Intermediate Languages Matter: Formal Languages and LLMs affect Neurosymbolic Reasoning

Alexander Beiser[1], David Penz[2] and Nysret Musliu[3]

[1]TU Wien, Vienna, Austria
[2]Johannes Kepler University Linz, Linz, Austria
[3]TU Wien, Vienna, Austria

Abstract

Large language models (LLMs) achieve astonishing results on a wide range of tasks. However, their formal reasoning ability still lags behind. A promising approach is Neurosymbolic LLM reasoning. It works by using LLMs as translators from natural to formal languages and symbolic solvers for deriving correct results. Still, the contributing factors to the success of Neurosymbolic LLM reasoning remain unclear. This paper demonstrates that one previously overlooked factor is the choice of the formal language. We introduce the intermediate language challenge: selecting a suitable formal language for neurosymbolic reasoning. By comparing four formal languages across three datasets and seven LLMs, we show that the choice of formal language affects both syntactic and semantic reasoning capabilities. We also discuss the varying effects across different LLMs.

Keywords

logical reasoning, neurosymbolic approaches, LLM/AI agents, prompting, few-shot learning

1. Introduction

Logical reasoning tasks pose a challenge to Large Language Models (LLMs), as they struggle to reason abstractly and correctly [1, 2, 3]. This leads to their sometimes spectacular failures, like deriving that birds have four legs [4]. One attempt to improve the abstract reasoning capability is Chain of Thought (CoT) [5] prompting. With CoT, LLMs are nudged to reason step-by-step. However, LLMs' step-by-step reasoning is generally *non-faithful* - even when all individual reasoning steps are correct on their own, the final conclusion can be false [6].

Neurosymbolic LLM reasoning enables faithful reasoning chains. It works in two steps: the first step translates a natural language-posed logical reasoning problem into a *formal intermediate language*. The translation uses the *in-context-learning* (ICL) capability of LLMs. The second step is to solve the translated problem by a symbolic reasoner. Novel neurosymbolic approaches, such as Logic-LM [7] and LINC [8], report substantial improvements over pure LLM prompting.

However, it remains unclear what the reasons for their reported success are. This comes, as there are a plethora of possible contributing factors, ranging from the LLM training data, over auxiliary systems (such as re-prompting on errors), to the choice of formal language. We investigate the choice of formal language, as it is rarely justified, let alone supported by empirical evidence, leaving its impact on neurosymbolic LLM reasoning largely uncharted.

Contributions. By measuring the impact of different formal languages, we take a first step toward better understanding why neurosymbolic systems obtain state-of-the-art results and how the choice of formal language affects reasoning. The main contributions of this work are:

- We introduce the *intermediate language challenge*: the choice of formal language for neurosymbolic LLM reasoning affects the reasoning performance.
- We conduct an extensive empirical study of four formal languages across three logical reasoning datasets (ProntoQA, ProofWriter, FOLIO) and seven LLMs (8B–671B).

The Second Workshop on Knowledge Graphs and Neurosymbolic AI (KG-NeSy), co-located with SEMANTiCS'25: International Conference on Semantic Systems, September 3–5, 2025, Vienna, Austria

alexander.beiser@tuwien.ac.at (A. Beiser)

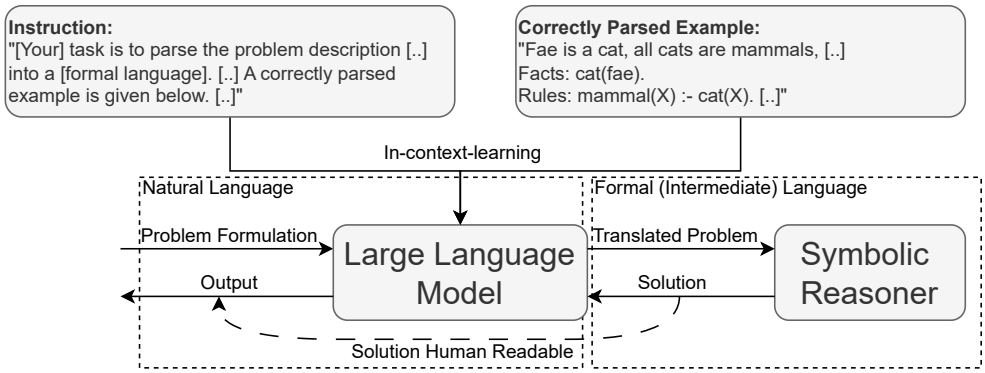

Figure 1: Neurosymbolic LLM reasoning: A problem formulated in natural language is translated by using in-context-learning into a formal language. Subsequently, a symbolic reasoner subsequently computes a solution to the problem, which is followed by the re-translation of the solution.

