ARG2ST at CQs-Gen 2025: Critical Questions Generation through LLMs and Usefulness-based Selection

Alan Ramponi,[1] **Gaudenzia Genoni,**[2] **Sara Tonelli**[1]
{alramponi,satonelli}@fbk.eu, gaudenzia.genoni@studenti.unitn.it

[1] Fondazione Bruno Kessler, Italy
[2] University of Trento, Italy

Abstract

Critical questions (CQs) generation for argumentative texts is a key task to promote critical thinking and counter misinformation. In this paper, we present a two-step approach for CQs generation that *i)* uses a large language model (LLM) for generating candidate CQs, and *ii)* leverages a fine-tuned classifier for ranking and selecting the top-k most useful CQs to present to the user. We show that such usefulness-based CQs selection consistently improves the performance over the standard application of LLMs. Our system was designed in the context of a shared task on CQs generation hosted at the 12th Workshop on Argument Mining, and represents a viable approach to encourage future developments on CQs generation. Our code is made available to the research community.[1]

1 Introduction

In the rapidly evolving field of argument mining (Stede and Schneider, 2018; Lawrence and Reed, 2020), the automated generation of critical questions (CQs) for argumentative texts has recently been introduced as a task to foster individuals' critical thinking and counter misinformation (Calvo Figueras and Agerri, 2024). CQs are defined as *the set of inquiries that could be asked in order to judge if an argument is acceptable or fallacious* (Calvo Figueras and Agerri, 2024) and have been proven useful for identifying fallacies (Musi et al., 2022; Ramponi et al., 2025) and evaluating argumentative essays (Song et al., 2014). Unlike automated fact-checking tasks that assign veracity labels to claims (Gupta and Srikumar, 2021; Valer et al., 2023, *inter alia*), CQs generation advances misinformation countering by moving beyond the absolutist notion of truth and offering a means to identify missing or potentially misleading arguments even without access to up-to-date factual knowledge (Calvo Figueras and Agerri, 2024).

To encourage research in this direction, a shared task on CQs generation has been proposed (CQs-Gen; Calvo Figueras et al., 2025) and hosted at the 12th Workshop on Argument Mining. The goal of the shared task is to investigate methods for generating useful CQs given an argumentative text as input. Participants are asked to provide three CQs per argumentative text, which are then subject to semi-automatic evaluation (Section 2).

In this paper, we present our research contribution for CQs generation. Motivated by recent advancements in NLP driven by large language models (LLMs), their pitfalls (e.g., outputs' reliability and consistency), and the shared task requirement of providing exactly $k = 3$ CQs per text, we propose a two-step approach that *i)* uses an LLM for generating n CQs (with $n > k$) and *ii)* leverages a fine-tuned classifier to select the top-k useful CQs to retain based on their confidence scores (Section 3). Results show that our usefulness-based selection leads to performance improvements across all the LLMs tested (Section 4). Finally, we provide a qualitative analysis and insights for future work (Section 5) and outline our conclusions (Section 6).

2 Data and Task Description

In this section, we provide details on the data provided by the shared task organizers (Section 2.1) and describe the task setup (Section 2.2).

2.1 Data Description

The data used for the CQs-Gen shared task is based on Calvo Figueras and Agerri (2025). The validation set provided to participants comprises 186 interventions, either from real debates or online discussions (i.e., argumentative texts). Among these, 80 are drawn from the US2016TV corpus (Visser et al., 2020, 2021), i.e., transcripts from televised debates for the 2016 US Presidential election, 72 from REGULATION ROOM DIVISIVE-

[1] Repository: https://github.com/dhfbk/cqs-gen.

Proceedings of the 12th Argument Mining Workshop, pages 301–313

NESS (RRD) (Konat et al., 2016), a corpus of user comments from the eRulemaking platform RegulationRoom.org, 20 from MORAL MAZE DEBATES (MMD) (Lawrence et al., 2018), a corpus for the homonymous BBC4 radio programme, and 14 from the US2016REDDIT corpus (Visser et al., 2020, 2021), i.e., Reddit posts reacting to the 2016 US political debates. Each intervention is annotated with one or more argumentation schemes based on the Walton et al. (2008)'s taxonomy and is accompanied by a set of CQs, categorized as `useful`, `unhelpful`, or `invalid` according to their effectiveness in challenging the arguments of the intervention (Appendix A). These CQs can be either LLM-generated or manually instantiated by annotators using fixed templates in line with Walton et al. (2008)'s theory, as described in Calvo Figueras and Agerri (2024). The test set instead comprises 34 interventions distributed as follows: US2016TV (17), RRD (11), and MMD (6).

2.2 Task Setup

The CQs-Gen shared task encourages the development of methods to counter misinformation and promote critical thinking. Participants are asked to design a system that, given an argumentative text as input, provides exactly three CQs that challenge the arguments in the intervention. Focusing on the internal structure and content of text, rather than external knowledge, these questions aim to uncover implicit assumptions, expose logical weaknesses, or highlight insufficient evidence.

For evaluation, each generated CQ is assigned the label of the closest reference CQ in the dataset, as determined by semantic similarity (Reimers and Gurevych, 2019).[2] CQs that match `useful` CQs are awarded $0.\overline{33}$ points, while those matching `unhelpful` or `invalid` CQs receive 0 points: therefore, for each intervention, a *punctuation* score between 0 and 1 can be obtained. However, if the similarity between the generated and the most similar reference CQ falls below a given similarity threshold[3] – also when the CQ is useful but it is not included in the reference set – the generated CQ remains unmatched and does not contribute any points to the score, requiring manual evaluation to assess its usefulness. The overall punctuation score for a system is given by the average of all punctuation scores obtained across interventions.

