

Can formal argumentative reasoning enhance LLMs performances?

Federico Castagna^a, Isabel Sassoon^a and Simon Parsons^b

^aBrunel University London

^bUniversity of Lincoln

Abstract. Recent years witnessed significant performance advancements in deep-learning-driven natural language models, with a strong focus on the development and release of Large Language Models (LLMs). These improvements resulted in better quality AI-generated output but rely on resource-expensive training and upgrading of models. Although different studies have proposed a range of techniques to enhance LLMs without retraining, none have considered computational argumentation as an option. This is a missed opportunity since computational argumentation is an intuitive mechanism that formally captures agents' interactions and the information conflict that may arise during such interplays, and so it seems well-suited for boosting the reasoning and conversational abilities of LLMs in a seamless manner. In this paper, we present a pipeline (*MQArgEng*) and preliminary study to evaluate the effect of introducing computational argumentation semantics on the performance of LLMs. Our experiment's goal was to provide a proof-of-concept and a feasibility analysis in order to foster (or deter) future research towards a fully-fledged argumentation engine plugin for LLMs. Exploratory results using the MT-Bench indicate that *MQArgEng* provides a moderate performance gain in most of the examined topical categories and, as such, show promise and warrant further research.

1 Introduction

Since the introduction of the Transformer technology [34] in 2017, the field of artificial intelligence has significantly advanced by moving towards a paradigm of 'pre-training' and 'fine-tuning' development [40] ultimately leading to the establishment of the so-called Large Language Models (LLMs). Indeed, as their name suggests, LLMs are precisely scaled-up versions (from the architecture size or data perspective) of pre-trained models. This increase in dimension entails interesting and unforeseen consequences¹ impacting the models' capabilities, such as improved arithmetic, multi-task understanding, and enhanced multi-lingual operations [36]. There is also research attesting how Theory of Mind (ToM), i.e. the (human) aptitude to impute mental state to others, may have spontaneously occurred in LLMs as a byproduct of their training [22]. These models thus seem to be endowed with a rich variety of skills that position them far above simple statistical tools which are proficient in language generation. Nonetheless, different scholars argue that LLMs still lack reasoning skills, logical thinking and writing competencies [24, 3, 16, 30]. Indeed, according to some recent studies [18], LLMs

are essentially unable to capture the role of language beyond statistics (and further scaling them up would not provide any adequate solution to this). As an example, we could consider the understanding these models have about causality: whilst they retain correlations from their training data, these do not always reflect reality. Additionally, LLMs can propose likely assertions but not definitive conclusions, which may also change in different instances [18].

Computational argumentation has become increasingly central as a core study within Artificial Intelligence [5] given its promising paradigm of modelling reasoning in the presence of conflict and uncertainty. Indeed, the main strength of this approach lies within the natural use of arguments as a means to formalise non-monotonic reasoning, showing how humans handle inconsistent information dialectically. In a nutshell, the idea is that correct reasoning concerns handling only statements whose stance and embedded data can be defended against any challenges moved by counterarguments.

Proposals of integration between computational argumentation and LLMs have already been outlined in [9, 8], confirming the suitability of such a combination. Building on these initial ideas, in this paper, we developed a simple pipeline (*MQArgEng*) to enable testing of the effectiveness of computational argumentation semantics in enhancing LLMs performances. Our experiment goal was to provide a proof-of-concept and a feasibility analysis in order to foster (or deter) future research towards a fully-fledged argumentation engine plugin for LLMs. Preliminary results suggest in favour of the former, underscoring also how further studies can potentially yield greater benefits.

The research original contributions presented herein are twofold and focus on the questions: (1) *Is it feasible to integrate computational argumentation within the Large Language Models workflow?* (2) *Does this yield enhancements in their performance?* *MQArgEng*, the novel LLM and argumentation pipeline introduced here, aims to address both issues. The paper is structured as follows. In Section 2, we describe the preliminary notions that ground the subsequent work. Section 3 introduces the approaches and tools we leveraged to structure the pipeline presented in Section 4, whose output is assessed via the MT-Bench and recorded in Section 5. These findings are then discussed in Section 6, whereas Sections 7 and 8 review related research and outline potential future directions before concluding in Section 9.

¹ Notice, however, that emergent abilities constitute a controversial topic and some researchers even argue against their existence [29].

2 Background

Before delving into the envisaged pipeline, we will briefly delineate the formalism characterising the engine responsible for the LLMs ‘reasoning augmentation’, i.e., computational argumentation. Afterwards, we will detail the open-source LLM we harnessed in our pipeline and the benchmark chosen for the final evaluation.