Our experiments show that the choice of formal language matters: first-order logic outperforms logic programming languages. This paper is the short version of *Intermediate Languages Matter: Formal Choice Drives Neurosymbolic LLM Reasoning* [9]. Here, we focus on the main results of the comparison of different formal languages, and (previously not shown) comparison of LLMs.

Structure. After this introduction we will present the necessary preliminaries and discuss related work (Section 2). We continue to introduce our main hypothesis - the intermediate language problem - that the choice of formal language affects reasoning performance (Section 3), which we follow with our experimental setup (Section 4), and our experimental results (Section 5). We close our paper with our conclusions in Section 6.

2. Preliminaries

We briefly present the necessary background material and definitions for understanding the paper. Recall that the main objective of this study is to compare the reasoning performance of different formal languages on modern LLMs. By taking the perspective of end-users, we treat LLMs as immutable black-box next-token predictor machines. Therefore, we are primarily interested in what effects different prompting strategies have on the reasoning performance and consider the effects of other techniques, such as fine-tuning, as out of scope. Throughout this paper, the terms *syntax* and *semantics* are used in their formal language sense.

2.1. Chain-of-Thought (CoT) prompting

Chain-of-Thought (CoT) prompting is an *in-context-learning* (ICL) technique which improves the reasoning capabilities of LLMs by adding additional information to a prompt [5]. CoT nudges the LLM to mimic a reasoning chain, where we show an example in the next listing.

```
The following example showcases the line of reasoning you have to follow:
---- Question ----
Each cat is a carnivore. Fae is a cat.
True or false: Fae is a carnivore
---- Reasoning ----
Fae is a cat. Each cat is a carnivore. So Fae is a carnivore.
```

2.2. Neurosymbolic LLM Reasoning

Figure 1 depicts the high-level schematics of neurosymbolic LLM reasoning. A natural language-posed problem is translated into its *formal language* representation by using ICL. ICL comprises of an *ICL-instruction* and an *ICL-example*. The instruction describes the general task, while the example showcases how to translate the natural language-posed problem into a formal language. We refer to the formal language of the ICL-example, as the *chosen* formal language.

In a second step, the symbolic reasoner solves the problem by obtaining a solution from the formal representation, which can be either re-translated into natural language or directly used as output. We do not employ backup strategies and use a as close as possible deterministic prompting (temperature 0), as we are interested in the unfiltered affect of the formal language on reasoning performance. We thereby differ from other related approaches like Logic-LM [7], Logic-LM++ [10], and LINC [8].

2.3. Related Work

Improving LLM's reasoning capability was approached by different angles. Model improvements include fine-tuning or pre-training to improve numerical capabilities [11] or syntax recognition of ASP with LLASP [12]. Prompting techniques are widely used, such as CoT prompting, part of the emergent ICL or *few-shot-learning* capability [13]. CoT improves LLMs' performance on reasoning tasks [5]. Recent reasoning-focused LLMs, like DeepSeek-R1 utilize internal CoT [14]. In contrast to these approaches we utilize CoT prompting and neurosymbolic AI. Neurosymbolic AI [15] is a broad field which ranges from differentiable logic [16] over visual question answering [17], to LLMs [7, 8]. For logical reasoning tasks in particular, *Logic-LM* [7] and *LINC* [8] are two proposed neurosymbolic approaches that combine LLMs with symbolic solvers. They translate a natural languages into a formal language - called autoformalization [18, 19]. Logic-of-Thought, which tackles logic-puzzles with a neurosymbolic approach [20], is also related. Although prior work employs an intermediate language, it seldom justifies the choice. We show — empirically — that the selected language decisively shapes reasoning performance.