[2]Semantic similarity in the official organizers' evaluation script is computed using the `stsb-mpnet-base-v2` model.

[3]The threshold used in the official evaluation script is 0.60.

3 Methods

Our approach to CQs generation consists of two stages. First, we use an LLM to generate candidate CQs and extract them from the raw output (Section 3.1). Second, we apply a fine-tuned classifier to the CQs, rank them by confidence score, and select the top-k candidate CQs to retain (Section 3.2).

3.1 Generation of Candidate Questions

The generation phase is conducted by prompting an LLM to obtain a raw output containing candidate CQs for a given argumentative text. Models and prompting strategies are described in Section 4.1.

Since LLMs' raw outputs often include extra text before or after the requested output, we carefully curate the post-processing. Specifically, to extract the n CQs from the raw output, we split the text by line breaks and retain only the lines starting with a capital letter that end in a question mark. If less than n CQs are detected, the remaining slots are filled with a placeholder value.

3.2 Usefulness-based Questions Selection

The CQs selection phase leverages a pretrained model that we specifically fine-tune using a dataset of `useful` and `non-useful` (i.e., `unhelpful` and `invalid` merged together) CQs. The fine-tuned model is therefore a binary classifier, and the confidence score for the predicted label is provided. This classifier is applied to all n candidate CQs. Models and dataset compositions that we tested are described in the experimental setup (Section 4.1).

We use the confidence score for the label `useful` as given by the classifier and rank the n candidate CQs by decreasing "usefulness". We then select the top-k CQs and use them as final output.

4 Experiments

In this section, we describe the experimental setup (Section 4.1) and the model selection process (Section 4.2). Then, we present the results (Section 4.3).

4.1 Experimental Setup

Models For the generation of candidate CQs, we experiment with different families of instruction-tuned LLMs of varying sizes in both zero-shot and few-shot settings. Specifically, we use `Llama-3-8B` and `Llama-3-70B` (Grattafiori et al., 2024), `Mixtral-8x7B` (Jiang et al., 2024), and `Qwen-2.5-7B` and `Qwen-2.5-32B` (Qwen et al.,

2025). Hyper-parameter settings for these models are reported in Appendix B.1. For CQs selection, we fine-tune transformer-based models using MaChAmp v0.4.2 (van der Goot et al., 2021) in a single task setting with default hyper-parameter values (Appendix B.1). We employ `BERT-base-uncased` (Devlin et al., 2019) and `RoBERTa-base` (Liu et al., 2019) as encoders, and use `[CLS]` and `<s>` special tokens for classification.

Prompts For CQs generation, we devise a set of prompts that include increasingly-detailed information about the task and the input argumentative text. To isolate the impact of individual information pieces from linguistic variation, we design prompts in a modular fashion (Appendix B.2). Specifically, starting from a prompt with only key information on the task and the desired output (`base`), we experiment with the inclusion of the argumentation schemes associated to the input intervention (`schemes`) and descriptions of what `useful` and `non-useful` CQs are (`desc` in different flavors). The provisionally best-performing prompt is also used for in-context learning experiments (Dong et al., 2024) in few-shot settings. Details on our prompts can be found in Appendix B.2.

Classifier data For fine-tuning the CQs usefulness classifier (Section 3.2), we collect all the CQs in the validation set and their associated labels, and divide the resulting set into 80%/20% train/test portions, i.e., obtaining `gold-train` and `gold-test` splits. We further collect CQs generated by small-sized LLMs (i.e., `Llama-3-8B`, `Qwen-2.5-7B`, and `Mixtral-8x7B`) using the `base` prompt with $n = 3$ across all 5 runs on the validation set (Section 4.2) along with assigned labels, leading to three synthetic sets: `synth-l`, `synth-q`, and `synth-m`, respectively. These different sets, including their concatenation (`all`), are used for determining the best data combination based on macro F_1 score on the `gold-test` split (Section 4.2).

4.2 Model Selection

We use the validation set and the evaluation script provided by shared task organizers[4] for selecting the most promising LLM configurations (i.e., underlying models and prompts) and usefulness-based CQs classifier. To ensure fair comparison between zero- and few-shot settings, we remove from the development set the interventions used in few-shot prompts (Appendix B.2.2). Given the small size of the resulting development set, we run all LLMs with 5 random seeds and select the best approaches based on average punctuation score.

Generation of candidate CQs We start by assessing the performance of small-sized LLMs across all prompts in a zero-shot setting with $n = 3$ to identify promising models and prompts to be used in further experiments. As shown in Appendix C.1, `Mixtral-8x7B` and `Llama-3-8B` outperform `Qwen-2.5-7B` across all prompts; we thus discard the latter from further experimentation. The `base` prompt provides the best overall performance for both `Mixtral-8x7B` and `Llama-3-8B` despite its simplicity. Among prompts with CQ descriptions, providing information only on what `non-useful` CQs are (i.e., $\texttt{desc}_{(\neg U)}$) is more reliable than providing definitions for `useful` CQs (i.e., $\texttt{desc}_{(U)}$) or their combination (i.e., $\texttt{desc}_{(U+\neg U)}$), even when extremely detailed (i.e., $\texttt{desc}_{(FULL)}$). However, $\texttt{desc}_{(\neg U)}$ still lags behind the `base` prompt in terms of performance. We further observe that using `schemes` leads to the worst performance across models. We hypothesize that this is due to the unavailability of precise information about the part of the input intervention where each argumentation scheme occurs. We thus select `Mixtral-8x7B` and `Llama-3-8B` with the `base` prompt for few-shot experiments; however, we observe that this direction is not viable: a substantial performance degradation occurs when including CQs examples in the prompt. Results are in Appendix C.2 to encourage research in this direction.

