Towards a New Architecture for Structured Debate Generation

Kyle Hu
Stanford University
kylehu@stanford.edu

Bhavya Shah
Stanford University
bhavya@stanford.edu

Abstract

While AI models have achieved impressive results in a variety of fields, they lag behind human performance in competitive debate. Large state-of-the-art LLMs like the latest Gemini and OpenAI GPT models tend to have issues with generating full, well-structured debate speeches, and their large size and inaccessible weights makes them poor candidates for fine-tuning or running locally. Our goal was to see how we might close the gap between LLM and human performance in generating competitive debate speeches through sophisticated prompting, multi-model chaining and similar architecture design, and grounding in popular debate schema.

We found that our complicated multi-model preprocessing pipeline actually hindered LLM performance, while more detailed prompts improved performance on large models like OpenAI's GPT-4o and Google's Gemini 2.5 Pro but reduced it it on smaller models like GPT-4o-mini and Gemini 2.5 Flash. This suggests that model-specific prompt engineering and iterative self-criticism cycles, and perhaps fine-tuning if that option is available, may be a more productive route for improving LLM performance on debate generation than breaking down the debate generation process into subtasks handled by complex inference-time pipelines.

1 Introduction

In the past few years, LLMs have reached human-like performance in domains as diverse as competitive coding and math. However, one domain on which they continue to lag behind is competitive debate. While large state-of-the-art LLMs like the latest Gemini and OpenAI GPT models can generate emotionally charged arguments to varying degrees of success, they tend to have issues generating full, well-structured debate speeches. In addition, the size of these models, and the fact that they are closed-source and closed-weight, makes them unwieldy for independent researchers and hobbyists who want to run them locally or to fine-tune them for their own purposes. Our goal was to see how we might close the gap between LLM and human performance in generating competitive debate speeches through sophisticated prompting and multi-model architecture alone, without requiring expensive fine-tuning.

2 Related Work

In 2021, IBM released a paper in Nature called "An autonomous debating system", the culmination of 10 years research on what it called "Project Debater" (Slonim et al., 2021). This was an attempt to create an AI system that could debate live with experienced human debaters in the competitive debate framework. Prior to the release of the paper, IBM had demonstrated Project Debater to some success at a conference in 2018, where the AI debater lost to two humans due to weak delivery- its attempts at incorporating humor were judged to be poor- but scored higher than them on the knowledge enrichment axis.

Tiwari et al's DebateBench (2025) collected about 30 hours' worth of competitive debate speech transcripts in Parliamentary Debate format and annotated these transcripts with the scores earned by these speeches as determined by the WUDC judging manual. Tiwari et al proceeded to test how accurately OpenAI's GPT-o1 and GPT-4o, and Claude Haiku 3.5 were able to score speech transcripts and predict speaker ranks when given the entire judging manual as context. They found that even the best-performing judge models, GPT-o1 for the ranking task and Haiku 3.5 for scoring, were unreliable, in part because the task of following and reasoning with the judging manual required "extensive context requirements", presumably due to the manual's complexity and length; after all, it is about

60 pages long. (Tiwari et al., 2025) Unfortunately, the DebateBench dataset available on HuggingFace only includes the transcripts, not the ground-truth scores as advertised, but the benchmarking methods discussed in the paper may prove useful for researchers trying to validate their own LLM judges and rubrics for debate generation tasks.

3 Core Ideas/Methodology

Before we delve into our methodology, it may be in order to take some time to review basic definitions and conventions used in competitive debate, for those not yet aware of them. The typical debate revolves around a single statement called the "motion" (for example ""This House opposes the norm to prefer the natural to the artificial" would be a motion). One team- conventionally called the "proposition"- is assigned to defend the motion, and the other team- "the opposition"- is assigned to oppose it. The standard debate consists of 3 rounds, with every round consisting of one speech from each of the two sides. In the parliamentary debate format, the team that opens the first round gives what is called the "prime minister" speech. Debate competition teams are generally expected to rely on their own knowledge, without access to search or books. The judging manual is often a book of around 50-100 pages. For our purposes we ended up consulting the Sofia WUDC manual, a document of 64 pages, and condensing its most salient points into a rubric of about 800 words.(CAP, 2025)