3. The Intermediate Language Challenge

We proceed to define our intermediate language challenge for neurosymbolic LLM reasoning. We assume to have given a natural language-posed reasoning problem $\mathcal{P}$ and a set of possible formal languages $\mathcal{L}$.

Definition 1. *The intermediate language challenge is the task of choosing a formal language $l \in \mathcal{L}$ for solving $\mathcal{P}$ with a high reasoning accuracy.*

Inherent to the intermediate language challenge is autoformalization [18].

Definition 2. *Let $l \in \mathcal{L}$ be a fixed formal language. Then, autoformalization aims for automatic and correct translation of $\mathcal{P}$ into l.*

While autoformalization is concerned with the correct translation from natural language into a fixed formal language l, the intermediate language challenge is about choosing a suitable formal language $l' \in \mathcal{L}$ s.t. autoformalization can be done effectively. We identify two root causes of the intermediate language problem: (i) Syntax affects LLMs' reasoning performance, and (ii) one logical problem can be translated into multiple formal languages.

Syntax affects LLMs' reasoning performance. Consider the following two logical reasoning problems: (1) "Tommi is a tumpus. Each tumpus is a wumpus. Is Tommi a wumpus?" (2) "Tommi is a cat. Each cat is an animal. Is Tommi an animal?" Recent work suggests that, on average, LLMs perform better for scenarios of type (2) than type (1) [1, 2]. From a semantic perspective, both scenarios require the application of modus ponens. Thus, as the *only* difference lies in the *syntax*, we can conclude that the syntax affects LLMs' reasoning capabilities. Going back to formal languages, observe that the syntax of formal languages differs. Therefore, we conclude that the choice of formal language affects LLMs' reasoning capabilities.

Logical problems can be encoded in different formal languages. Take the logical reasoning problem (2) from the paragraph above. This problem can be encoded in different formal languages, such as logic programming or first-order logic (FOL), while maintaining semantic correctness.

4. Experiment Setup

To show the impact of the intermediate language challenge, we investigate a set of formal languages $\mathcal{L} = \{\text{Pyke}, \text{ASP}, \text{NLTK}, \text{FOL}\}$. We conduct experiments on three different datasets, ProntoQA [1], ProofWriter [21], and FOLIO [22]. Let $\mathcal{D}$ be a given dataset, then each data instance $\mathcal{P} \in \mathcal{D}$ can be

considered a reasoning problem. Each $\mathcal{P}$ is translated into a formal language by the LLM according to a specification (prompting style). We prompt the LLM with a prompting style that adheres to Figure 1 - i.e., we provide an ICL instruction and an ICL example. We use a set of prompting styles, where they differ in the syntax of the ICL example, such as wrapping the example in markdown syntax. Importantly, we enable comparability between formal languages by using the same set of prompting styles for each formal language.

4.1. Formal Languages

We will provide a brief overview of the formal languages $\mathcal{L}$ used for our experiments.

Pyke: The logic programming derivative Pyke [23] expresses rules similar to *if-then* statements. Pyke derives conclusions by forward, or backward chaining algorithms.

ASP: In the non-monotonic logic programming paradigm Answer Set Programming (ASP) [24, 25] a program is written as a set of rules, which is first grounded [26] and then solved [27].

NLTK: The natural language toolkit [28] is a Python library that enables an integration of FOL with Prover9 [29]. We assume familiarity with the semantics of FOL.

FOL: We assume familiarity with the syntax and semantics of FOL. For our experiments, we implemented a *parser* that translates FOL to NLTK formulas, which are then solved by Prover9.

4.2. Datasets

We perform experiments on three datasets. We used one partly hand-crafted ICL-example (training data) per dataset/formal language, which is not part of the test set. Each test set configuration resembles the configuration of Logic-LM.

ProntoQA [1]. The ProntoQA dataset is a generated dataset. We use the fictional character version with a reasoning depth of 5. A random answer has a probability of 50% for getting a correct answer (closed-world assumption - CWA), and a test set with 500 samples is used.

ProofWriter [21]. ProofWriter is a generated dataset. We chose a reasoning depth of 5. A random answer has a probability of about 33% to get a correct answer (open-world assumption - OWA). The test set has 600 samples.