2.1 Computational Argumentation

Human interactions are mostly dialectical in nature and revolve around exchanges of arguments and the dialogues that unfold from such interplays. Arguments convey information and, throughout a dispute, they may conflict with other statements. This means that challenging or supporting one’s viewpoint often involves argumentative reasoning. This paramount role is also underpinned by scholars who claim that the function of reasoning is indeed argumentative: “Reasoning has evolved and persisted mainly because it makes human communication more effective and advantageous” [25]. Drawing from dialectical resolutions of inconsistent information as they occur in everyday interactions, computational argumentation provides an intuitive approach to formally capture arguments and their semantics in order to approximate human reasoning. Crucial to this is the notion of argumentation frameworks, where arguments are treated as abstract entities rendered as nodes in a graph, and every directed edge connects the conflicting arguments of the network:

Definition 1 (Abstract AFs [12]). *An argumentation framework (AF) is a pair: $AF = \langle AR, C \rangle$ where AR is a set of arguments, and C is the ‘attack’ binary relation on AR , i.e. $C \subseteq AR \times AR$.*

The idea conveyed by this formalism is that correct reasoning is rendered via the acceptability of a statement: an argument is *justified* (acceptable) only if it is defended against any counterarguments.

Definition 2 (Semantics for Abstract AFs [12]). *Let $AF = \langle AR, C \rangle$, and let $S \subseteq AR$ be a set of arguments. Let also $(X, Y) \in C$ denote the conflict existing between an argument X and its target Y :*

- S is conflict-free iff $\forall X, Y \in S: (X, Y) \notin C$;
- $X \in AR$ is acceptable w.r.t. S iff $\forall Y \in AR$ such that $(Y, X) \in C: \exists Z \in S$ such that $(Z, Y) \in C$;
- A conflict-free extension S is an admissible extension iff $X \in S$ implies X is acceptable w.r.t. S ;
- An admissible extension S is a complete extension iff $\forall X \in AR: X$ is acceptable w.r.t. S implies $X \in S$. The minimal complete extension (with respect to set inclusion) is called the grounded extension, whereas a maximal complete extension (with respect to set inclusion) is called a preferred extension;
- A stable extension S is such that iff $\forall Y \in AR$, if $Y \notin S$, then $\exists X \in S$ such that $(X, Y) \in C$.

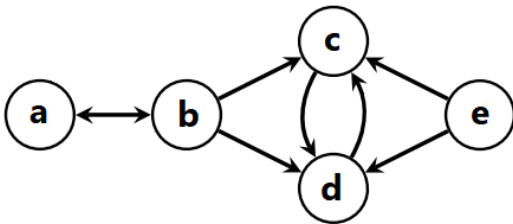


Figure 1. An abstract argumentation framework.

Figure 1 provides a graphical example of an AF, its arguments and the conflicting relations existing between them. Following the semantics described in Definition 2, we can identify the complete (i.e., $\{e\}$, $\{a, e\}$, $\{b, e\}$), grounded (i.e., $\{e\}$), preferred and stable (i.e., $\{a, e\}$, $\{b, e\}$) extensions of the AF.

2.2 Mistral 7B

At the time of the design of this experiment, Mistral 7B [20] was among the most popular and powerful open-source LLMs with seven billion parameters. Engineered by Mistral AI², the model hinges upon a Transformer based architecture [34] characterised by grouped-query attention [1] paired with sliding-window attention [4] and other memory-saving features (i.e., rolling buffer cache, pre-filling and chunking [20]). Such an LLM has proved to outperform Llama 2 (7B and 13B [33]) on a variety of benchmarks and Llama 34B [32] in maths and coding tasks. Given its 8k token context length and its viable inference computational requirement (which makes it accessible also from a consumer-grade laptop), this model seemed a good candidate for our study. However, we needed to run the model in a conversational setting, thus we resorted to the fine-tuned version hosted on the Hugging Face repository³. This version, i.e., *Mistral-7B-Instruct-v0.2*, improves over the base model by increasing the context length up to 32k tokens (without employing sliding-window attention). That being said, discussing an LLM performance is meaningful only when compared with its competitors on specifically designed benchmarks, such as the MT-Bench.

2.3 MT-Bench

The crucial role that LLMs are starting to play in our daily lives has rendered the evaluation aspects as pivotal as the training of models, tuning parameters, or development of new heuristics to increase performance. A variety of benchmarks have thus been created to examine LLMs from different perspectives [10]. MT-Bench is a multi-turn benchmark that presents 80 challenging queries, divided into two sub-questions each, covering 8 different categories: writing, role-playing, reasoning, math, coding, extraction, stem and humanities [42]. The idea behind the benchmark is to appraise the conversational capabilities of the tested LLM on a broad range of topics using another LLM as a judge. The evaluator can then focus on one of the following assessment methods (or combine them): *pairwise comparison*, *single answer grading* and *reference-guided grading*. The first approach determines the best among two concurrent models according to their replies to the specific MT-Bench questions. The second assessment involves testing a single model’s responses to the aforementioned query by scoring its output in a 1-10 range. On the other hand, reference-guided grading may be employed by both of these two previous approaches when a sample answer would help the evaluation. We decided to appraise our pipeline against the MT-Bench due to the wide array of subjects it accounts for: even in the case that the addition of the argumentation engine plugin will not outscore the base model across all the benchmarks, we would still be able to single out the categories in which it performs better than its contender. Another equally important consideration is that MT-Bench ensures a cheaper and quicker valuation compared to human assessors.

