CriticalBrew at CQs-Gen 2025: Collaborative Multi-Agent Generation and Evaluation of Critical Questions for Arguments

Roxanne El Baff **Dominik Opitz** **Diaoulé Diallo**
Institute of Software Technology, German Aerospace Center (DLR), Germany
roxanne.elbaff@dlr.de

Abstract

This paper presents the *CriticalBrew* submission to the CQs-Gen 2025 shared task, which focuses on generating critical questions (CQs) for a given argument. Our approach employs a multi-agent framework containing two sequential components: 1) **Generation**: machine society simulation for generating CQs and 2) **Evaluation**: LLM-based evaluation for selecting the top three questions. The first models collaboration as a sequence of thinking patterns (e.g., *debate* → *reflect*). The second assesses the generated questions using zero-shot prompting, evaluating them against several criteria (e.g., depth). Experiments with different open-weight LLMs (small vs. large) consistently outperformed the baseline, a single LLM with zero-shot prompting. Two configurations, agent count and thinking patterns, significantly impacted the performance in the shared task's CQ-usefulness evaluation, whereas different LLM-based evaluation strategies (e.g., scoring) had no impact. Our code is available on GitHub[1].

1 Introduction

Critical thinking is essential in a world overflowing with opinionated texts. Questioning arguments encourages deeper analysis, which can unravel fallacious reasoning (e.g., ad hominem and weak evidence) or strengthen agreement.

Recent research has shown that large language models (LLMs) have excelled in several tasks, including argument mining and question answering. However, it is crucial to acknowledge three issues that arise when using LLMs for generation, as stated by Calvo Figueras and Agerri (2024): hallucination (Huang et al., 2025), the lack of continuous up-to-date knowledge (Gao et al., 2023), and the relativity of what is true (Chang et al., 2024). In their work, Calvo Figueras and Agerri (2024) mitigate these three issues by using LLMs to generate critical questions to uncover the blind spots of argumentative text rather than relying on LLMs' direct answers. For that, they create a reference dataset containing political oral debate arguments (Visser et al., 2021; Lawrence et al., 2018) along with three critical questions by combining a hybrid approach relying on Walton's argumentation theory (Walton et al., 2008) and augmenting their dataset with LLM prompting. The resulting dataset is manually evaluated for relevance and validity. The CQs-Gen 2025 Shared Task (Calvo Figueras et al., 2025) employs this dataset to generate three CQs for an argument.

Building on this task setup, this paper presents our *CriticalBrew* submission to the CQs-Gen 2025 (Calvo Figueras et al., 2025). Our approach employs a collaborative multi-agent framework comprising two sequential components for **generation** and **evaluation**, which aligns with recent trends favoring compound LLM systems over standalone models (Zaharia et al., 2024).

1. Generation. This component builds on the machine society simulation approach by Zhang et al. (2024), originally employed for reasoning tasks (e.g., chess). More precisely, it models agents' collaboration as a sequence of thinking patterns (e.g., *debate* → *reflect*) where each agent impersonates a personality trait, either *easy-going* or *overconfident*. For instance, a society simulation uses n agents where each initially solves a *task*; in this case, generating critical questions for an argument. Then, in r subsequent rounds, each agent re-generates CQs by reflecting on previous answers or debating with other agents. Our experiments exploit several settings based on two main attributes: (i) number of agents (1–3), n, with different combinations of personality traits (e.g., one easy-going and one overconfident), and (ii) number of rounds (0–3), r, with different permutations of thinking patterns

[1] https://github.com/roxanneelbaff/critical_questions_generation

(e.g., two rounds where n agents first debate and then reflect). Zhang et al. (2024) adopt the Society of Mind concept (Minsky, 1986): interacting modules lead to emergent intelligence, aligning with the critical thinking needed for CQ generation.

2. Evaluation. This component selects the top three questions from the first component based on quality criteria, using zero-shot prompting. It explores several methods, including ranking a set of CQs, scoring each CQ, and using a two-stage prompting approach for scoring. Each of these methods assesses criteria such as *depth*, *reasoning*, and *specificity*.