FOLIO [22]. FOLIO is a (partly) expert-written dataset. A random answer is correct with about 33% (OWA). The FOLIO test set has 204 samples. We do not use ASP and Pyke on FOLIO, as FOLIO instances require classical logic concepts which are effectively impossible to encode in standard logic programming.

4.3. Large Language Models

We compare the formal languages on seven LLMs, ranging from 8B to 671B parameters. For all experiments we set the temperature to 0, to obtain a near-deterministic behavior. We restricted the maximum number of new tokens to be 2048 and did not perform any additional modifications to the LLMs. We are primarily interested in how the intermediate language affects small language models (SLMs) with $\approx$ 8B parameters, due to their lower resource consumption. Further, we focus on chat models, as reasoning models build upon them. We used the following LLMs of approximately 8B parameters: *GPT-4o-mini*[1], *Ministral-8B*[2], *Llama-8B*[3]. and *DeepSeek-8B*[4]. To study the effects when using bigger models, we additionally perform experiments on *DeepSeek-32B*[5] ($\approx$ 32B parameters) and *DeepSeek-V3* ($\approx$ 671B parameters) models. To verify the results on state-of-the-art reasoning models we performed benchmarks on *DeepSeek-R1*[6] as well. We prompted *DeepSeek-R1* with both 2048 and 20480 max-output-tokens, due to increased output token generation of the reasoning model.

[1] https://platform.openai.com/docs/models/gpt-4o-mini
[2] https://mistral.ai/news/ministraux
[3] https://openrouter.ai/meta-llama/llama-3.1-8b-instruct
[4] https://openrouter.ai/deepseek/deepseek-r1-distill-llama-8b
[5] https://openrouter.ai/deepseek/deepseek-r1-distill-qwen-32b
[6] https://api-docs.deepseek.com/news/news1226

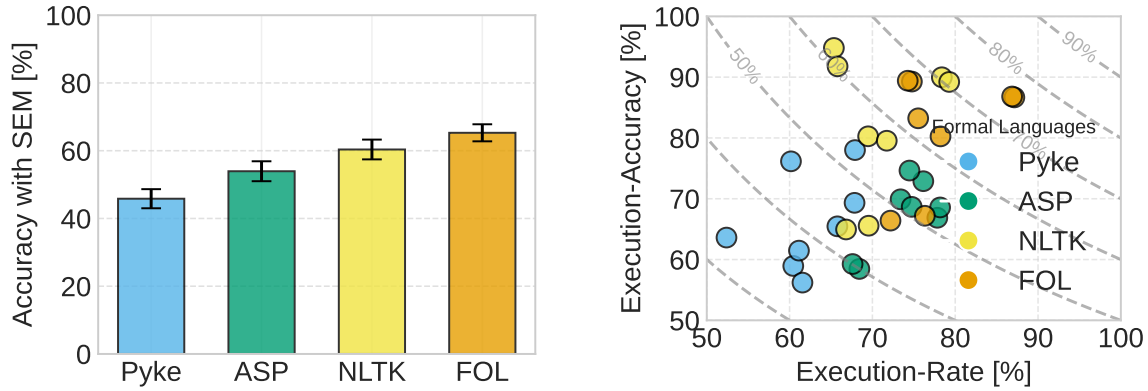

Figure 2: Left: We show the effects of the formal languages, averaged across all prompting styles, LLMs, and the ProntoQA and ProofWriter datasets. Error bars show the SEM $n = 112$. Right: Scatter plots comparing execution-rate to execution-accuracy for the formal languages. A single dot shows an average across prompting styles averaged over ProntoQA and ProofWriter datasets and all LLMs ($n = 14$). Contour lines show overall-accuracy in steps of 10%.

4.4. Baselines

The *chance* is the probability of getting a correct answer by a random draw. Chance is 50% for ProntoQA, as it has a CWA, and 33% for ProofWriter and FOLIO, as they have an OWA. Additionally, we use the following baselines[7].

Std. - refers to standard prompting. The LLM is given a short instruction on the task, the natural language-posed problem, and a short example of how the LLM shall answer the question.

CoT - refers to CoT prompting. It extends standard prompting by nudging the LLM to reason step-by-step by employing CoT.