² <https://mistral.ai/>.

³ <https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2>

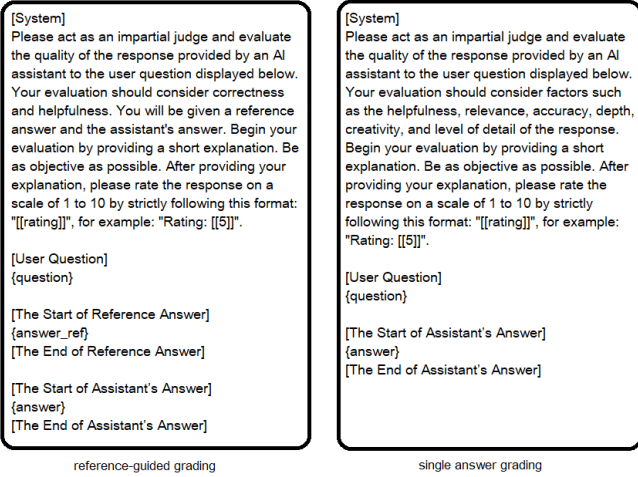


Figure 2. Examples of MT-Bench prompting templates for single question reference guided (left) and standard (right).

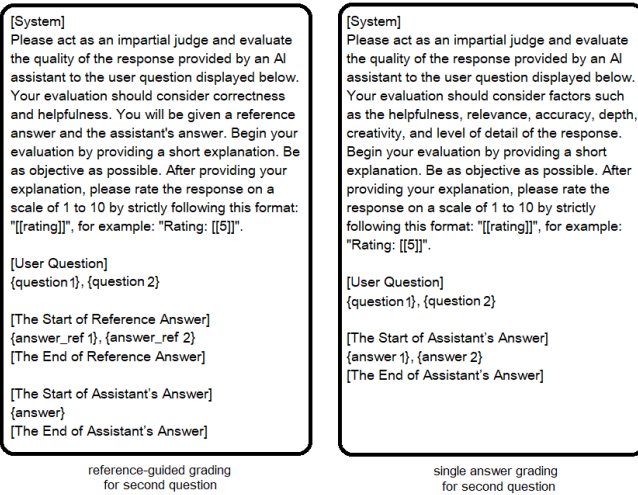


Figure 3. Examples of MT-Bench prompting templates for multi questions reference guided (left) and standard (right).

3 Methodology

The pipeline we devised consists of multiple components, the core of which revolves around a plugin. The plugin we engineer can be thought of as an argumentation engine, i.e., a piece of software that leverages the power of computational argumentation to probe the responses of the underlying LLM in order to select the ones that contain acceptable information. The APSARTIX Solver is the tool responsible for identifying arguments and semantics. Additionally, to make the overall process more accessible and less resource-intensive, we quantized the underlying model. Finally, we designed the final output of the system to be generated from a prompt including a Zero-shot-Chain-of-Thought for a more precise outcome. In the following sections, we briefly outline the mentioned methodologies.

3.1 Quantization

In its essence, quantization is a computational and memory cost-saving approach. Initially described in [19], it is based on the idea of representing the model parameters (i.e., weights and activations)

in low-precision data types, such as `float16` or `int8` rather than `float32`. This procedure usually ensures a substantial reduction in memory storage, energy consumption and yields faster inference [15], but it is a trade-off with a (small) loss in accuracy as well. In our experiment, we resort to a quantized version of Mistral-7B-Instruct-v0.2, henceforth denoted as *MQInstruct*. In particular, we loaded the pre-trained model weights in 4-bit precision to decrease memory footprint by (approximately) 4x times while also enabling nested quantization to preserve further 0.4 bits/parameter. Additionally, to speed up computation, we changed the data type from the default `float32` value to `bfloat16`.

