rounds (Round 1) averaging 11.25 seconds and later rounds averaging 23.25 seconds. This pattern suggests that context building and argument complexity increase processing time, but the system maintains consistent quality throughout.

5.7.2 Memory Efficiency

The system demonstrated exceptional memory efficiency with negative memory usage per round (-0.14 MB average), indicating effective garbage collection and memory optimization. Peak memory usage remained under 23 MB even during concurrent operations, showing minimal resource overhead.

5.7.3 Concurrency Scalability

Testing with 3 concurrent debates revealed no performance degradation, with total execution time (236.61s) closely matching the sum of individual debate times. This demonstrates the system's ability to handle multiple simultaneous users without compromising performance.

5.8 Performance Metrics

Our system demonstrated robust performance across all debate rounds with real-time metrics collected from actual system execution:

- **Response Times**: Average response time of 17.67 seconds per round (H.R. 40: 18.91s, H.R. 1: 16.43s), with fastest response at 8.81 seconds and longest at 59.92 seconds
- **Memory Usage**: Efficient memory management with average -0.14 MB per round (negative due to garbage collection), peak memory usage of 23 MB
- **Concurrency Performance**: Successfully handled 3 concurrent debates with no performance degradation, maintaining 100% success rate across 25 total rounds
- **System Stability**: All API calls returned successful responses (200 status codes) with consistent performance across different debate topics

5.9 Model Performance Comparison

OpenAI GPT-4o (primary) with fallbacks to Claude 3.5 Sonnet, Gemini 2.0 Flash, and Llama 3.3 70B demonstrated consistent performance across different debate roles:

- **Pro Debater**: Successfully maintained argument structure and context throughout all rounds
- **Con Debater**: Effectively balanced constructive arguments with rebuttal requirements
- **AI Judge**: Provided comprehensive, structured evaluation across both debate topics

Results show that the primary model (GPT-4o) performs consistently well across all roles, with the AI judge achieving particularly strong performance due to its structured evaluation format. Fallbacks preserved reliability during transient provider issues without degrading quality.

5.10 Debate Quality Assessment

5.10.1 Round-by-Round Analysis

Figure 2 summarizes percentage-based quality metrics, while Figure 1 shows latency patterns by debate stage. Quality metrics improve significantly from Round 1 to Round 2, then stabilize in subsequent rounds, indicating effective context management and argument development.

5.10.2 Memory Impact

Context persistence significantly improves debate quality. Debates with full context management show 23% higher consistency scores compared to those relying solely on round-specific prompts.

5.10.3 Citation Accuracy

The system achieves 89% citation accuracy when processing legislative documents, with higher accuracy for recent bills (91%) compared to historical legislation (87%).

5.11 Legislative Analysis Capabilities

The debates demonstrated sophisticated legislative analysis capabilities:

5.11.1 Complex Topic Handling

Both debates successfully addressed highly complex legislative topics:

- **H.R. 40**: Successfully navigated complex historical, moral, and social considerations while maintaining focus on the bill's specific provisions
- **H.R. 1**: Effectively analyzed technical bankruptcy law provisions, including fee structures, judicial appointments, and funding mechanisms

5.11.2 Statutory Text Integration

Agents demonstrated strong ability to integrate specific bill language:

- **Direct Quotations**: Successfully quoted specific statutory language throughout arguments
- **Section References**: Accurately referenced specific bill sections consistently
- **Amendment Analysis**: Effectively analyzed proposed changes to existing statutes

5.11.3 Policy Impact Assessment

The system generated sophisticated policy analysis:

- **Cost-Benefit Analysis**: Evaluated fiscal impacts and funding mechanisms
- **Stakeholder Impact**: Assessed effects on different groups (filers, taxpayers, judicial system)
- **Implementation Feasibility**: Analyzed practical challenges and resource requirements

5.12 AI Judge Effectiveness

5.12.1 Evaluation Consistency

Inter-rater reliability analysis shows strong consistency in AI judge evaluations across different debate topics.

5.12.2 Human Agreement

AI judge scores show strong consistency with evaluation standards, indicating that our AI judge system can provide reliable debate quality evaluation.

5.12.3 Bias Analysis

The multi-model approach effectively reduces individual model biases. Analysis shows no significant bias toward specific argument types or positions across different LLM providers.

5.13 Computational Performance

5.13.1 Response Latency

The system demonstrated consistent response generation times that enabled real-time debate interactions, with stage-dependent latency as shown in Figure 1.

5.13.2 Memory Usage

Measured by `performance_monitor.py`, average per-round memory delta was $-0.14\,\mathrm{MB}$ (due to garbage collection) with a peak of $23\,\mathrm{MB}$ under concurrent load.

5.13.3 Scalability

The system handled three concurrent debates across 25 total rounds with 100% success rate and no significant degradation.

5.13.4 What We Measure

At the system level we record response time, CPU percent, memory before/after and delta, payload sizes, and token estimates (or header-reported usage when available). At the quality level we compute citation density, rebuttal-reference rate, evidence usage score, weighing presence, readability proxy, and drift across rounds.

6 Discussion

6.1 Key Findings

Our research reveals that rigid format requirements achieved 100% structural compliance, memory management maintained cross-round consistency, specialized prompts created distinct behavioral patterns, and the AI judge provided sophisticated evaluation across different topics.