4.5. Experimental Evaluation

We conduct our experiments on an adapted Logic-LM implementation. Our adaptation includes an ASP symbolic solver based on Clingo [27], a new Pyke implementation, and an adapted NLTK/FOL solver implementation. We conduct experiments for 4 formal languages and 8 prompting styles, leading to 32 total experiments for ProntoQA and ProofWriter. Including the 4 baseline experiments, we report 36 experiments, respectively. For FOLIO, we conduct 20 experiments in total (Pyke and ASP cannot be measured). This leads to a total of 92 experiments per LLM and 644 experiments in total. The overall number of queries is 43680 per LLM and overall 305760. Let $\#D$ be the dataset size, $\#\text{EXEC}$ the number of correctly parsed instances, and $\#\text{TRUE}$ the number of correctly solved instances. *Syntactically correct* refers to a translation that adheres to the defined formal language, whereas *correctly solved* refers to a correct syntactical translation and the correct output of the solver. The *execution-rate* is the fraction of correct syntactical outputs (Exec-Rate, $\frac{\#\text{EXEC}}{\#D}$), *execution-accuracy*, is the fraction of correctly solved instances of all syntactically correct ones (Exec-Acc, $\frac{\#\text{TRUE}}{\#\text{EXEC}}$), and *overall-accuracy* is the fraction of correctly solved instances over the entire dataset (Overall-Acc, $\frac{\#\text{TRUE}}{\#\#\text{D}}$). Observe: *Overall-Acc* $=$ *Exec-Acc* $\cdot$ *Exec-Rate*. Baselines which do not use neurosymbolic reasoning are considered to have an execution-rate of 100%, while their execution-accuracy resembles their overall-accuracy, as they are not required to adhere to a formal language.

5. Results

We show the experimental results in Figure 2 and Table 1. In this paper we focus on the main results of the formal languages and provide further information on reasoning model performance and averaged LLM performance. Further results are shown in the main paper, such as an ablation study on eight different prompting styles and how well formal languages work on each dataset. Overall we report that FOL achieves the best results, followed by NLTK, ASP, and lastly Pyke. We report these findings in

[7]We additionally show the results of Neurosymbolic baselines in the main paper.

Lang.	Overall Results		GPT-4o-mini			Ministral-8B		
	Avg.	SEM	Avg.	SEM	Max	Avg.	SEM	Max
Std.	/	/	/	/	70.20	/	/	48.80
CoT	/	/	/	/	84.00	/	/	86.60
Pyke	45.83	2.82	59.76	4.56	93.80	39.93	5.15	69.60
ASP	53.94	2.94	61.00	4.17	97.20	34.98	3.42	61.40
NLTK	60.36	2.92	72.74	5.42	99.80	75.20	6.40	99.60
FOL	**65.29**	2.52	**72.85**	4.75	**100.00**	**75.94**	6.40	**100.00**

Lang.	Llama-8B			DeepSeek-8B			DeepSeek-32B		
	Avg.	SEM	Max	Avg.	SEM	Max	Avg.	SEM	Max
Std.	/	/	52.00	/	/	**87.40**	/	/	**99.20**
CoT	/	/	68.80	/	/	**87.40**	/	/	98.80
Pyke	20.76	5.43	61.00	1.43	0.48	5.80	53.44	3.48	79.40
ASP	6.78	1.15	19.33	23.98	4.14	46.40	**63.74**	1.94	77.60
NLTK	**54.94**	5.76	**93.20**	15.77	2.13	29.00	59.44	5.58	96.00
FOL	51.97	4.74	77.00	**33.15**	4.58	69.80	61.11	5.67	87.00

Lang.	DeepSeek-V3			DeepSeek-R1			DeepSeek-R1 (20480)		
	Avg.	SEM	Max	Avg.	SEM	Max	Avg.	SEM	Max
Std.	/	/	98.00	/	/	57.60	/	/	97.40
CoT	/	/	99.80	/	/	**81.80**	/	/	99.00
Pyke	68.20	2.94	82.40	12.88	4.93	73.17	77.28	4.76	98.00
ASP	78.32	5.80	99.00	**48.17**	3.09	74.00	**88.82**	1.81	98.40
NLTK	**84.07**	5.40	**100.00**	20.89	3.89	46.67	80.33	5.52	98.40
FOL	76.54	6.57	**100.00**	21.09	4.11	43.17	85.45	4.12	**99.80**

Table 1
Overall and per LLM results. LLMs prompted with temperature 0 and max-output-tokens 2048, except for DeepSeek-R1 (20480). All values in $[\%]$. For overall results Avg. and SEM, $n = 112$. For per LLM result: $n = 16$ for the neurosymbolic approaches and $n = 2$ for the baselines.