Selection of useful CQs To choose the CQs selection classifier, we compare the performance of `BERT-base-uncased` and `RoBERTa-base` models when fine-tuned using either `gold-train`, synthetic sets (i.e., `synth-l`, `synth-q`, and `synth-m`), or a combination thereof (i.e., `all`) (Section 4.1). As shown in Appendix C.3, using the `all` set for fine-tuning consistently improves the performance across models, leading to 0.7563 macro F_1 for `BERT-base-uncased` and 0.7341 macro F_1 for `RoBERTa-base`. We therefore select the `BERT-base-uncased` model fine-tuned with the `all` data variant as our CQs selection classifier.

4.3 Results

The best-performing small-sized LLMs and prompts derived from the model selection (Section 4.2) – i.e., `Mixtral-8x7B` and `Llama-3-8B`,

[4] https://github.com/hitz-zentroa/shared-task-critical-questions-generation.

Model	Prompt	n	Selection	Punctuation	
MIXTRAL-8X7B	base	3	no	$0.6758_{\pm 0.01}$	
MIXTRAL-8X7B	base	5	rand	$0.6878_{\pm 0.01}$	
MIXTRAL-8X7B	base	5	yes	$\mathbf{0.7231}_{\pm 0.02}$	*
LLAMA-3-8B	base	3	no	$0.6510_{\pm 0.01}$	
LLAMA-3-8B	base	5	rand	$0.6058_{\pm 0.01}$	
LLAMA-3-8B	base	5	yes	$\mathbf{0.6790}_{\pm 0.01}$	
QWEN-2.5-32B	base	3	no	$0.6543_{\pm 0.01}$	
QWEN-2.5-32B	base	5	rand	$0.6499_{\pm 0.02}$	
QWEN-2.5-32B	base	5	yes	$\mathbf{0.6732}_{\pm 0.02}$	
LLAMA-3-70B	base	3	no	$0.6903_{\pm 0.02}$	
LLAMA-3-70B	base	5	rand	$0.7162_{\pm 0.02}$	
LLAMA-3-70B	base	5	yes	$\mathbf{0.7618}_{\pm 0.02}$	*
LLAMA-3-70B	desc$_{(U+\neg U)}$	3	no	$0.6958_{\pm 0.02}$	
LLAMA-3-70B	desc$_{(U+\neg U)}$	5	rand	$0.6922_{\pm 0.01}$	
LLAMA-3-70B	desc$_{(U+\neg U)}$	5	yes	$\mathbf{0.7279}_{\pm 0.02}$	*

Table 1: Results on the development set for different LLMs and the best prompt strategies in a zero-shot setting with/without CQs selection. We report the average punctuation score with standard deviation across 5 runs with different random seeds. Models for which a best run has been selected for testing are indicated with *.

both with base – as well as large-sized LLMs with promising prompts from preliminary experiments – i.e., Llama-3-70B with base and desc$_{(U+\neg U)}$ and Qwen-2.5-32B with base – are finally compared with and without the CQs selection classifier. Specifically, to assess whether a classifier for selecting the most useful CQs helps in improving performance, we compare the results obtained on the validation set by the aforementioned LLMs when *i)* directly instructed to generate exactly $n = 3$ CQs – with no selection (i.e., "no"), *ii)* instructed to generate $n = 5$ CQs followed by random selection of $k = 3$ CQs (i.e., "rand"), and *iii)* instructed to generate $n = 5$ CQs that are then given to the usefulness-based CQs classifier to keep the top-k ($k = 3$) most useful CQs (i.e., "yes"). Results in Table 1 show that using the usefulness-based CQs classifier (i.e., "yes") consistently improves the performance over the "no" and "rand" selection strategies. This indicates that our two-step approach for CQs generation is more effective compared to the standard application of LLMs for the task.

For test set evaluation in the context of the CQs-Gen shared task, we select the best run (among the 5 runs with different random seeds) for the three top-performing models (marked with "*" in Table 1), i.e., Llama-3-70B with prompt base (RUN$_1$) and desc$_{(U+\neg U)}$ (RUN$_2$), and Mixtral-8x7B with prompt base (RUN$_3$), all with CQs selection. In Table 2, we report the punctuation scores and the distribution of CQ labels obtained by all runs in the test set. Llama-3-70B in RUN$_1$ and RUN$_2$ performs similarly, while Mixtral-8x7B in RUN$_3$ yields a slightly lower outcome. All models show a consistent drop in performance compared to their average scores on the validation set (Table 1).[5] Results for RUN$_2$, which achieved the best score on the test set (i.e., 0.4510), were manually revised by the CQs-Gen shared task organizers to evaluate the remaining 19 unlabeled questions: of these, 5 were classified as useful, 1 as unhelpful, and 13 as invalid (RUN$_{2\text{FINAL}}$), raising the final punctuation score to 0.50. We should mention that the lack of manual evaluation for the validation set may have impacted the reliability of model selection – a limitation that the shared task organizers aim to resolve through a fully automated evaluation in the future.