To our knowledge, this is the first collaborative multi-agents approach with LLM-based evaluation in the computational argumentation field.

Our experiments use three open-weight LLMs with different size ranges: Llama-3.1 8B, Mistral Small 3.1 (24B), and Llama-3.3 8B. We report their performance using the overall punctuation, the task's evaluation metric. This score is based on the semantic similarity between a generated CQ and reference data, followed by labeling each CQ as useful or not. The score corresponds to the proportion of CQs labeled useful. The highest overall punctuation was **0.78** on the validation and **0.55** on the test sets. Results show that for the **Generation** component, employing more agents improves models' performance. However, the number of rounds has no effect. Additionally, thinking patterns (e.g., only reflecting vs. only debating) significantly impact performance, unlike personality traits.

2 Related Work

Recent research has explored the use of large language models (LLMs) in the field of computational argumentation. Intersecting this trend with the increasing use of multi-agent systems, our approach combines both directions.

LLM in Computational Argumentation. Recent research in computational argumentation explores the potential of LLMs in tackling existing and new problems (Chen et al., 2024; El Baff et al., 2024; Ziegenbein et al., 2024). For example, Chen et al. (2024) assess LLMs on argument mining and generation tasks, showing their effectiveness with little or no training data, using zero- or few-shot prompts. In turn, Calvo Figueras and Agerri (2024) generate critical questions using a hybrid approach boosted by an LLM for a given argument. Our approach leverages large language models (LLMs) without relying on training data.

LLM Agents as Collaborators. Current work shows that compound LLM systems outperform a standalone LLM (Zaharia et al., 2024; Yao et al., 2023). Our approach adapts Zhang et al. (2024)'s approach, which deploys multi-agent LLM societies, impersonating different personality traits and collaborating via thinking patterns (debate or reflection). These simulations are tested on logic-based tasks (e.g., chess). In contrast, we employ this approach within computational argumentation, detailed in Section 4.1.

LLM Agents as Evaluators. LLMs are also increasingly used as evaluators (Kim et al., 2023), with different methods proposed. Liu et al. (2023) scores a text criterion (e.g., "evaluate coherence") per prompt, while Qin et al. (2023) and Sun et al. (2023) use ranking for evaluation. Our use of LLMs as evaluators is not exhaustive. It rather focuses on a subset of methods, such as ranking, scoring, and two-step prompting, to evaluate the critical questions and pick the top ones, as detailed in Section 4.2.

3 Task Description and Data

We describe the CQs-Gen 2025 dataset and evaluation (Calvo Figueras et al., 2025), used in our experiments.

Overview. CQs-Gen promotes critical thinking by automatically generating useful critical questions (CQs) given an argumentative text. More precisely, given a real oral debate intervention, a model generates three CQs to challenge it.

Dataset. The dataset consists of oral debates from the U.S. 2016 elections (Visser et al., 2021) and the Moral Maze (Lawrence et al., 2018). Each entry consists of one intervention, its corresponding CQs, and other metadata, such as argumentation schema. Each CQ is labeled for its *usefulness* in challenging the given intervention. A CQ can be either *useful* if it challenges the argument, *unhelpful* if it is valid but unlikely to challenge the argument, or, otherwise, *invalid*. The validation set comprises 186 labeled entries.

Evaluation. The CQs-Gen evaluation script first checks if the generated CQ is similar to one of the *useful* CQs in the reference data. If similarity is detected, the CQ is then labeled as *useful*, *unhelpful*,

or *invalid*.[2] The performance of a model is measured by the *overall punctuation*[3] score defined as the proportion of CQs labeled *useful*. We interpret results using this score.

4 Approach

This section outlines our two-component approach[4]: a **Generation** component that generates critical questions (CQs) for an argument, and an **Evaluation** component that selects 3 questions. Below, we describe each component.

4.1 Generation via Collaboration

The Generation component takes an argument and outputs a set of CQs. Initially, each agent generates three questions using a zero-shot prompt. Then, the system applies a sequence of thinking patterns over r rounds, resulting in $3 \times n$ CQs. This approach is adapted from Zhang et al. (2024). Below, we explain the concepts underlying social simulation and then detail how it works.