3.2 ASPARTIX Solver

The primary goal underpinning computational argumentation theories is to enable the resolution of conflicting knowledge by identifying the most appropriate (i.e. justified) pieces of information. Reinforcing this is the fact that decision-making processes can be encoded as problems whose solutions are rendered by the calculation and evaluation of AFs: an argumentation solver is essentially a reasoning tool driven by the same logic. Such an argumentative decision-making apparatus can be a useful addition to any software application concerning defeasible reasoning, as advocated by the comprehensive study of Bryant and Krause [7]. In our pipeline, we leverage the ASPARTIX tool (first introduced in [14]) as a driving component of the argumentation engine plugin. Its inner workings, summarized in Figure 4, consist of an Answer-Set-Programming (ASP) solver whose input comprises a batch of ASP-Encodings prescribing the specific computation to be applied on the input AF to achieve the required output.

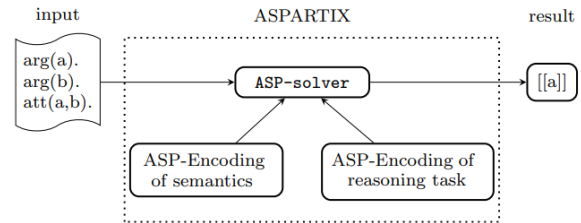


Figure 4. Overview of the ASPARTIX workflow [13].

3.3 CoT

In order to enhance LLMs' logical thinking without resorting to unceasing updates or retraining of the models, a number of strategies were introduced by the AI community. Chain of Thought (CoT), probably the most influential of such propositions, consists of a prompting technique that details a series of intermediate reasoning steps to achieve better performance in arithmetic, symbolic and commonsense inferences [37]. Zero-shot-CoT is a simplified version of CoT that is task-agnostic and does not require few-shot examples [21]. To work properly, this strategy necessitates a double prompt comprising: extracting the rationale and extracting the actual answer based on such an uncovered rationale. The first prompt of this approach, rendered by the standard template "Let's think step by step", will also be a component of the final input conveyed into our pipeline.

4 MQArgEng

MQArgEng is the pipeline that we devised by leveraging *MQInstruct* as the underlying model and an argumentation engine as a plugin

software in charge of guiding the overall workflow. The pipeline is naive in the sense that we developed it as a simple proof-of-concept to test our hypothesis: can formal argumentative reasoning enhance LLMs performances? Figure 5 depicts the proposed pipeline’s high-level operations, which can be delineated as follows:

1. **User prompt.** This component plainly refers to any input provided by the user. The same prompt is fed both to the underlying LLM within the plugin and the model that would provide the final system output.
2. **Mistral 7B Instruct.** It indicates the pivotal LLM that drives the argumentation engine and leads to the output reply. In our experiment, we made use of MQInstruct (i.e., the quantized version of Mistral-7B-Instruct-v0.2).
3. **Argument generation.** During this step, the user input is rephrased in order to request that the underlying model produces three short replies to the user prompt and lists three supporting arguments for each generated answer. All the elicited responses will thus compose the set of arguments AR .
4. **Conflict detection.** At this stage, MQInstruct will be requested to analyse each argument included in AR to record any inconsistency existing among the information they convey. The outcome of this process will comprise the set of attack relations C .
5. **Argumentation framework.** This element overtly represents the formation of an AF from the previously devised arguments (AR) and their relations (C).
6. **ASPARTIX.** So far, the work of the argumentation engine has produced an AF. Now, the ASPARTIX solver has all the required ingredients to compute the *grounded* extension (or *preferred* in case the grounded extension is empty) and its members.
7. **Output reply.** The final outcome of the system results from an additional prompt to the LLM with the same initial user input, augmented by the summarized information embedded in the computed acceptable (grounded/preferred) arguments by ASPARTIX. This additional data would guide the model reasoning to achieve a more effective response.

Being quite minimal, the pipeline (MQArgEng) does not provide any kind of fine-grained parsing of arguments throughout its procedures. This usually entails that ‘oddly shaped’ arguments such as poetical verses, lyrics and similar, mathematical formulae and lines of code, were not properly accounted for. Nonetheless, given the simplicity in recognizing the latter (mostly due to their indentation), we manage to provide heuristics that approximately handle arguments containing lines of code.

To appraise MQArgEng, we conduct an extensive assessment of its and MQInstruct (i.e., the baseline) replies against the MT-Bench.

5 Evaluation

As is customary for the evaluation of the replies of MT-Bench [42], we leveraged GPT-4 [28]⁴ as a judge and required it to score the LLMs responses from both MQInstruct and MQArgEng in a range of 1 to 10 according to the *single answer grading* approach. When dealing with questions where reference answers were given⁵ (this

⁴ We made use of the version of GPT-4 available in <https://chat.openai.com/> on January 2024.

⁵ We closely followed the MT-Bench prompts as recorded on HuggingFace. There was only one exception (i.e., question id 103) in the benchmark where we resorted to adding a reference answer since (judge) GPT-4 was failing to recognize that the provided candidates’ responses should account for the previous question’s logical riddle.