Run	U	UH	I	NE	P (labeled)	P (all)
RUN$_1$	45	26	10	21	0.5556	0.4412
RUN$_2$	46	23	14	19	0.5542	0.4510
RUN$_3$	41	20	16	25	0.5325	0.4020
RUN$_{2\text{FINAL}}$	51	24	27	–	0.5000	**0.5000**

Table 2: Distribution of CQ labels and results in the test set. **U**: useful; **UH**: unhelpful; **I**: invalid; **NE**: not_able_to_evaluate; **P (labeled)**: Punctuation score over labeled CQs only, i.e., U/(U+UH+I); **P (all)**: Official punctuation score over all CQs including those labeled as NE, i.e., U/(U+UH+I+NE). For RUN$_2$, we also include the final counts after manual review by the shared task organizers (RUN$_{2\text{FINAL}}$).

5 Qualitative Analysis and Future Work

We conduct a qualitative analysis of the manually reviewed results from RUN$_2$ (i.e., RUN$_{2\text{FINAL}}$) on the test set, proposing a classification of the generated CQs according to the type of argumentative gap they attempt to expose. Results are in Table 3. We recall that, for the 34 interventions in the test set, the output of the run consists of 102 questions (3 per intervention) generated by Llama-3-70B with prompt desc$_{(U+\neg U)}$ and usefulness-based selection, and that the punctuation score achieved by RUN$_{2\text{FINAL}}$ is 0.50 (see Table 2).

For the purpose of the analysis, each CQ is assigned a label based on its underlying argumentative function, structure, and semantics. For instance, questions that request supporting data for a specific claim (e.g., one of the CQs generated for the intervention with identifier "HOLT_122":

[5] Note that the similarity threshold was adjusted from 0.60 to 0.65 by shared task organizers for test set evaluation.

Type	# useful	# non-useful
ALTERNATIVE MEASURES	4	2
ALTERNATIVE EXPLANATION	2	2
ASSUMPTIONS	0	1
BASIS/RATIONALE	2	1
CAUSAL FACTORS	0	1
COMPARISON	1	1
CONSEQUENCES	2	4
CREDIBILITY	0	1
DEFINITION	3	1
EVIDENCE	**15**	**9**
EXAMPLES	1	2
EXPLANATION	1	1
GENERALIZATION	2	2
IMPACT/EFFECT	3	0
IMPLICATION	0	3
OTHER	9	10
POLICY DETAILS	1	6
RESPONSE TO CONCERNS	3	2
ROOT CAUSES	2	2
Total	51	51

Table 3: Distribution of the CQs from $\text{RUN}_{2\text{FINAL}}$ according to the type of argumentative gap they attempt to expose, divided into `useful` and `non-useful` categories. EVIDENCE-related CQs represent the most frequent type across both groups (indicated in bold). The row highlighted in gray groups all CQs for which no clear semantic category can be identified.

"What evidence is there to support the claim that race relations are bad in this country?") are labeled as EVIDENCE. When no clear semantic category emerges, the question is classified in the group OTHER (gray row in Table 3). In some cases, a single CQ includes elements that could be associated with multiple labels (e.g., one of the CQs generated for the intervention with identifier "MP_24": *"What would be the consequences of allowing banks to 'crystallise the debts' and how would it affect the economy?"*, which pertains to both CONSEQUENCES and IMPACT/EFFECT categories); for the sake of consistency and simplicity, in this analysis we assign only the most salient type (in this case, CONSEQUENCES), leaving a more granular categorization for future work. The annotation was carried out manually by a native Italian speaker with advanced proficiency in English and background in data science and Italian studies.

Overall, the qualitative analysis aligns well with findings reported by Calvo Figueras and Agerri (2024). In particular, we observe that the most common type of CQ generated by `Llama-3-70B` asks for EVIDENCE to support a claim: 15 out of 51 `useful` CQs (29.40%) fall into this category. This type is also the most frequent among `unhelpful` and `invalid` questions (9 out of 51; 17.64%), representing 24% of the total questions generated for this run. Among the `useful` CQs, other frequent types include ALTERNATIVE MEASURES (e.g., *"Are there other measures* [...]*"*), though at lower frequencies (4; 7.84%); questions about DEFINITION (e.g., *"How do you define* [...]*"*), IMPACT/EFFECT questions (e.g., *"How does* [something] *affect* [...]*"*) and RESPONSE TO CONCERNS questions (i.e., *"How does* [someone] *address the concerns of* [...]*"*) each occur 3 times (5.88%). Among `non-useful` questions, the second most common type is POLICY DETAILS (e.g., *"What specific policies* [...]*"*, 6; 11.76%), followed by CONSEQUENCES questions (e.g., *"What are the potential consequences of* [...]*"*, 4; 7.84%).

Beyond these initial observations, however, a larger sample size would be needed to identify broader groupings and statistically determine whether any patterns can be directly linked to either `useful` or `non-useful` questions. Indeed, at this stage, rather than being systematically tied to a specific flawed type, `non-useful` questions appear to reflect broader limitations that LLMs face in generating CQs – namely, the introduction of irrelevant concepts, bad reasoning, and insufficient specificity, as discussed by Calvo Figueras and Agerri (2024).

In future work, we aim to manually inspect automatically evaluated CQs to assess the reliability of semantic similarity-based scoring. We also plan to improve our methodology by combining the usefulness-based CQs selection approach with strategies such as chain-of-thought prompting (Wei et al., 2022) and by fine-tuning LLMs – an approach that has shown state-of-the-art performance on several argument mining tasks (Cabessa et al., 2024) – using low-rank adapters (Hu et al., 2022).