The Concepts for Collaboration

Zhang et al. (2024) explore collaboration mechanisms with multiple agents by focusing on three concepts: **individual traits** assigned to each agent, **thinking patterns** applied in each round, and a **collaborative strategy** defining their sequence.

Individual Traits. The framework defines two agent traits: *easy-going* (t_e) associated with democratic harmony (Mutz, 2006; Held, 2006) and *overconfident* (t_o), more resistant to others' opinions (Moore and Healy, 2008).

Thinking Pattern. Zhang et al. (2024) explore two thinking patterns: **debate** (p_d) and **reflect** (p_r). Each pattern defines how an agent regenerates new CQs based on the answers from the previous round. In the debate pattern, each agent considers all the agents' answers, including their own, while in the reflection, they consider only their own.

Collaborative Strategy. A collaborative strategy defines the sequence of thinking patterns applied in rounds. At each round, all agents employ the same thinking pattern, p_r or p_d (Du et al., 2023).

[2] If no similarity is found between the generated CQ and any useful reference CQ, the generated CQ is labeled as *unable to label*, requiring manual evaluation.

[3] *Score* and *overall punctuation* are used interchangeably.

[4] Our initial approach included an *argument mining* step where we transformed each argument into a structured text, decomposed into argument components, but this step did not perform well. See appendix A for more details.

Evaluation	# Prompts	Description
Basic	1	A single prompt selects the top n critical questions (CQs) based on evaluation criteria (depth, relevance, reasoning, and specificity).
Scoring	1	A single prompt scores all CQs from 1–5 across all criteria and averages the result.
Ranking	# criteria	For each criterion, a prompt ranks all CQs in order of quality (e.g., depth).
Two-Step	2 $\times$ # criteria $\times$ # CQ	For each CQ-criterion, one prompt presents the argument and CQ, then another prompt scores a criterion.

Table 1: Overview of the LLM-based evaluators. For each method (*Evaluation*), we report the number of prompts per argument (*# Prompts*) and a *Description*. $\#criteria$ refers to the number of evaluation criteria, and $\#CQ$ refers to the number of critical questions.

These concepts are employed in a Machine society, as explained next.

Machine Society Simulation

Similar to Zhang et al. (2024), a machine society has n LLM agents, each with a trait (t_e or t_o), collaborating over r rounds of thinking patterns (p_d, p_r). Initially, each agent generates three CQs for an argument. Then, in each round, each agent generates three CQs. If the society has more than one agent, we use the evaluator component (Section 4.2); otherwise, we return the agent's output.

4.2 LLM Agents as Evaluators

A machine society can output more than n critical questions when it includes at least two agents. For that, we employ LLM-based evaluator agents to choose the top 3 CQs (Table 1). We employ four methods, focusing on criteria selected based on findings from Calvo Figueras and Agerri (2024): depth, relevance, reasoning, and specificity. We list the four methods below, ordered by the number of prompts needed per task.

Basic. An agent is prompted with an argument and list of CQs to select the top 3 CQs. Our prompt instructs the agent to select top CQs based on the criteria mentioned (Appendix C).

Scoring. Similar to basic, an agent is prompted with an argument and list of CQs. However, the agent, using one prompt, is instructed to score each criterion for each CQ, similar to (Kim et al., 2023),

from 1 to 5. Then, the top 3 CQs are selected based on the highest mean value of criteria scores.

Ranking. We employ one agent per criterion (depth, reasoning, relevance, and specificity) to rank (Sun et al., 2023) the set of CQs. An agent is prompted with an argument and a list of CQs and returns the ranked CQs for a specific criterion. Then, the top 3 CQs are selected based on the highest mean ranks of all criteria.

Two-step Prompting. We employ a two-step prompting strategy designed for complex reasoning tasks (Seo et al., 2025; Hama et al., 2024). The LLM is prompted with the argument and one CQ in the first step. In the second step, given a critical question, we prompt an agent for each criterion: depth, reasoning, and specificity. The top 3 CQs with the highest criteria average are then selected.

5 Experiments and Results

This section reports our experiment settings, results on the validation set, and Gen-CQs 2025 submission results on the test set.