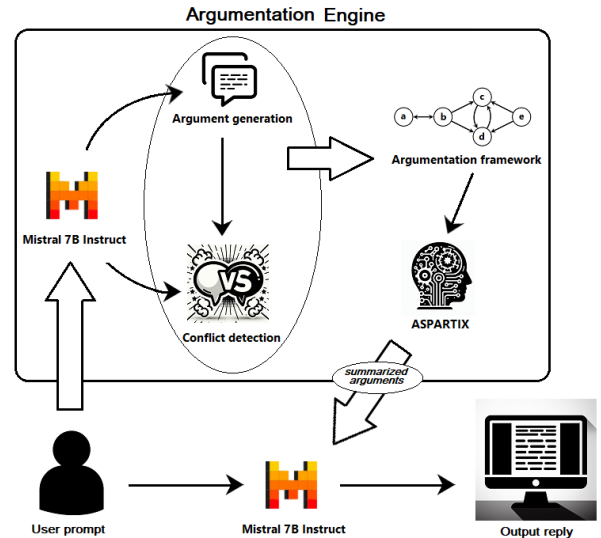


Figure 5. MQArgEng: Naive pipeline employing the argumentation engine.

mostly involves math, coding and reasoning topics) we opted for the *reference-guided grading*. In particular, we adopted the prompting templates depicted in Figure 2 and Figure 3. Notice that we also acted as a human ‘second marker’ to check GPT-4 assessment. That is because it is sometimes the case where the primary judge cannot properly understand the question, erroneously considers wrong right replies (or vice versa), or inconsistently grade them. As such, we required GPT-4 to generate multiple evaluations, and (as a rule of thumb) we decided the official grade was the numerical score that was the mode and appeared at least three times. This has been also made necessary by the fact that we could not harness the automated MT-Bench evaluation proposed on FastChat, given that our pipeline is not an LLM per se.

Table 1. Evaluation scores over the MT-Bench.

Categories	MQInstruct	MQArgEng	Δ
Writing	8.15	8.05	↓ -0.10
Roleplaying	6.80	6.75	↓ -0.05
Reasoning	4.05	4.25	↑ +0.20
Math	2.80	2.80	=
Coding	5.30	5.50	↑ +0.20
Extraction	5.55	5.70	↑ +0.15
STEM	7.35	7.60	↑ +0.25
Humanities	7.75	8.10	↑ +0.35
Average	5.96	6.09	↑ +2, 18%

6 Discussion

Before diving into the analysis, we should disclose the curious fact that the total score of MQInstruct does not correspond to the one presented on the current leaderboard⁶. This ensues for a series of reasons. For example, the model we leveraged was a quantized version of Mistral-7b-Instruct v.02, which is supposedly less accurate. Furthermore, we added Zero-shot-CoT to the prompt that, although

⁶ <https://chat.lmsys.org/?leaderboard>.

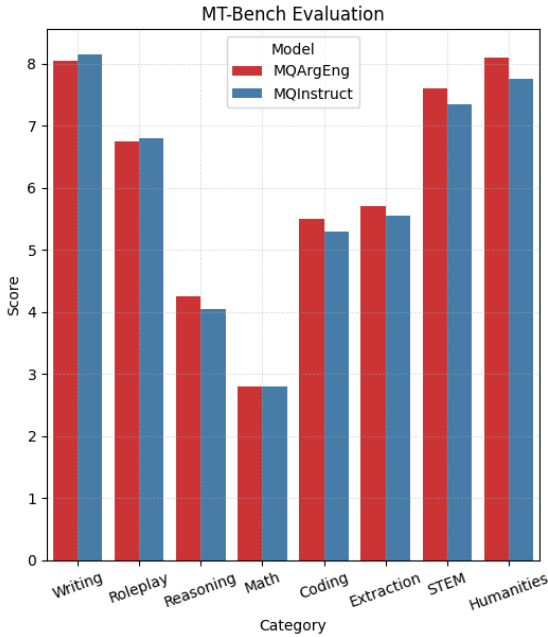


Figure 6. Graphical rendering of the evaluation scores from Table 1.