We had an initial dataset of 20 debate motions. We tried testing how different OpenAI models (GPT-4o-mini, GPT-4.1-mini, GPT-5-mini) performed against themselves when prompted to generate speeches for a competitive debate. Their outputs were scored using GPT O3 as a judge using a very short self-designed rubric of one paragraph and 5 possible scores; however, it turned out that this rubric was scoring things too leniently (with scores even for these small and naively prompted models averaging 4.35), and we had to throw out our results.

After scrapping this rubric and writing a new and much longer one of 800 words, we engineered lengthy, detailed prompts for each stage of the debate. Prime minister speeches, response speechs, and concluding speeches had their own unique prompts.

Next, we implemented a more complex architecture with the aid of the Dspy framework. This architecture would involve a preliminary module that would take in the motion and the prime minister's assigned position on that motion and output a list of suggested argument directions, which would then be fed to the model for generating the prime minister speech. We add a special layer for argument extraction and refutation in hopes that this will improve the way our responses are structured. The refutations will be passed into the response generator as context. Every speech except the concluding one was then fed to an intermediate layer to extract either its most salient or its weakest arguments. These extracted arguments were then passed into a rebuttal layer instructed through Dspy signature to issue compelling rebuttals to each of the arguments individually and to identify any contradictions. The list of rebuttals would then be passed as context into the response-generating module along with the transcript of the preceding speech and the assigned motion and position.

As a third, additional method separate from this pipeline, we compiled a list of common logic schema- general argument tropes, if you will- popularly used in competitive debate on different domains, like economics, politics & governance, and law & justice. (As an example of what we mean by "general argument trope", take the popular black market argument, which is often used in debates on economic issues to argue against governmental product bans of all sorts.) In our semantic parser layer, we include an output field that extracts the domain of the motion, and from our logic schema store we retrieve the list of schema used in this domain. We then add a "slot-filling" module that, given a motion, outputs arguments following these tropes but applied to our specific motion and position. These arguments are then appended to the list of suggested arguments outputted by the argument-generation layer before these argument suggestions are passed into the speech generation model.

By the time we finished improving our rubric, now an 800 word list, we had lost access to LiteLLM and the models available through it. Instead of re-running the experiment from earlier, we moved onto testing if our first two methods, e.g. more sophisticated prompting and multi-model chaining, would improve the generated speeches for OpenAI models alone, which we had API access to. This required a rewrite of the Dspy scaffolding which had been written under the assumption that we would have access to LiteLLM. Using the new rubric and OpenAI o3 as our LLM judge, we tested if the longer, more detailed prompts would improve the quality of generated debate speeches over our previous concise prompts. We tested this on two models: OpenAI's gpt-4o and 4o-mini. For each turn the rubric assigned a score between 50 to 100 on 4 axes: argumentation and analysis, engagement and rebuttal, role fulfillment, and clarity of expression, in order of highest to lowest weight; at the end of the debate, the team with the highest average score over turns would be selected as the winner. We saved the scores of the turns as well as the record of which team won. This time, we pit models with our improvement attempts (either stronger prompting or more complex architecture) against the "baseline": the same model but without those improvements, recorded their scores, and counted how many times the "improved" models won against the baselines.

After running these OpenAI model tests, we repeated the test more systematically on Gemini models accessed through Vertex, this time comparing “baseline” model performance (using lengthy prompts, however) with the performance with the additional pipeline layers and with the logic schema layer. We ran these experiments on 9 debate motions for each base model (Gemini 2.5 pro and Gemini 2.5 flash) and each possible architecture pairing. The architecture pairings were matched twice, with their roles (proposition or opposition, opening or responding) switched, as an attempt to account for the possibility of speaker ordering or position impacting performance and skewing results.