6 Conclusion

We present a two-step approach for CQs generation along with a qualitative analysis and insights on the results obtained in the context of the CQsGen shared task hosted at the 12th Workshop on Argument Mining. Our experiments show that usefulness-based CQs selection leads to substantial gains in performance compared to using LLMs only. We hope that our approach may encourage future developments in CQs generation and stimulate research on similar tasks more broadly.

Limitations

The experiments, results, and findings in this paper are based on the dataset of interventions that was provided in the context of the CQs-Gen shared task. Interventions in the dataset are in English, and a large fraction of them concern political topics in the US context. Further research is needed to ensure that results and insights hold for other languages, topics, and contexts that are not represented in the dataset. Due to resource constraints, we employ a limited set of models in our experiments. We are aware that higher results could have been obtained with larger and/or closed-source LLMs. However, our goal was to investigate the effectiveness of a two-step approach for CQs generation using freely available and widely used models.

Acknowledgments

This work has been funded by the European Union's Horizon Europe research and innovation program under grant agreement No. 101070190 (AI4Trust).

References

Jérémie Cabessa, Hugo Hernault, and Umer Mushtaq. 2024. In-context learning and fine-tuning gpt for argument mining. *arXiv preprint arXiv:2406.06699*.

Banca Calvo Figueras and Rodrigo Agerri. 2025. Benchmarking critical questions generation: A challenging reasoning task for large language models. *arXiv preprint arXiv:2505.11341*.

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*, Vienna, Austria. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1128, Miami, Florida, USA. Association for Computational Linguistics.

Robert M Fano. 1961. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29:793–794.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Ashim Gupta and Vivek Srikumar. 2021. X-fact: A new benchmark dataset for multilingual fact checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 675–682, Online. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Barbara Konat, John Lawrence, Joonsuk Park, Katarzyna Budzynska, and Chris Reed. 2016. A corpus of argument networks: Using graph properties to analyse divisive issues. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3899–3906, Portorož, Slovenia. European Language Resources Association (ELRA).

John Lawrence and Chris Reed. 2020. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

John Lawrence, Jacky Visser, and Chris Reed. 2018. BBC Moral Maze: Test your argument. In *Computational Models of Argument - Proceedings of COMMA 2018*, Frontiers in Artificial Intelligence and Applications, pages 465–466, Amsterdam, The Netherlands. IOS Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

E. Musi, M. Aloumpi, E. Carmi, S. Yates, and K. O'Halloran. 2022. Developing fake news immunity: Fallacies as misinformation triggers during the pandemic. *Online Journal of Communication and Media Technologies*, 12(3):e202217.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, and 24 others. 2025. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Alan Ramponi, Camilla Casula, and Stefano Menini. 2024. Variationist: Exploring multifaceted variation and bias in written language data. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 346–354, Bangkok, Thailand. Association for Computational Linguistics.

Alan Ramponi, Agnese Daffara, and Sara Tonelli. 2025. Fine-grained fallacy detection with human label variation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 762–784, Albuquerque, New Mexico. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Yi Song, Michael Heilman, Beata Beigman Klebanov, and Paul Deane. 2014. Applying argumentation schemes for essay scoring. In *Proceedings of the First Workshop on Argumentation Mining*, pages 69–78, Baltimore, Maryland. Association for Computational Linguistics.

Manfred Stede and Jodi Schneider. 2018. *Argumentation Mining*. Morgan & Claypool, San Rafael, CA, USA.

Giovanni Valer, Alan Ramponi, and Sara Tonelli. 2023. When you doubt, abstain: A study of automated fact-checking in Italian under domain shift. In *Proceedings of the 9th Italian Conference on Computational Linguistics (CLiC-it 2023)*, pages 433–440, Venice, Italy. CEUR Workshop Proceedings.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Jacky Visser, Barbara Konat, Rory Duthie, Marcin Koszowy, Katarzyna Budzynska, and Chris Reed. 2020. Argumentation in the 2016 US presidential elections: annotated corpora of television debates and social media reaction. *Language Resources and Evaluation*, 54(1):123–154.

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, Cambridge, United Kingdom.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Appendix

A Categories of Critical Questions

In their GitHub repository,[6] the organizers of the CQs-Gen shared task have provided guidelines outlining the criteria used to label questions as `useful`, `unhelpful`, or `invalid` in the validation and test sets. As a reference, we summarize the descriptions below. Furthermore, in Table 4 we report the number of interventions per corpus in the validation and test sets, and in Table 5 we show the distribution of the three CQs categories in the validation set, broken down by corpus and distinguishing between LLM-generated and theoretical questions.

Useful *"The answer to this question can potentially challenge one of the arguments in the text. One should not take the arguments in the text as valid without having reflected on this question."*

Unhelpful *"The question is valid, but it is unlikely to challenge any of the arguments in the text. This may be in cases where: a) the answer to the question is common sense; b) the answer to the question is a well-known fact that does not generate controversy; c) the question is very complicated to understand and it would be impractical to question the arguments; d) the question is answered in the text itself."*

Invalid *"A question is invalid when the answer to this question cannot serve to invalidate or diminish the acceptability of the arguments of the text. This can be in cases where: a) the question is unrelated to the text; b) the question introduces new concepts not present in the text; c) the question does not challenge any argument defended in the text (for example, when the question challenges the opposite position to the one defended in the text); d) the question is too general and could be applied to any text; e) the question is not critical of the text (e.g. a reading-comprehension question)."*

Set	Corpus				Total
	US2016tv	RRD	MMD	US2016reddit	
Validation	80	72	20	14	186
Test	17	11	6	–	34

Table 4: Number of interventions per corpus in the validation and test sets.