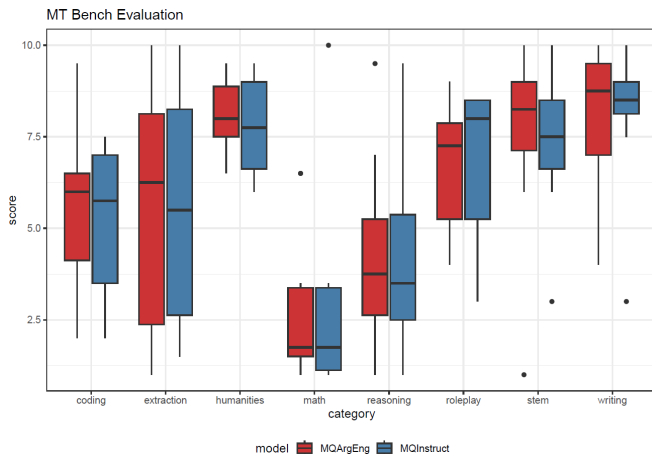


Figure 7. Data distribution of the evaluation scores from Table 1.

helpful in the reasoning and maths questions, it is sometimes a disadvantage for writing and roleplaying. Also, consider that our evaluation slightly differed from the one presented in [42] since we introduced a human moderator in the process, and we prompted GPT-4 multiple times for the same question. That being said, our findings are not affected by the difference in score with the leaderboard. We conducted the experiment under the same conditions with both the baseline model (MQInstruct) and the proposed pipeline, and the only metrics we were interested in were the recorded performances and their delta, as shown in Table 1. Figure 6 illustrates how MQArgEng outscored its competitor in 5/8 categories (reasoning, coding, extraction, stem, humanities) whilst tied in the math category of the MT-Bench. These results are further corroborated by Figure 7 where the interquartile range of the data distribution reflects the same improvements of MQArgEng over the baseline, accounting also for the same number of outliers. Overall, given the exploratory stage of the attempted experiment, the scores achieved against the appraised

benchmark are quite positive and warrant future research. In particular, the overall +2, 18% is well in scale with the usual improvement granted by LLM research. For instance, the recently released Llama 3 70B-Instruct increased its performance by about 7% compared with Claude 3-Sonnet on the same benchmarks (notice this improvement is the result of a full cycle of pre-training and fine-tuning on over 15T of data, thus definitely more resource intensive than our naive plugin approach [26]). We can summarize the pros and cons of our findings as follows.

6.1 Advantages

- ◇ As developed herein, an argumentation engine plugin proves to be scalable to any other model (potentially including Small Language Models [27] or even future releases of new LLMs). Granted the generation of multiple different responses and the capability of comparing and detecting inconsistency among the information they embed, there are indeed no restrictions on the underlying model to be leveraged by the pipeline.
- ◇ In addition, there are also no constraints in employing different LLM architectures: designed as an external plugin, an argumentation engine operates regardless of the model structure whether it is a Transformer, the recently introduced Mamba [17] or any other.
- ◇ Even if the argumentation engine does not outscore the baseline across the board (i.e., writing and roleplaying), it can be extremely useful when deployed for those categories. For example, the plugin could be triggered only when in the presence of such subjects, thus enhancing the LLM output quality, and remain inactive otherwise.
- ◇ The final reply after the injection of the information summarized by the argumentation engine leverages also zero-shot-CoT. It is thus safe to assume that more advanced prompting techniques could be employed, possibly leading to higher performances.

6.2 Limitations

- ◇ The parsing of the arguments for further processing along the pipeline was quite straightforward and did not account for nuances in the argument syntax. This mostly affected when arguments were written as lyrics/poems and maths formulae.
- ◇ The overall output heavily relies on the quality of the argument generated by the underlying model, which can slightly vary from one iteration to the other.
- ◇ Similarly, from the underlying model and the available hardware depend the inference speed. For example, making use of a very large LLM (e.g., the open source Grok-1 [38]) with inadequate GPU support could suffice for generating output, but the latency between each generation will render the plugin highly ineffective to run.
- ◇ The quality of the replies provided by the argumentation engine within MQArgEng varies between the first and the second sub-questions of the queries of the MT-bench. Indeed, the second sub-questions may sometimes result in lower-score responses. The explanation of this behaviour is related to the initial argument generation from the underlying LLM that may happen to focus on the first half of the prompted queries. This, in turn, drives the engine into reasoning only on one of the sub-questions ending in forcing also the final output of the pipeline to concentrate only on a partial answer.

The poor performance of MQArgEngine on maths-related questions is thus not surprising given the aforementioned limitations (given

also that, in general, LLMs have proven to struggle with such topics). The rough parsing of the generated arguments often failed to recognize the math formulae belonging to the unfolding of an equation as a unique element, resulting in arguments consisting of incomplete formulae. Similar consideration holds for the writing and roleplay category, where poetical verses were not properly accounted for. Nonetheless, there were much fewer requests for handling limericks or similar, hence the reason for such a high score in writing subjects compared to maths. In addition, we argue that both writing and roleplay are affected by a somewhat reduced creativity due to the summarized argument injection combined with the requested step-by-step output generation (zero-shot-CoT).