Figure 2 (left) and Table 1 (left top), where we show averaged results with the standard error of the mean (SEM). For averaging the formal languages, we compute the average across all LLMs, prompting styles, and the datasets ProntoQA and ProofWriter, leading to $n = 112$. To account for a fair comparison and incorporation of the results of the reasoning model, we only used the 20480 token results for DeepSeek-R1 for these averages. For the problems in the datasets, we do not encounter difficulties when solving in terms of *intractability* - a combinatorial explosion in the solver - we never exceed 60s computation time. Therefore, we are not required to use special strategies for tackling intractability, such as symmetry breaking [30] or tackling the ASP bottleneck [31]. In Figure 2 (right), we show averaged scatter plots of the execution-rate (x-axis) vs. execution-accuracy (y-axis). Each dot represents a formal language with a specific prompting style, averaged across all LLMs and ProntoQA, and ProofWriter datasets ($n = 14$). The overall-accuracy is obtained by multiplying a point's x-position with its respective y-position. We report that Pyke performs approximately equally well on execution-rate and execution-accuracy, while ASP's execution-rate tends to stay relatively high. Further, NLTK's execution-accuracy is relatively high, as it stays above 60%. FOL's behavior is similar to NLTK's. Still FOL has a higher execution-rate, resulting in a higher overall-accuracy.

In Table 1 (all except left top) we show the individual results of the formal languages on the LLMs. We report that the results differ widely between LLMs. Considering the average case, FOL achieves the highest results on three (GPT-4o-mini, Ministral-8B, and DeepSeek-8B), NLTK on two (Llama-8B and DeepSeek-V3) and ASP on three (DeepSeek-32B, DeepSeek-R1, and DeepSeek-R1 (20480)) LLMs. However, these values are often in the range of the SEM, therefore, inconclusive. For example, on GPT-4o-mini FOL has 72.85% and an SEM of 4.75%, whereas NLTK has 72.74% and an SEM of

5.42%. Regarding the best results, FOL achieves on four (GPT-4o-mini, Ministral-8B, DeepSeek-V3, and DeepSeek-R1 (20480)), NLTK on two (Llama-8B and DeepSeek-V3), and the baselines on three (DeepSeek-8B, DeepSeeek-32B, and DeepSeek-R1) LLMs. We report the largest performance improvements on SLMs, such as GPT-4o-mini, Ministral-8B, and Llama-8B. However, on SLMs also the performance fluctuates the most. For example, ASP does not perform well on Llama-8B, whereas Pyke does not perform well on DeepSeek-8B. The reasoning model DeepSeek-R1 needs more output tokens to generate suitable responses, when compared to the chat models. Although not a surprise, this has consequences. While for all chat LLMs a max-output-token size of 2048 is sufficient, for DeepSeek-R1 only a max-output-token size of 20480 yields reliably non-truncated results.

6. Conclusion and Discussion

Logical reasoning tasks pose a problem to LLMs, as they remain limited in their ability to perform probabilistic retrieval [3]. Neurosymbolic approaches help, by constraining the probabilistic nature to the translation step of a natural language-posed problem into a formal language [7, 8]. Therefore, the reasoning step itself is not affected by the probabilistic nature of LLMs.

In this paper, we discuss the effect of the chosen formal language on a model's reasoning performance. We introduce the *intermediate language challenge*, which refers to the problem of choosing a suitable formal language for neurosymbolic reasoning. In our experiments, we compare Pyke, ASP, NLTK, and FOL as formal languages. The results show that FOL performs best, followed by NLTK and ASP, with Pyke coming last. When analyzing the behavior of different formal languages we observed translation errors which were unique to the formal language, and errors common across different formal languages. For *Pyke* in particular, we notice that LLMs format the output incorrectly, by missing line breaks. When translating to *ASP*, LLMs struggle to distinguish the two notions of negation: *strong*, written as $-$, and *default*, written as not. This results in program statements such as *-not p1(wren)*, which are syntactically incorrect. The syntactic errors between *NLTK* and *FOL* are similar. Examples include incorrectly setting parentheses or using a predicate with multiple arities - e.g., $p14(X)$ and $p14(X, Y)$.