Finally, as a sanity check for our LLM judge, we directly matched our baseline Gemini 2.5 pro with Gemini 2.5 mini to verify if the larger model performed better under direct competition with the smaller, as one might expect. Since this was simply a sanity check, we did not match each model pair twice on the same prompt, since this would've been too expensive; instead, we had model pairs alternate between motions on who would be the proposition and who would be the opposition. The motivation for this sanity check was that in our initial experiments using our revised rubric, we found that baseline GPT-4o-mini's speeches (on naive prompts) were actually scored higher than GPT-4o's, contrary to what we should have expected. We wanted to test if this was the fault of the LLM judge systematically favoring lower-quality speeches, or if this was simply a result of the fact that we were testing mini against enhanced mini, and so it did not have to contend with as sophisticated of a opponent, enabling it to make simple and straightforward rebuttals without too much of a score penalty. This sanity check was designed to test if the unexpected behavior held under direct competition.

4 Experimental Results

We gathered data from our experiments pitting our "improved" models with their baseline, tracking the win rate of the models of the "improved" models over the baseline.

When we tested models that had been fed the more detailed prompts against models that had been fed the original, less detailed prompts, we found that the benefits diverged based on the size of the model. For the larger model, GPT-4o, more detailed prompting led the model to win 80% of the time against the model with the original prompts. (Fig. 1) When it was provided with detailed prompts, GPT-4o's average score over the 3 turns was 77.2, compared to 75.467 for the original, less detailed prompts. (Fig. 2) However, for GPT-4o-mini, the behavior was in stark contrast: the model with the more detailed prompt won just 37.5% of the time (Fig. 1), and its average score was 75.583 compared to 76.792 with the less detailed prompt. (Fig. 2) Perhaps interestingly, when fed the less detailed prompt, GPT-4o-mini actually performed better, if scores are any indicator, than GPT-4o (Fig. 2); at this point we had not tried to pit the two models directly against each other to see if this behavior would persist under direct competition.

As for the "enhanced" model that had been fitted with many preliminary and intermediate context layers, this performed much worse than the "baseline" model without these layers. This was true of both the larger model, GPT-4o, in which case the "enhanced" version lost 70% of the time, and the small model, GPT-4o-mini, whose enhanced version lost 100% of the time. (Fig 3) The average score of the more complicated model over 3 turns was 73.83 compared to 76.87 for the baseline. (Fig 4) For mini, the difference was even more stark: the average score of the more complicated model over 3 turns was 70.2 compared to 76.93 for the baseline mini. (Fig 4)

We then ran similar experiments on Gemini 2.5 pro and Gemini 2.5 flash, as stated in the preced-

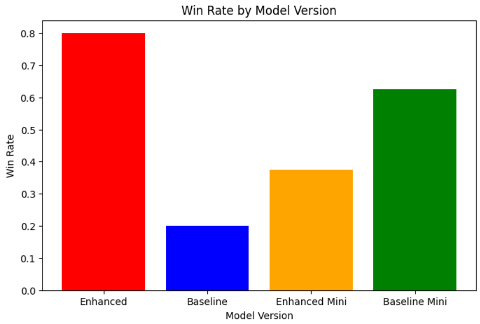

Figure 1: Win rate of the "enhanced" models vs the "baseline" models, using gpt-4o (the two left columns) and gpt-4o-mini (the two right columns). Here "enhanced" just means we made the prompts for each turn more detailed and rigorous

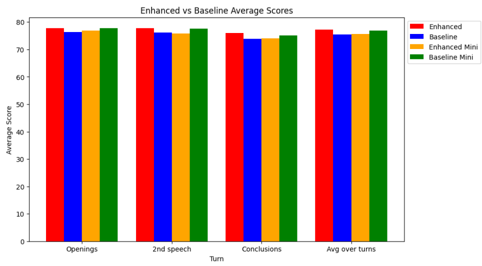

Figure 2: The scores of the "enhanced" models (again, gpt-4o and gpt-4o-mini) vs the "baseline" models (same models but with less in-depth prompting) on each of the 3 turns plus the average score over the turns. "Enhanced", "baseline", and "mini" mean the same thing as above.