[6] https://github.com/hitz-zentroa/shared-task-critical-questions-generation.

B Further Experimental Details

B.1 Hyper-parameter Values

Generation For the generation of candidate CQs using LLMs, we rely on the default hyper-parameter values as provided in the `transformers` library (Wolf et al., 2020). We only avoid greedy decoding by setting `do_sample = True` and constrain the minimum and maximum number of tokens to generate (using `min_new_tokens` and `max_new_tokens`). Specifically, the maximum number of tokens is set to `128` or `192` when requiring $n = 3$ or $n = 5$ CQs in the output, respectively, whereas the minimum number of tokens is set to `32`. We load `Mixtral-8x7B`, `Qwen-2.5-32B`, and `Llama-3-70B` in `4`-bits due to resource constraints, whereas the remaining models are loaded in `8`-bits. The five random seeds used for the experiments are `0`, `42`, `101`, `31`, and `4321`.

Classification For the model used in the CQs selection stage, we employ default MaChAmp's hyper-parameter values (van der Goot et al., 2021) as detailed in Table 6. We select the best model to be used based on macro F_1 score on a 20% held-out data split. We use 5 epochs of fine-tuning and {32, 64} as search space for the batch size, of which 64 emerged as the best batch size value.

B.2 Prompts

We here provide details on our modular prompts for the zero-shot setting (Appendix B.2.1) as well as prompts adapted for few-shot experiments (Appendix B.2.2). All prompts are built starting from the prompt template presented in Figure 1.

B.2.1 Zero-shot Setting

Starting from a `base` prompt,[7] we experiment with adding information on either the argumentation schemes of the intervention (`schemes`) or detailed description about what CQs are (`desc`).

Prompt `base` A prompt that provides specific task instructions and clear guidance on the expected output. It includes only the free text of the prompt template in Figure 1 and the input `$intervention`.

[7] Our `base` prompt led to higher performance in preliminary experiments compared to the baseline prompt provided by shared task organizers. We here provide their prompt for reference: "*Suggest 3 critical questions that should be raised before accepting the arguments in this text:*`\n\n$intervention\n\n`*Give one question per line. Make the questions simple, and do not give any explanation reagrding why the question is relevant.*"

Corpus	Useful			Unhelpful			Invalid			Total
	LLM	Theory	All	LLM	Theory	All	LLM	Theory	All	
US2016tv	1117	270	1387	166	283	449	116	169	285	2121
RRD	912	110	1022	217	84	301	71	9	80	1403
MMD	224	24	248	56	15	71	49	4	53	372
US2016reddit	122	11	133	60	12	72	33	2	35	240
Overall	**2375**	**415**	**2790**	**499**	**394**	**893**	**269**	**184**	**453**	**4136**

Table 5: Distribution of CQs categories across corpora in the validation set. For each category, we report the number of LLM-generated CQs, the number of theory-derived CQs, and their combined totals (shown in gray).

Prompt template

You are given an argumentative text in the form of an intervention. Your task is to generate $n useful critical questions that should be raised before accepting its arguments. The intervention is as follows:

`$intervention`

`$additional_context`

`$few-shot_examples`

Provide exactly $n useful critical questions, each strictly on a separate line and ending with a question mark. Keep the questions concise and do not add any comments or explanations.

Output:

Figure 1: Template used for modular prompt construction. The `base` prompt consists of the core text (namely, the task instructions, the `$intervention`, and the output requirements). Modular components – i.e., `$additional_context` (either argumentation `schemes` or CQ `descriptions`) and/or `$few-shot_examples` – can be inserted to extend the `base` prompt. The number `$n` of critical questions to generate is a variable parameter, with $n \geq k$.

Hyperparameter	Value
Optimizer	AdamW
β_1, β_2	0.9, 0.99
Dropout	0.2
Epochs	5
Batch size	64
Learning rate	1e-4
LR scheduler	Slanted triangular
Weight decay	0.01
Decay factor	0.38
Cut fraction	0.3

Table 6: Hyper-parameter values employed for fine-tuning the usefulness-based CQs selection classifier.

Prompt `schemes` A prompt where supplementary information on argumentation schemes that occur in the intervention is added to the `base` prompt in place of the `$additional_context` placeholder of the prompt template (Figure 1). The addition is as follows:

> Below are the argumentation schemes associated with the arguments in the intervention:
>
> `$ARG_SCHEMES`

`$ARG_SCHEMES` is a placeholder for the set of argumentation schemes associated with the intervention, automatically extracted from the validation set with duplicates removed. Based on the appendix tables in Calvo Figueras and Agerri (2024), similar scheme names are normalized into a human-readable standard format, following the mapping presented in Table 7.

Normalized name	#	Argumentation scheme(s)
Argument from example	175	Example, ERExample
Practical reasoning	135	PracticalReasoning, ERPracticalReasoning
Argument from cause to effect	55	CauseToEffect
Argument from consequences	38	Consequences, NegativeConsequences, PositiveConsequences
Ad hominem	29	GenericAdHominem, ERAdHominem, Ad hominem
Argument from sign	25	Sign, SignFromOtherEvents
Argument from verbal classification	25	VerbalClassification
Circumstantial ad hominem	22	CircumstantialAdHominem
Argument from fear appeal	14	FearAppeal, DangerAppeal
Argument from analogy	11	Analogy
Argument from expert opinion	10	ExpertOpinion, ERExpertOpinion
Argument from position to know	10	PositionToKnow
Argument from values	10	Values
Argument from popular opinion	8	PopularOpinion
Argument from alternatives	6	Alternatives
Argument from popular practice	6	PopularPractice
Argument from authority	4	ArgumentFromAuthority
Argument from bias	4	Bias
Direct ad hominem	2	DirectAdHominem

Table 7: Normalized names and total number of occurrences for the 28 argumentation schemes in the validation set.