7 Related Works

As previously anticipated, in an attempt to provide effective solutions to LLMs reasoning shortcomings, several training-free approaches have been proposed in the literature. For example, Self-Consistency Chain, Tree or Graph of Thoughts, respectively, CoT-SC, ToT, and GoT. The limitations of the previously mentioned CoT strategy mostly concern the absence of a procedure to plan or analyse multiple reasoning paths before generating the output, and this is exactly the enhancement yielded by CoT-SC, ToT and GoT. Indeed, Self-Consistency Chain of Thought starts from standard CoT promptings and samples a set of candidate outputs before selecting the answer that is the most consistent among the generated reasoning path [35]. Tree of Thoughts frames each problem as a search over a tree, where each node is a partial solution [39]. Graph of Thoughts, instead, envisages the information generated by an LLM as an arbitrary graph, distilling dependencies between such information units and enhancing reasoning by focusing on the core elements of the network [6]. Against these three options, we argue that endowing LLMs’ pipelines with a reasoning engine driven by computational argumentation may provide a more intuitive (e.g. grounded on dialectical logic, unlike CoT-SC), cheaper (e.g. less resource-expensive to be implemented, unlike ToT and GoT) and comprehensive alternative (e.g., effective on a variety of topics, unlike the limited use cases showed for ToT and GoT). Argumentative reasoning is particularly suited for models that parse, work and generate natural language. Recall that AFs are graphs whose edges represent paths determining the status of each node. Then, semantically computing an argumentation framework allows planning the most appropriate sequence of ‘thoughts’ (arguments) to achieve the desired result. Such sequences account for divergent information, thus also mimicking and (potentially) outperforming the CCoT (Contrastive Chain of Thought) prompting technique, which generally handles only one contrastive sample at a time [11]. Unrelated to Chain of Thought, another approach that elicits information from an external engine (i.e., MuJoCo [31]) can be found in Google’s Mind’s Eye [23]. Similarly to our pipeline, it adds the output of MuJoCo to the LLM’s input and proves how this increases the model reasoning capabilities under the UTOPIA benchmark. Regardless, this procedure requires also the presence of a text-to-code converter to encode data for the engine whose proficiency strictly revolves around physics knowledge. On the other hand, our approach presents a simplified pipeline (harnessing only one LLM), and it is driven by computational argumentation, which ensures augmented capabilities across a broader range of topics (as testified by the MT-Bench evaluation). One last interesting technique to mention is the *step-back prompting* introduced in [41]. Indeed, the main idea concerns prompting an LLM to take a step back from the main problem to ask a question about a high-level concept.

The answer will then guide the model reasoning about the solution to the original issue. Despite the positive outcome, this method requires the generation of step-back questions which are unique for each task in order to retrieve the most relevant facts. On the contrary, our plugin is more flexible and presents a one-size-fits-all design regardless of the circumstances.

8 Future Directions

Given the positive outcome of the preliminary study reported herein, we envisage a number of possible research extensions aimed at establishing the plugin’s usefulness and increasing its efficiency:

- We believe that employing a stronger underlying LLM (e.g., GPT-4, Claude 3 [2]) would improve the overall pipeline output resulting in a higher score on the MT-Bench. That is mostly due to steps a) the generation of arguments and b) the comparisons of arguments to detect conflicting information, both of which solely rely on the capabilities of the leveraged underlying model. Alternatively, another option would be harnessing a separate model specialized to (or fine-tuned for) diversify argument generation and comparison, thus taking charge of steps a) and b).
- Other lines of improvement could originate from the adoption of more advanced prompting techniques to be employed in the final input received by the model or by resorting to better LLMs that would act as judges in the evaluation. This may occur by leveraging the latest cutting-edge model or a fine-tuned version specialized in such an assessment task.
- We argue that better performances could be achieved by also combining together the generated arguments and their supports into single, more complete arguments. Similarly, we posit that a more accurate fine-grained parsing of the ‘oddly shaped’ arguments could improve the overall performance of the pipeline.

9 Conclusion

Is it feasible to integrate computational argumentation within the Large Language Models workflow? Does this yield enhancements in performance? Motivated by the present shortcomings faced by LLMs with reasoning tasks, in this paper, we have addressed both questions with a positive outcome. In order to do so, we proposed *MQArgEng*, a pipeline that incorporates a computational argumentation engine to guide an LLM output process. We also evaluated it using an experiment to compare its performance to a standard (*MQInstruct*). The results proved how our proposed engine, a simple plugin tool (characterised by having no constraints in terms of the underlying model or their architecture), suffices to increase the MT-Bench scores against the baseline. Such an improvement concerned most of the examined categories, showing particular strength for questions classed as humanities, stem, reasoning and coding whilst slightly failing to achieve equally good scores in questions from categories such as writing and roleplaying. Multiple research directions can lead this study to further boost the plugin’s effectiveness by, for example, resorting to state-of-the-art underlying models or employing a more fine-grained parsing of the arguments involved. Overall, we can deem this preliminary experiment as successful and showing promise and warrant further work.