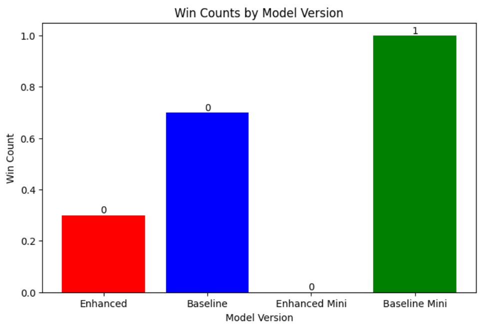

Figure 3: Win rate of the "enhanced" models vs the "baseline" models, using gpt-4o (the two left columns) and gpt-4o-mini (the two right columns). Here "enhanced" means we added many intermediate modules for what we thought might be helpful subtasks

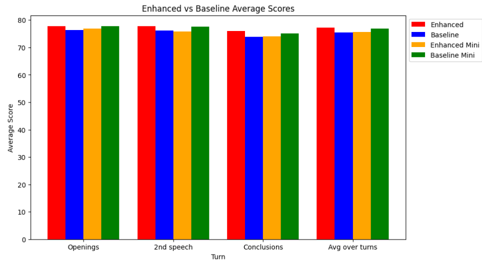

Figure 4: The scores of the "enhanced" models (again, gpt-4o and gpt-4o-mini) vs the "baseline" models (same models but without the added layers) on each of the 3 turns plus the average score over the turns. "Enhanced", "baseline", and "mini" mean the same thing as above.

ing section, but due to having misplaced the baseline prompts from earlier, this would be on the revised, longer prompts. We found that both the extnsive pipeline and logic schema additions led to an abysmal win rate against the baseline: of 11.1% each for the Gemini 2.5 pro and 5.56% for Gemini 2.5 flash. (Figure 5)

With Gemini 2.5 Pro, the argument-rebuttal pipeline "enhancements" significantly worsened performance as scored by the OpenAI o3 judge: from an average of 80.2 across turns for the baseline to 77.5 for the pipeline. The logical schema scored even lower, with an average of 76.3 across turns. As for the smaller Gemini 2.5 Flash, we see similar behavior: the baseline scores an average of 77.6, compared to 75.0 with the added pipeline, and an abysmal 71.9 with the schema. It may be worth clarifying again that these experiments were done with the Method 3 schema pipeline completely separate from the Method 2 pipeline; they were not combined. (Figure 6)

As for the sanity check results, we did not find anything amiss when we pitted Gemini 2.5 Pro against Gemini 2.5 Flash; Pro won 100% of the time, and its average score was 79.7, higher- as expected- than Flash's 76.3. (Figure 7)

5 Insights and Discussion

While we were able to see some improvement in debate speeches generated by the larger GPT-4o model simply through more detailed prompting, this improvement did not carry over to the smaller model. This makes sense due to smaller models having shallower attention mechanisms, which prevent them from handling long, detailed, and complex contexts over multiple steps. Our attempts at

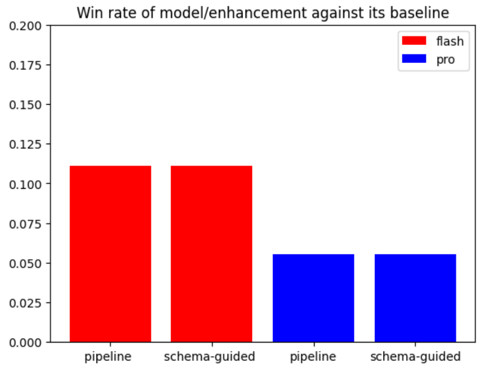

Figure 5: The win rates of the "enhanced" models (e.g. with the 2nd method pipeline or with the 3rd method logic schema) against the "baseline" models (same models but without the added layers). The models used were Gemini 2.5 Flash and Gemini 2.5 Pro.