Prompt `desc` A prompt where supplementary information on critical questions is added to the `base` prompt in place of the `$additional_context` placeholder of the prompt template (Figure 1). The addition is one of the following:

- `desc`$_{(FULL)}$, i.e., a detailed bulleted description of `useful` and `non-useful` CQs:

> Useful critical questions may:
> - challenge or clarify a claim by asking for evidence or explanation;
> - examine the consequences of the argument;
> - explore alternatives to the proposed idea;
> - check the generalizability beyond the given case;
> - uncover assumptions that may be implicit.
>
> Non-useful critical questions:
> - ask common sense questions or refer to well-known facts;
> - are overly complex, unclear, or vague;
> - are already answered in the text or are unrelated to the text;
> - introduce new concepts not present in the text;
> - do not challenge the argument or fail to be critical (e.g., reading-comprehension questions).
>
> You must avoid non-useful questions.

- `desc`$_{(U)}$, i.e., an abridged description of `useful` CQs:

> Useful critical questions may ask for evidence, examine consequences, explore alternatives, test generalizability, or uncover hidden assumptions.

- `desc`$_{(\neg U)}$, i.e., an abridged description of `non-useful` CQs:

> You must avoid questions that are vague, overly complex, irrelevant, repetitive, introduce new concepts, restate common knowledge, or fail to critically challenge the arguments.

- `desc`$_{(U+\neg U)}$, i.e., the combination of `desc`$_{(U)}$ and `desc`$_{(\neg U)}$ as a single description:

> Useful critical questions may ask for evidence, examine consequences, explore alternatives, test generalizability, or uncover hidden assumptions.
>
> You must avoid questions that are vague, overly complex, irrelevant, repetitive, introduce new concepts, restate common knowledge, or fail to critically challenge the arguments.

You must avoid questions that are vague, overly complex, irrelevant, repetitive, introduce new concepts, restate common knowledge, or fail to critically challenge the arguments.

In all four versions, the description of `non-useful` questions is based on the guidelines provided by the shared task organizers (Appendix A), while the description of `useful` CQs is derived by scrutinizing examples labeled as `useful` in the validation set. The $\texttt{desc}_{(FULL)}$ version is more comprehensive, whereas $\texttt{desc}_{(U)}$, $\texttt{desc}_{(\neg U)}$, and $\texttt{desc}_{(U+\neg U)}$ are introduced to facilitate the generation of CQs by small-sized models, which, in our preliminary experiments, we observe may struggle with longer input prompts.

B.2.2 Few-shot Settings

One- or three-shot examples can be added to the `base` prompt in place of the `$few-shot_examples` placeholder of the prompt template (Figure 1), with or without `$additional_context` preceding. Below, example interventions and related CQs are referenced by their identifier in the validation set.

One-shot This setting includes a single example intervention and its corresponding output, matching the format expected in the model's final response. We design two variants:

- *all-useful*, where the intervention is followed by three `useful` CQs:

Here is an example of an intervention, followed by three useful critical questions:

`$TRUMP_125_1`

Output:

`$TRUMP_125_1_T__1`
`$TRUMP_125_1_T__14`
`$TRUMP_125_1_T__10`

- *mixed*, where three examples of `non-useful` questions are also provided:

Here is an example of an intervention, followed by three non-useful questions (negative examples) and three useful critical questions (positive examples):

`$TRUMP_125_1`

Non-useful questions:

`$TRUMP_125_1_T__7`
`$TRUMP_125_1_T__25`
`$TRUMP_125_1_T__0`

Output:

`$TRUMP_125_1_T__1`
`$TRUMP_125_1_T__14`
`$TRUMP_125_1_T__10`

Three-shot This setting includes three example interventions and their corresponding outputs. We design two variants:

- *all-useful*, where each intervention is followed by three `useful` CQs:

Here are three examples of interventions, each followed by three useful critical questions:

`$CLINTON_130_1`

Output:

`$CLINTON_130_1_T__8`
`$CLINTON_130_1_T__7`
`$CLINTON_130_1_T__11`

`$TRUMP_125_1`

Output:

`$TRUMP_125_1_T__1`
`$TRUMP_125_1_T__14`
`$TRUMP_125_1_T__10`

`$CLINTON_57`

Output:

`$CLINTON_57_T__3`
`$CLINTON_57_T__12`
`$CLINTON_57_T__10`

- *mixed*, where three examples of `non-useful` questions are also provided for each intervention:

Here are three examples of interventions, each followed by three non-useful questions (negative examples) and three useful critical questions (positive examples):

```
$CLINTON_130_1

Non-useful questions:

$CLINTON_130_1_T__2
$CLINTON_130_1_T__19
$CLINTON_130_1_T__17

Output:

$CLINTON_130_1_T__8
$CLINTON_130_1_T__7
$CLINTON_130_1_T__11

$TRUMP_125_1

Non-useful questions:

$TRUMP_125_1_T__7
$TRUMP_125_1_T__25
$TRUMP_125_1_T__0

Output:

$TRUMP_125_1_T__1
$TRUMP_125_1_T__14
$TRUMP_125_1_T__10

$CLINTON_57

Non-useful questions:

$CLINTON_57_T__5
$CLINTON_57_T__13
$CLINTON_57_T__7

Output:

$CLINTON_57_T__3
$CLINTON_57_T__12
$CLINTON_57_T__10
```