Our experiments simulate machine societies based on two configurations: the number of agents and rounds. More precisely, we employ n agents where $1 \leq n \leq 3$, and r rounds where $0 \leq r \leq 3$. In total, we simulate 113 machine societies[5].

5.1 Settings and Baselines

We run each simulation using three open-weight LLMs, varying in their parameter size: Llama-3.1 8B (L8B), Mistral Small 24B (M24B), and Llama-3.3 70B (L70B) (Touvron et al., 2023). For our implementation, we use the LangGraph Python agent framework[6], along with LangChain, allowing us to output structured data and save each *state* as a JSON object for each LLM answer (Appendix D).

To test the **1. Generation** component, we run all simulations with the *basic* evaluator defined in Section 4.2. The best-performing simulations are defined based on the highest score[7] (§3). Then, to test the **2. Evaluation** component, we rerun the best simulation per LLM type from the previous stage with the evaluation methods defined in §4.2. All our results are reported on the *validation set* with 186 arguments.

[5]We have 9 agent-trait groups and 15 pattern sequences, yielding 135 simulations, but, for single agent (t_e or t_o), 11 of the 15 sequences are excluded for containing p_d, resulting in 135 – 22 = 113 combinations.

[6]https://langchain-ai.github.io/langgraph/

[7]Referred to as the *overall punctuation* in CQs-Gen 2025.

LLM		Agents	Rounds	Pattern	Traits	Score	3/3 %
Llama 8B	↑	3	3	$p_r p_d p_d$	$t_e t_e t_e$	0.71	0.40
		3	2	$p_r p_d$	$t_e t_e t_e$	0.70	0.42
		2	2	$p_d p_d$	$t_e t_e$	0.69	0.42
	↓	1	1	p_r	t_o	0.59	0.26
		2	2	$p_r p_r$	$t_e t_e$	0.59	0.23
		2	2	$p_d p_r$	$t_e t_o$	0.60	0.31
	○	1	0	–	t_e	0.68	0.39
		1	0	–	t_o	0.66	0.29
		1	0	–	–	0.68	0.38
Mistral 24B	↑	3	3	$p_d p_d p_r$	$t_e t_e t_o$	**0.78**	0.54
		3	0	–	$t_e t_e t_e$	0.76	0.53
		3	1	p_r	$t_e t_o t_o$	0.76	0.53
	↓	1	3	$p_r p_r p_r$	t_o	0.70	0.42
		2	3	$p_d p_d p_r$	$t_e t_e$	0.70	0.41
		3	3	$p_r p_d p_r$	$t_o t_o t_o$	0.71	0.40
	○	1	0	–	t_o	0.74	0.45
		1	0	–	t_e	0.73	0.47
		1	0	–	–	0.73	0.43
Llama 70B	↑	2	3	$p_d p_d p_d$	$t_e t_e$	**0.78**	0.53
		2	3	$p_r p_r p_d$	$t_e t_o$	0.77	**0.55**
		3	2	$p_d p_r$	$t_o t_o t_o$	0.77	0.53
	↓	3	3	$p_r p_r p_r$	$t_e t_e t_e$	0.71	0.45
		3	3	$p_d p_r p_r$	$t_e t_e t_o$	0.71	0.41
		3	3	$p_d p_r p_d$	$t_e t_e t_o$	0.71	0.39
	○	1	0	–	t_o	0.73	0.44
		1	0	–	t_e	0.72	0.43
		1	0	–	–	0.73	0.44

Table 2: Performance of the three LLMs on the validation set ($N = 186$), showing top (↑), worst (↓), and baseline (○) setups. Each machine society is defined by number of *Agents*, number of *Rounds*, Thinking *Pattern* (p_d, p_r), and Personality *Traits* (t_e, t_o). The overall punctuation (*Score*) is reported as a proportion of *useful* questions, and 3/3 is the argument rate where 3 questions were labeled useful.

Baseline. We use three baselines, each of which is a standalone LLM with zero-shot prompting to generate the three CQs with three settings: with no personality trait, t_o, and t_e. The baselines are shown in Table 2 marked with (○).