Our results between different LLMs vary widely and suggest that for neurosymbolic LLM reasoning the usage of huge LLMs is not always justified, as for our tasks we already achieved 100% max-accuracy on GPT-4o-mini and Ministral-8B (8B models). Further, while for all chat models max-output-tokens of 2048 was sufficient, this was not the case for DeepSeek-R1, where we needed more (we increased the parameter to 20480). Interestingly, our results indicate that ASP uses less output tokens, as it achieved decent results on DeepSeek-R1 (2048). Further, we observed the largest improvements w.r.t. to the baselines and the largest variations in performance on small language models (SLMs). We hypothesize that the performance differences can be explained with a lack or abundance of the formal languages in the training data. This opens the door for crafting custom intermediate languages and fine-tuning the SLMs on these custom languages, which we plan to explore in a future study.

Declaration on Generative AI

During the preparation of this work, the authors used OpenAI O3 and Grammarly in order to: Grammar and spelling check. The seven LLMs (*GPT-4o-mini*, *Ministral-8B*, *Llama-8B*, *DeepSeek-8B*, *DeepSeek-32B*, *DeepSeek-V3* and *DeepSeek-R1*) were used, as discussed in Section 4.

Acknowledgements

This research was supported by Frequentis, FWF grant 10.55776/COE12, and the Dieberger-Peter Skalicky Stipend.

References

[1] A. Saparov, H. He, Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought, ICLR23 (2023).

[2] A. K. Lampinen, I. Dasgupta, S. C. Y. Chan, H. R. Sheahan, A. Creswell, D. Kumaran, J. L. McClelland, F. Hill, Language models, like humans, show content effects on reasoning tasks, PNAS Nexus 3 (2024). doi:10.1093/pnasnexus/pgae233.

[3] D. Panas, S. Seth, V. Belle, Can Large Language Models Put 2 and 2 Together? Probing for Entailed Arithmetical Relationships, in: NeSy24, 2024, pp. 258–276. doi:10.1007/978-3-031-71170-1_21.

[4] B. Y. Lin, S. Lee, R. Khanna, X. Ren, Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models, in: EMNLP20, 2020, pp. 6862–6868. doi:10.18653/v1/2020.emnlp-main.557.

[5] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, D. Zhou, Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, in: NeurIPS22, 2022, pp. 24824–24837.

[6] Q. Lyu, S. Havaldar, A. Stein, L. Zhang, D. Rao, E. Wong, M. Apidianaki, C. Callison-Burch, Faithful Chain-of-Thought Reasoning, in: IJCNLP23, 2023, pp. 305–329. doi:10.18653/v1/2023.ijcnlp-main.20.

[7] L. Pan, A. Albalak, X. Wang, W. Wang, Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning, in: EMNLP23, 2023, pp. 3806–3824. doi:10.18653/v1/2023.findings-emnlp.248.

[8] T. Olausson, A. Gu, B. Lipkin, C. Zhang, A. Solar-Lezama, J. Tenenbaum, R. Levy, LINC: A Neurosymbolic Approach for Logical Reasoning by Combining Language Models with First-Order Logic Provers, in: EMNLP23, 2023, pp. 5153–5176. doi:10.18653/v1/2023.emnlp-main.313.

[9] A. Beiser, D. Penz, N. Musliu, Intermediate Languages Matter: Formal Choice Drives Neurosymbolic LLM Reasoning, 2025. URL: https://arxiv.org/abs/2502.17216.

[10] S. Kirtania, P. Gupta, A. Radhakrishna, LOGIC-LM++: Multi-Step Refinement for Symbolic Formulations, in: ACL24, 2024, pp. 56–63. doi:10.18653/v1/2024.nlrse-1.6.

[11] M. Geva, A. Gupta, J. Berant, Injecting Numerical Reasoning Skills into Language Models, in: ACL20, 2020, pp. 946–958. doi:10.18653/v1/2020.acl-main.89.