All questions used in the few-shot settings are selected from theory-derived CQs in the validation set; we exclude LLM-generated CQs to prevent over-amplification of synthetic language use. Since theory-derived CQs are instantiated from templates based on argumentation schemes (Calvo Figueras and Agerri, 2024), we ensure that no template is repeated within the sets of `useful` and `non-useful` example questions. In the *mixed* version, however, we include pairs of a `useful` and a `non-useful` question derived from the same template, encouraging the model to focus on the semantic quality of the question rather than relying on their underlying argumentative structure (see Table 8 for examples).

C Further Experimental Results

C.1 Zero-shot Experiments

In Table 9 we report the results on the development set for small-sized LLMs using all the prompt strategies designed for the zero-shot setting.

C.2 Few-shot Experiments

In Table 10 we report the results on the development set for small-sized LLMs in the one-shot setting using the *all-useful* and *mixed* prompt variants. Since the performance in the one-shot setting proved unsatisfactory, due to limited time and resources we did not proceed further with the prompts designed for the three-shot setting. We leave this additional investigation for future work.

C.3 CQs Classifier Experiments

In Table 11 we report the results of the usefulness-based CQs selection models when using different data variants for fine-tuning.

For usefulness-based CQs selection, we also experimented with a strategy based on the most relevant n-grams for the `non-useful` class (i.e., the `unhelpful` and `invalid` merged together). We computed the weighted, positive, and normalized pointwise mutual information (PMI; Fano, 1961) score for each n-gram ($n \in 1, 2, 3$) and class using Variationist (Ramponi et al., 2024), calculated over the `all` data set variant described in Section 4.1. We then selected the top-k ($k = 20$) unigrams, bigrams, and trigrams associated with the `non-useful` class, for a total of 60 keywords. We used the resulting keywords to match candidate CQs to remove from the $n = 5$ generated ones, if any. As a fallback (i.e., when there were no matches), we simply picked the first three CQs. However, this strategy did not consistently improve the performance over the LLMs' application without any selection; therefore, we discarded it.

Theory-derived CQ template	Argumentation scheme	Useful CQ (ID)	Non-useful CQ (ID)
Are there special circumstances pertaining to <subjecta> that undermine its generalisability to other <subjectx> that <featF>?	Argument from Example	TRUMP_125_1_T__1	CLINTON_57_T__7
Did <expertE> really assert that <eventA>?	Argument from Expert Opinion	CLINTON_57_T__10	TRUMP_125_1_T__0
Is <eventA> consistent with known evidence in <domainD>?	Argument from Expert Opinion	CLINTON_57_T__12	TRUMP_125_1_T__25
Are there any events other than <eventB> that would more reliably account for <eventA>?	Argument from Sign	TRUMP_125_1_T__14	TRUMP_125_1_T__7

Table 8: Four pairs of `useful` and `non-useful` questions, derived from the same theoretical template, are included in the *mixed* version of the prompt for one- and few-shot settings. These examples are intended to help the model to discriminate between `useful` and `not-useful` CQs based on semantic content rather than argumentative structure.

Model	Prompt					
	base	schemes	desc$_{(FULL)}$	desc$_{(U)}$	desc$_{(\neg U)}$	desc$_{(U+\neg U)}$
MIXTRAL-8X7B	**0.6758**$_{\pm 0.01}$	0.6262$_{\pm 0.01}$	0.6557$_{\pm 0.01}$	0.6284$_{\pm 0.02}$	0.6594$_{\pm 0.02}$	0.6452$_{\pm 0.01}$
LLAMA-3-8B	**0.6510**$_{\pm 0.01}$	0.5869$_{\pm 0.03}$	0.6076$_{\pm 0.01}$	0.5982$_{\pm 0.02}$	0.6149$_{\pm 0.01}$	0.5905$_{\pm 0.01}$
QWEN-2.5-7B	0.5359$_{\pm 0.02}$	0.4725$_{\pm 0.02}$	**0.5756**$_{\pm 0.01}$	0.5490$_{\pm 0.01}$	0.5476$_{\pm 0.02}$	0.5359$_{\pm 0.02}$

Table 9: Results on the development set for small-sized LLMs using different prompts in a zero-shot setting. We report the average punctuation score with standard deviation across 5 runs with different random seeds.

Model	Shot variant	
	all-useful	mixed
MIXTRAL-8X7B	**0.5847**$_{\pm 0.02}$	0.5719$_{\pm 0.01}$
LLAMA-3-8B	0.1377$_{\pm 0.02}$	**0.4619**$_{\pm 0.02}$

Table 10: Results on the development set for small-sized LLMs in the one-shot setting with prompt `base` and different shot variants (cf. Appendix B.2.2). We report the average punctuation score with standard deviation across 5 runs with different random seeds.

Data variant	Model	
	BERT	RoBERTa
`gold-train`	0.6910	0.6946
`synth-l`	0.7365	0.6916
`synth-m`	0.7392	0.7095
`synth-q`	0.7327	0.6999
`all`	**0.7563**	0.7341

Table 11: Results for different classification models when fine-tuned on different data variants for the sake of usefulness-based CQs selection. We report the macro F_1 score on the `gold-test` split.