5.2 Generation Component Results

Table 2 summarizes the best, worst, and baseline simulation per LLM.

Overview. All three LLM of different sizes outperform their baselines: L8B (0.71) by 3%, and M24B and L70B (0.78) by 4-5%. Despite being significantly smaller, the M24B model performs comparably to the L70B model. Also, for the *3/3* %, both models achieve a range of $0.53 - 0.55$

LLM	Evaluation	Pattern	Traits	Score	3/3 %
Llama 8B	**Basic**	$p_r p_d p_d$	$t_e t_e t_e$	0.71 (0.78)	0.40
	Scoring			0.69 (0.77)	0.42
	Ranking[3]			0.71 (0.80)	0.42
	2-step			0.66 (0.76)	0.36
Mistral 24B	**Basic**	$p_d p_d p_r$	$t_e t_e t_o$	**0.78** (0.82)	0.54
	Scoring[1]			**0.78** (0.82)	**0.58**
	Ranking			**0.78** (0.82)	0.57
	2-step			0.76 (0.80)	0.52
Llama 70B	**Basic**	$p_d p_d p_d$	$t_e t_e$	**0.78** (0.82)	0.53
	Scoring[2]			**0.78** (0.82)	0.55
	Ranking			**0.78** (0.82)	0.54
	2-step			0.77 (0.80)	0.55

Table 3: Performance of the three LLMs, showing their best machine society configurations (from Table 2) across **Basic**, **Scoring**, **Ranking**, and **Two-step**. Each society is defined by a thinking *Pattern* (p_d, p_r) and Personality *Traits* (t_e, t_o). *Score* is the overall punctuation, and 3/3 is the rate of arguments with all questions useful. Best per-LLM is underlined, overall best in **bold**, and submitted simulations marked with [1,2,3].

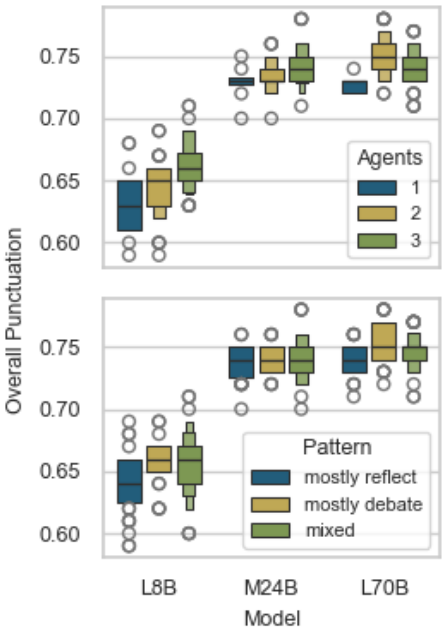

Figure 1: Boxplots for the *Overall Punctuation* per LLM (L8B, M24B and L70B) with two configurations: ***Agent** Counts* (Top) and *Thinking **Pattern*** (Bottom).

compared to 0.42 for the smallest model, L8B.

For each configuration (e.g., number of agents), we measured significance using ANOVA in cases of normality (Kruskal-Wallis otherwise). If $p < 0.05$, we conducted posthoc analysis (independent t-test in case of normality, Mann- Whitney otherwise) with Bonferroni correction. We report below the results where $p < 0.05$.[8]

Number of Agents. Agent count significantly influenced performance for all models; L8B, M24B[9] and L70B ($p < 0.05$). For L70B, both two- and three-agent setups outperformed single-agent. For L8B, three agents performed significantly better than one or two. Figure 1-Top shows score distributions per model and agent count.

Thinking Patterns We compare three types: *mostly debate*, *mostly reflection* and *mixed* (at least one round of each). We report a significant effect for L8B and L70B[9] ($p < 0.05$). For L8B, *mostly reflection* differed significantly from the other two. Figure 1-Bottom shows score distributions per model and pattern type.

5.3 Evaluation Component Results

We re-evaluate the CQs of the best-performing society simulation configurations from Table 2 using the four evaluation methods. The scores shown in Table 3 are similar across all methods within each LLM, especially for *Basic*, *Scoring*, and *Ranking*. A more complex method (two-step) does not yield better results. For the *3/3 %*, M24B achieved the highest score with 0.58.