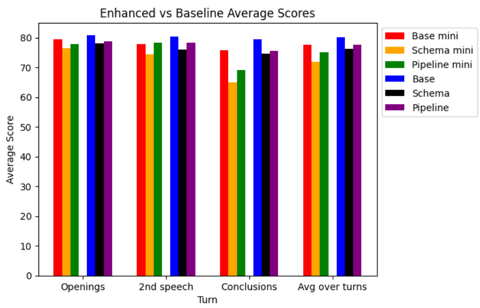

Figure 6: The scores of the base and "enhanced" models (e.g. with the 2nd method pipeline or with the 3rd method logic schema). The models used were Gemini 2.5 Flash (here labeled as "mini") and Gemini 2.5 Pro.

	Motion	Flash role	Prop role	Winner	Flash scores	Pro scores	Turns	Reason
0	This House Believes that developing countries ...	Proposition	Opposition	Pro	[78, 76, 74]	[80, 82, 79]	[Turn(speaker_position='prop_1', team='Proposi...	1. Employment & Inclusivity (largest clash) •...
1	This House regrets the norm of association bet...	Opposition	Proposition	Pro	[79, 77, 76]	[82, 83, 80]	[Turn(speaker_position='prop_1', team='Proposi...	1. Core Clash – Is harm baked into the norm o...
2	This House believes that China should pursue a...	Proposition	Opposition	Pro	[77, 75, 72]	[82, 80, 78]	[Turn(speaker_position='prop_1', team='Proposi...	1. Argumentation & Analysis (45 %) • Economic...
3	This House opposes the expectation that romant...	Opposition	Proposition	Pro	[80, 77, 76]	[80, 79, 78]	[Turn(speaker_position='prop_1', team='Proposi...	1. Definition & Burden of Proof – Prop 1 set ...
4	This House would heavily ease labour regulatio...	Proposition	Opposition	Pro	[76, 78, 77]	[80, 82, 81]	[Turn(speaker_position='prop_1', team='Proposi...	1. Central Clash – Aggregate Demand vs. Labou...
5	This House regrets the glorification of the ca...	Opposition	Proposition	Pro	[77, 75, 74]	[78, 76, 77]	[Turn(speaker_position='prop_1', team='Proposi...	1. Definition & Burden – Prop 1 gave a clear,...
6	This House believes that works of modern ficti...	Proposition	Opposition	Pro	[77, 75, 72]	[80, 78, 77]	[Turn(speaker_position='prop_1', team='Proposi...	1. Argumentation & Analysis (45%) • Proposit...
7	This House opposes the norm to prefer the natu...	Opposition	Proposition	Pro	[79, 78, 77]	[82, 80, 78]	[Turn(speaker_position='prop_1', team='Proposi...	1. Argumentation & Analysis • Proposition set...

Figure 7: The results of the sanitycheck when run on Gemini 2.5 Flash (here labeled as "mini") and Gemini 2.5 Pro using th lengthier prompts. Perhaps we should have run on GPT-4o and GPT-4o-mini and on the naive prompts since that's where we got the odd behavior earlier, but I lost access to those promps, and it is too late for that now

architecture enhancements performed worse than the baseline at both sizes, probably because the added layers actually constricted the outputs and because of error cascading across the chain of additional models.

One possible explanation for any unexpected discrepancies may be that our LLM judge just wasn't as accurate as we hoped, despite our detailed rubric and using an intelligent model as our "judge". To see if this was the case, we considered testing our LLM judge on past debates with annotated human scores, but the main HuggingFace dataset we had found, DebateBench, did not seem to come with scores. This leads us to suggestions for future work.