References

- [1] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

- [2] Anthropic. The Claude 3 model family: Opus, Sonnet, Haiku. 2024.
- [3] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, et al. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*, 2023.
- [4] I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.
- [5] T. J. Bench-Capon and P. E. Dunne. Argumentation in artificial intelligence. *Artificial intelligence*, 171(10-15):619–641, 2007.
- [6] M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690, 2024.
- [7] D. Bryant and P. Krause. A review of current defeasible reasoning implementations. *The Knowledge Engineering Review*, 23(3):227–260, 2008.
- [8] F. Castagna, N. Kökciyan, I. Sassoon, S. Parsons, and E. Sklar. Computational argumentation-based Chatbots: a survey, 2024.
- [9] F. Castagna, P. McBurney, and S. Parsons. Explanation–Question–Response dialogue: An argumentative tool for explainable AI. *Argument & Computation*, (Preprint):1–23, 2024.
- [10] Y. Chang, X. Wang, J. Wang, Y. Wu, K. Zhu, H. Chen, L. Yang, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang, and X. Xie. A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109*, 2023.
- [11] Y. K. Chia, G. Chen, L. A. Tuan, S. Poria, and L. Bing. Contrastive chain-of-thought prompting. *arXiv preprint arXiv:2311.09277*, 2023.
- [12] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.
- [13] W. Dvořák, A. Rapberger, J. P. Wallner, and S. Woltran. Aspartix-v19-an answer-set programming based system for abstract argumentation. In *International Symposium on Foundations of Information and Knowledge Systems*, pages 79–89. Springer, 2020.
- [14] U. Egly, S. A. Gaggl, and S. Woltran. Aspartix: Implementing argumentation frameworks using answer-set programming. In *International Conference on Logic Programming*, pages 734–738. Springer, 2008.
- [15] H. Face. Quantization. (https://huggingface.co/docs/optimum/en/concept_guides/quantization, (last accessed 19/04/2024).
- [16] S. Frieder, L. Pinchetti, R.-R. Griffiths, T. Salvatori, T. Lukasiewicz, P. C. Petersen, A. Chevalier, and J. Berner. Mathematical capabilities of chatgpt. *arXiv preprint arXiv:2301.13867*, 2023.
- [17] A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.
- [18] K. Hammond and D. Leake. Large language models need symbolic ai. 2023.
- [19] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.
- [20] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7B, 2023.
- [21] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.
- [22] M. Kosinski. Theory of mind may have spontaneously emerged in large language models. *arXiv preprint arXiv:2302.02083*, 2023.
- [23] R. Liu, J. Wei, S. S. Gu, T.-Y. Wu, S. Vosoughi, C. Cui, D. Zhou, and A. M. Dai. Mind’s eye: Grounded language model reasoning through simulation. *arXiv preprint arXiv:2210.05359*, 2022.
- [24] K. Mahowald, A. A. Ivanova, I. A. Blank, N. Kanwisher, J. B. Tenenbaum, and E. Fedorenko. Dissociating language and thought in large language models: a cognitive perspective. *arXiv preprint arXiv:2301.06627*, 2023.
- [25] H. Mercier and D. Sperber. Why do humans reason? Arguments for an argumentative theory. In *Behavioral and brain sciences*, volume 34, pages 57–74. Cambridge University Press, 2011.
- [26] Meta. Introducing Meta Llama 3: The most capable openly available LLM to date. 2024. (<https://ai.meta.com/blog/meta-llama-3/>, (last accessed 19/04/2024).
- [27] K. Mok. The Rise of Small Language Models. 2024. (<https://thenewstack.io/the-rise-of-small-language-models/> (last accessed 25/04/2024).
- [28] OpenAI. GPT-4 technical report, 2023.
- [29] R. Schaeffer, B. Miranda, and S. Koyejo. Are emergent abilities of large language models a mirage? *arXiv preprint arXiv:2304.15004*, 2023.
- [30] H. H. Thorp. ChatGPT is fun, but not an author. *Science*, 379(6630):313–313, 2023. doi: 10.1126/science.adg7879. URL <https://www.science.org/doi/abs/10.1126/science.adg7879>.
- [31] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.
- [32] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [33] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [35] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models, 2023.
- [36] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.
- [37] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.
- [38] xAI. Open Release of Grok-1. 2024. (<https://x.ai/blog/grok-os>, (last accessed 21/04/2024).
- [39] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.
- [40] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.
- [41] H. S. Zheng, S. Mishra, X. Chen, H.-T. Cheng, E. H. Chi, Q. V. Le, and D. Zhou. Take a step back: Evoking reasoning via abstraction in large language models. *arXiv preprint arXiv:2310.06117*, 2023.
- [42] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena, 2023.