5.4 Submission

We submit CQs from the best models per LLM type (Table 3). As performance is similar across *evaluators*, we manually inspect the test set ($N = 34$) and choose semantically diverse CQs per argument (submitted simulations are marked with [1,2,3] in Table 3). Only the first submission outperforms the shared task's baselines, with an overall punctuation of **0.55**: **M24B** uses three agents (2 × easy-going, 1 × overconfident), three patterns (p_d, p_d, p_r), and *scoring* evaluation.

6 Conclusion

In this work, we employed a social machine framework to generate critical questions (CQs) for an argument that had been previously adapted in logical domains. Our approach outperformed standalone LLMs. We found that the number of collaborating agents and the choice of thinking pattern have a positive impact on the generation of CQs. However, alternative evaluation strategies do not show any additional benefit. To enable further investigation, we release our experimental data.

[8] number of rounds, and the personality trait (mixed traits vs. one-type trait) had no to little significant effect.

[9] Posthoc results had no pairwise significance difference.

References

Blanca Calvo Figueras and Rodrigo Agerri. 2024. Critical questions generation: Motivation and challenges. In *Proceedings of the 28th Conference on Computational Natural Language Learning*, pages 105–116, Miami, FL, USA. Association for Computational Linguistics.

Blanca Calvo Figueras, Jaione Bengoetxea, Maite Heredia, Ekaterina Sviridova, Elena Cabrio, Serena Villata, and Rodrigo Agerri. 2025. Overview of the critical questions generation shared task 2025. In *Proceedings of the 12th Workshop on Argument Mining*.

Tyler A Chang, Katrin Tomanek, Jessica Hoffmann, Nithum Thain, Erin MacMurray van Liemt, Kathleen Meier-Hellstern, and Lucas Dixon. 2024. Detecting hallucination and coverage errors in retrieval augmented generation for controversial topics. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4729–4743.

Guizhen Chen, Liying Cheng, Anh Tuan Luu, and Lidong Bing. 2024. Exploring the potential of large language models in computational argumentation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2309–2330, Bangkok, Thailand. Association for Computational Linguistics.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Roxanne El Baff, Khalid Al Khatib, Milad Alshomary, Kai Konen, Benno Stein, and Henning Wachsmuth. 2024. Improving argument effectiveness across ideologies using instruction-tuned large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4604–4622, Miami, Florida, USA. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2.

Kenta Hama, Atsushi Otsuka, and Ryo Ishii. 2024. Emotion recognition in conversation with multi-step prompting using large language model. In *Social Computing and Social Media*, pages 338–346, Cham. Springer Nature Switzerland.

David Held. 2006. *Models of democracy*. Polity.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. 2023. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

John Lawrence, Jacky Visser, and Chris Reed. 2018. Bbc moral maze: Test your argument. In *Comma*.

Jingjing Li, Yifan Gao, Lidong Bing, Irwin King, and Michael R. Lyu. 2019. Improving question generation with to the point context. 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 3216–3226, Hong Kong, China. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Marvin Minsky. 1986. *Society of mind*. Simon and Schuster.

Don A Moore and Paul J Healy. 2008. The trouble with overconfidence. *Psychological review*, 115(2):502.

Diana C Mutz. 2006. *Hearing the other side: Deliberative versus participatory democracy*. Cambridge University Press.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, et al. 2023. Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*.

Seongbum Seo, Sangbong Yoo, and Yun Jang. 2025. A prompt chaining framework for long-term recall in llm-powered intelligent assistant. In *Proceedings of the 30th International Conference on Intelligent User Interfaces*, pages 89–105.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agents. *arXiv preprint arXiv:2304.09542*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Ramani, Rohan Taori, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Jacky Visser, John Lawrence, Chris Reed, Jean Wagemans, and Douglas Walton. 2021. Annotating argument schemes. *Argumentation*, 35(1):101–139.

Douglas Walton, Christopher Reed, and Fabrizio Macagno. 2008. *Argumentation schemes*. Cambridge University Press.