6 Future work

Though our augmented pipeline design ended up hindering the performance of the Gemini and GPT models we tested, it is still possible that with careful design changes one may find an architecture that actually improves upon their performance. We invite future researchers to experiment with the inputs and outputs of our intermediate argument extraction, rebuttal, and logical schema slot-filling layers as well as the input fields of the final speech-generating model to ensure the output model is not constricted in its scope. In addition, since we only tested on commercial, closed-source, closed-weight, and relatively large models that we had access to, whose gaps their creators must have expended substantial resources to close, one promising direction of future research would be to test if our augmented pipelines do improve the performance of open-source, smaller, more accessible models. This would be interesting as larger closed-source models are not ideal for all purposes, such as running locally or further domain-specific training. If our architecture augmentations are found to be unhelpful to open-source performance too, then researchers can progress to the compute-expensive step of last-resort: fine-tuning.

Future research can also focus on expanding context windows so that model turns can focus on the whole debate, instead of just rebuttals to the immediately preceding speech. Another area of research could be to expand on our logic schema modules; instead of simply identifying relevant debate strategies and argument tropes from a set list and doing primitive slot-filling, it may prove useful to retrieve successful, fully fleshed out examples of these tropes from past debates as a style reference.

Because existing benchmarks like DebateBench appear to be inaccessible- in the case of DebateBench, the dataset on HuggingFace only contains transcripts of speeches and not the scores as advertised in the DebateBench paper- future research may involve compiling a list of past competitive debates transcripts scraped from the Web and YouTube, accompanied by their human scores if available. The rationale for this is that it would allow us to test if our LLM judge (with the same 800-word rubric and the same OpenAI o3 reasoning model) truly approximates human judgment, and if not, to continually refine our rubric until it does, or take other steps towards convergence. We had hoped to do this but ran out of time.

7 Conclusions

Though our multi-step model chaining and logic schema did not improve performance above the SOTA closed-source vanilla models, the marginal improvements we obtained through better prompting demonstrates prompt engineering's potential for improving the structure and expressiveness of LLM-generated debate speeches. Further work is needed before we can discard our architecture augmentations on smaller open-source models, which we did not get a chance to test; however, if we take our results on models like Gemini Flash and GPT-4o-mini as a heuristic, and the relatively small improvements from prompting, we may conclude that the best way forward in the quest for better LLM-generated debates is simply the most computationally expensive: domain-specific fine-tuning.

8 Ethical Considerations

This project focuses on refining LLM performance in a small niche with little application outside that niche, so its ability to cause harm is limited. One could speculate on how access to an improved debate generation model may incentivize cheating in debate tournaments or similar settings, but this is an unlikely scenario that can be easily resolved by requiring participants to do their preparation on devices provided by tournament organizers without access to these models so that the opportunities to cheat are kept to a minimum. Debate tournaments already prohibit or soft-prohibit internet search aids, so this would not be much of an additional imposition. It is the authors' belief- presumably shared with course staff, who assigned us to work on something related to debate agents- that the pedagogical benefits obtained through improving LLM performance far outweigh the limited potential for abuse.

9 Authorship Statement

Kyle wrote initial code using the Dspy framework, which Bhavya later turned into working code. Kyle contributed scaffolding for the argument extration-rebuttal pipeline, while Bhavya contributed the logic schema pipeline. After lengthening the prompts and rubric, Bhavya ran experiments on OpenAI models he had access to on a handful of debate motions, which Kyle post-processed and turned into logical visualizations. Kyle then adapted Bhavya's program so it would fit a more systematic, logical experiment flow and re-ran the experiments with Gemini models accessed through Google's Vertex. Kyle took care of the sanity check experiments as well. Kyle took care of the poster and the report

10 References

References

Sofia WUDC 2026 CAP. 2025. *Debating Judging Manual*.

Noam Slonim, Yonatan Bilu, and Carlos Alzate et al. 2021. An autonomous debating system. *Nature*, 591:379–384.

Utkarsh Tiwari, Aryan Seth, Adi Mukherjee, Kaavya Mer, Kavish, and Dhruv Kumar. 2025. Debatebench: A challenging long context reasoning benchmark for large language models. *Preprint*, arXiv:2502.06279.