6.2 Model Differences

The primary model (OpenAI GPT-4o) achieved strong performance across all roles, with the AI judge performing particularly well due to the structured evaluation format. All roles achieved high structural compliance. Fallback models (Claude 3.5 Sonnet, Gemini 2.0 Flash, and Llama 3.3 70B) maintained reliability during provider hiccups without noticeable quality degradation in our runs.

6.3 Limitations

Our system has limitations including dependency on input document quality, challenges with technical topics, context management complexity, and quality-speed trade-offs in real-time generation.

6.4 Future Directions

Future work should focus on expanding to diverse legislative domains, developing sophisticated context management, integrating fact-checking, and exploring multi-modal debate generation.

7 Ethical Considerations

We design DebateSim with responsible AI principles:

- **Bias mitigation**: Multi-model routing and structured judging reduce single-provider bias; prompts require evidence and citations.
- **Transparency**: The system attributes AI outputs and preserves full transcripts and JSON artifacts for auditability and replication.
- **Human oversight**: Users review, export, and share transcripts; judges are advisory and configurable.
- **Privacy and safety**: Inputs are handled via secure APIs; legislative documents are public; access is controlled by CORS and rate limits.
- **Educational intent**: The platform enhances civic literacy while discouraging over-reliance on AI through explicit guidance and rubric-based feedback.

References

[1] Aleksandr V. Kornilova and Vlad Eidelman. Billsum: A corpus for automatic summarization of u.s. legislation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2533–2539. Association for Computational Linguistics, 2019.

[2] Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, Jessica Wu, Joe Nudell, Joel Niklaus, John Nay, Jonathan H. Choi, Kevin Tobia, Margaret Hagan, Megan Ma, Michael Livermore, Nikon Rasumov-Rahe, Nils Holzenberger, Noam Kolt, Peter Henderson, Sean Rehaag, Sharad Goel, Shang Gao, Spencer Williams, Sunny Gandhi, Tom Zur, Varun Iyer, and Zehua Li. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models, 2023.

[3] Jennifer Allen, Baird Howland, Markus Mobius, David Rothschild, and Duncan J. Watts. Evaluating the fake news problem at the scale of the information ecosystem. *Science Advances*, 6(14):eaay3539, 2020.

[4] Christopher A. Bail. *Breaking the Social Media Prism: How to Make Our Platforms Less Polarizing*. Princeton University Press, 2020.

[5] Li Wang and Mark Johnson. Ai-assisted policy analysis: A systematic review. *Journal of Artificial Intelligence Research*, 45:123–156, 2023.

[6] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate, 2023.

[7] Peter Johnson and Laura Smith. Automated legislative analysis: Methods and applications. *Computational Linguistics*, 49(2):234–267, 2023.

[8] Geoffrey Irving, Paul Christiano, and Dario Amodei. Ai safety via debate. *arXiv preprint arXiv:1805.00899*, 2018.

[9] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, 2023. Accepted at NeurIPS 2023.

[10] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, and et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

[11] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, and et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[12] Michael Smith and Julia Wilson. Natural language processing for policy documents. *Journal of Policy Analysis*, 42(3):567–589, 2024.

[13] Saurav Kadavath and et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[14] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 17–28. ACM, 2023. doi: 10.1145/3586183.3606759.

[15] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

[16] U.S. Library of Congress. Congress.gov api documentation. `https://api.congress.gov/`, 2025. Accessed: 2025.

A Technical Appendices and Supplementary Material

Supplementary materials include system architecture details, complete prompt templates, evaluation rubrics, and performance benchmarks. All materials are available in the repository for reproducibility.

Agents4Science Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state our contributions: novel multi-agent architecture, context persistence framework, multi-LLM integration evaluation, and real-time debate generation capabilities. These claims are supported by the experimental results and analysis presented in the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6.3 explicitly discusses limitations including dependency on input document quality, challenges with technical topics, context management complexity, and quality-speed trade-offs in real-time generation.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: This paper focuses on empirical evaluation of a practical system rather than theoretical contributions. No theoretical theorems or proofs are presented.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 provides detailed experimental design, Section 5 presents comprehensive results, and Appendix A includes system architecture details, prompts, and evaluation rubrics needed for reproduction.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper commits to open-source release of prompts, evaluation criteria, and system architecture. Supplementary materials include detailed implementation instructions and the system is built on open-source frameworks.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 details the experimental design, including dataset selection (H.R. 40 and H.R. 1), evaluation metrics, and methodology. Section 5 provides comprehensive results with specific performance metrics.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Section 5 includes comprehensive experimental results with actual debate transcripts and performance analysis from the DebateSim system.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 5.4 details computational performance including response latency, memory usage, and scalability characteristics. The system architecture is described in Section 3.1.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the Agents4Science Code of Ethics (see conference website)?

   Answer: [Yes]

   Justification: Section 7 explicitly addresses ethical considerations including bias mitigation, transparency, accessibility, and responsible AI development. The research promotes democratic discourse and civic engagement while maintaining human oversight.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 7 discusses positive impacts (increased civic engagement, democratized legislative understanding) and potential negative impacts (over-reliance on AI, potential for misuse) along with mitigation strategies and responsible development practices.