[12] E. Coppolillo, F. Calimeri, G. Manco, S. Perri, F. Ricca, LLASP: Fine-tuning Large Language Models for Answer Set Programming, in: KR24, 2024, pp. 834–844. doi:10.24963/kr.2024/78.

[13] M. Shanahan, Talking about Large Language Models, Com. ACM 67 (2024) 68–79. doi:10.1145/3624724.

[14] B. Sel, A. Al-Tawaha, V. Khattar, R. Jia, M. Jin, Algorithm of thoughts: enhancing exploration of ideas in large language models, in: ICML24, 2024.

[15] A. d. Garcez, L. C. Lamb, Neurosymbolic AI: the 3rd wave, Artif Intell Rev 56 (2023) 12387–12406. doi:10.1007/s10462-023-10448-w.

[16] S. Badreddine, A. d'Avila Garcez, L. Serafini, M. Spranger, Logic Tensor Networks, AI 303 (2022) 103649. doi:10.1016/j.artint.2021.103649.

[17] T. Eiter, T. Geibinger, N. Higuera, J. Oetsch, A logic-based approach to contrastive explainability for neurosymbolic visual question answering, in: IJCAI23, 2023, pp. 3668–3676. doi:10.24963/ijcai.2023/408.

[18] Y. Wu, A. Q. Jiang, W. Li, M. N. Rabe, C. Staats, M. Jamnik, C. Szegedy, Autoformalization with Large Language Models, 2022. URL: http://arxiv.org/abs/2205.12615.

[19] J. Liu, S. Cao, J. Shi, T. Zhang, L. Hou, J. Li, How Proficient Are Large Language Models in Formal Languages? An In-Depth Insight for Knowledge Base Question Answering, in: ACL24, 2024.

[20] N. Li, P. Liu, Z. Liu, T. Dai, Y. Jiang, S.-T. Xia, Logic-of-thought: Empowering large language models with logic programs for solving puzzles in natural language, 2025. URL: https://arxiv.org/abs/2505.16114.

[21] O. Tafjord, B. Dalvi, P. Clark, ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language, in: IJCNLP21, 2021, pp. 3621–3634. doi:10.18653/v1/2021.findings-acl.317.

[22] S. Han, H. Schoelkopf, Y. Zhao, Z. Qi, M. Riddell, W. Zhou, J. Coady, D. Peng, Y. Qiao, L. Benson, e. al., FOLIO: Natural Language Reasoning with First-Order Logic, in: EMNLP24, 2024, pp. 22017–22031. doi:10.18653/v1/2024.emnlp-main.1229.

[23] B. Frederiksen, Applying Expert System Technology to Code Reuse with Pyke, 2008. URL: https://pyke.sourceforge.net/PyCon2008-paper.html.
[24] M. Gelfond, N. Leone, Logic programming and knowledge representation—The A-Prolog perspective, AI 138 (2002) 3–38. doi:10.1016/S0004-3702(02)00207-2.
[25] T. Schaub, S. Woltran, Special Issue on Answer Set Programming, Künstliche Intell. 32 (2018) 101–103. doi:10.1007/s13218-018-0554-8.
[26] R. Kaminski, T. Schaub, On the Foundations of Grounding in Answer Set Programming, TPLP23 23 (2023) 1138–1197. doi:10.1017/S1471068422000308.
[27] M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, P. Wanko, Theory Solving Made Easy with Clingo 5, ICLP16 52 (2016) 1–15. doi:10.4230/OASICS.ICLP.2016.2.
[28] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python, O'Reilly Media Inc., 2009.
[29] W. McCune, Prover9 and Mace4, 2010. URL: http://www.cs.unm.edu/~mccune/Prover9.
[30] T. Fahle, S. Schamberger, M. Sellmann, Symmetry breaking, in: T. Walsh (Ed.), CP01, 2001, pp. 93–107.
[31] A. Beiser, M. Hecher, K. Unalan, S. Woltran, Bypassing the ASP Bottleneck: Hybrid Grounding by Splitting and Rewriting, in: IJCAI24, 2024, pp. 3250–3258. doi:10.24963/ijcai.2024/360.