Douglas N. Walton. 1996. *Argumentation Schemes for Presumptive Reasoning*. Lawrence Erlbaum Associates, Mahwah, NJ.

Yiqiu Xu, Alessio Frosini, Mattia Vanni, Anisa Rula, and Roberto Navigli. 2024. Frase: Frame-based semantic enrichment for sparql query generation. *arXiv preprint arXiv:2503.22144*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Matei Zaharia, Omar Khattab, Lingjiao Chen, Jared Quincy Davis, Heather Miller, Chris Potts, James Zou, Michael Carbin, Jonathan Frankle, Naveen Rao, and Ali Ghodsi. 2024. The shift from models to compound ai systems. https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/.

Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. 2024. Exploring collaboration mechanisms for LLM agents: A social psychology view. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14544–14607, Bangkok, Thailand. Association for Computational Linguistics.

Timon Ziegenbein, Gabriella Skitalinskaya, Alireza Bayat Makou, and Henning Wachsmuth. 2024. LLM-based rewriting of inappropriate argumentation using reinforcement learning from machine feedback. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4455–4476, Bangkok, Thailand. Association for Computational Linguistics.

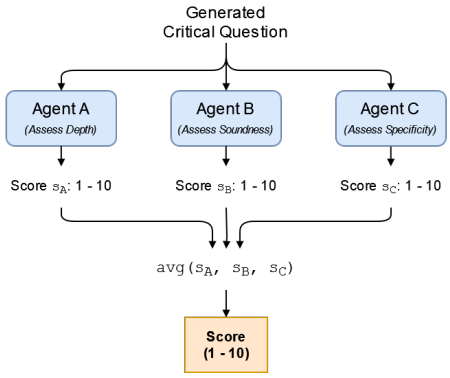

Figure 2: Schematic process of scoring generated questions where an LLM scores one criterion.

A Pre-step - Argument Formulizer

We explored an additional step with a Formulizer agent. Inspired by Walton's Argumentation Schemes (Walton, 1996), the Formulizer transforms each natural-language argument into a structured representation. Specifically, arguments were decomposed into a main claim, explicit supporting premises, implicit assumptions, potential areas of weakness, and conclusions when present. The intuition behind this approach was that structured formulation could help the Generator agent to identify critical questions by explicitly highlighting ambiguous premises, unstated assumptions, and argumentative weaknesses. Prior work in other domains has shown that structured semantic representations can improve generation quality (Li et al., 2019; Xu et al., 2024) of large language models. Our preliminary experiments indicated that large models, such as GPT-based systems, might benefit from this structured context. However, when applying the Formulizer on smaller, open-source models selected for the main experiments, no measurable improvement was found. This suggests that, in the context of critical question generation, additional structured input might help sufficiently powerful models but could introduce confusion or unnecessary complexity for smaller models. Based on our findings, we have not included the Formulizer agent into the final system.

B Evaluator

Within the two-sept evaluator, each LLM evaluate one criterion for each CQ, as shown in Figure 2.

C Prompts

All prompts can be found here: https://github.com/roxanneelbaff/critical_questions_generation/tree/main/prompts.

D Technical Details

Implementation We used LangGraph to build the multi-agents workflows for machine societies and for the LLM-based evaluators. Also we use LangChain[10] along with TogetherAI[11]. All three models were loaded via the TogetherAI API and are as follows:

- Llama 3.1 8B: `meta-llama/Llama-3.1-8B-Instruct`
- Mistral Small (24B): $mistralai/Mistral\text{-}Small24BInstruct2501$
- Llama 3.3 70B: meta-llama/Llama-3.3-70B-Instruct

We used the default temperature, 0.7, when running our experiments.

Data For each workflow, representing a machine society, we save each *state*; a state represents the output of all LLMs after being prompted in a round: whether to initially generate the three critical questions, reflect or debate. Also the scores/ranks from the 4 evaluators are saved at each stage. This will allow for expanded analysis.[12]

[10] https://www.langchain.com
[11] https://www.together.ai
[12] https://github.com/roxanneelbaff/critical_questions_generation/tree/main/output