An Evaluation of Open Source LLMs for Neuro-Symbolic Integration

Stefano Sambri, Atefeh Ghanbari and Fabrizio Riguzzi*

Dipartimento di Matematica e Informatica, Università di Ferrara
Via Machiavelli 30, 44121 Ferrara, Italy

Abstract

Large Language Models (LLMs) have shown impressive capabilities but still struggle in reasoning. Even with advanced prompting techniques such as Chain-of-thought, they often make reasoning mistakes. A recent approach to overcome this difficulty consists in integrating an LLM with an external reasoner, realizing a form of neuro-symbolic integration. The LLM in this case is used to translate the multi-modal unstructured description of the problem (text, images) into a formal representation, often based on logic, which is then provided to the reasoner that computes the answer. Closed source models such as ChatGPT 4o have impressive performance for this task but they are expensive and require the data to be uploaded in the cloud, which poses privacy problems. In this paper we investigate the performance of smaller open source models on the problem of describing images using Prolog facts, to be used by a downstream reasoner.

Keywords

Neuro-symbolic integration, Large Language Models, Logic Programming

1. Introduction

Large Language Models (LLMs) have significantly advanced the field of Natural Language Processing (NLP) [1], particularly in areas such as text generation, translation, and question answering [2]. These models represent a major milestone in artificial intelligence (AI), enabling near-human levels of language understanding and production. Yet, despite these impressive capabilities, a fundamental question remains: Can LLMs truly reason? [3, 4]

Reasoning is a fundamental component of human intelligence and remains one of the central challenges in the field of AI [5]. While LLMs demonstrate some reasoning capabilities, they often struggle with tasks requiring symbolic, causal, and relational consistency.

To address these limitations, prompting strategies like Chain-of-Thought (CoT) have been introduced [6]. However, open-source models still fall short in complex reasoning tasks, while advanced closed-source systems such as GPT-4o exhibit better performance [7] albeit with concerns related to cost, transparency, and privacy.

In this study, we evaluate open-source LLMs within a neuro-symbolic framework for translating multimodal input into formal logic representations. We investigate the task of generating Prolog code from images. Our findings offer insights into the current capabilities and limitations of open-source LLMs, and highlight the persistent challenges that must be addressed to achieve more robust and generalizable reasoning in future AI systems.

The paper is organized as follows. Section 2 introduces LLMs and Section 3 presents the datasets that have been used for the evaluation. Section 4 discusses the research questions we aim to answer with this study. Section 5 shows the performance of the closed-source model GPT-4o, while Section 6 those of open source models. Section 7 concludes the paper.

CILC 2025: 40th Italian Conference on Computational Logic, June 25–27, 2025, Alghero, Italy
*Corresponding author.
stefano.sambri@unife.it (S. Sambri); atefeh.ghanbari33@gmail.com (A. Ghanbari); fabrizio.riguzzi@unife.it (F. Riguzzi)
https://ml.unife.it/fabrizio-riguzzi (F. Riguzzi)
0000-0003-1654-9703 (F. Riguzzi)
© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

CEUR-WS.org/Vol-4003/paper03.pdf

2. Large Language Models

Large Language Models (LLMs) have significantly advanced AI capabilities in natural language generation, reasoning, and decision-making [8]. However, they exhibit persistent limitations in tasks requiring structured or symbolic reasoning.

2.1. Reasoning Challenges in LLMs

Despite recent breakthroughs in natural language processing, models such as ChatGPT-4o continue to face core limitations in symbolic and logical reasoning. These challenges are even more evident in open-source LLMs, highlighting the need for comprehensive evaluation in neuro-symbolic integration settings.

At the core, LLMs rely on statistical patterns rather than genuine comprehension. Though they can mimic reasoning, their limited semantic and logical understanding may at times produce plausible yet incorrect outputs — a phenomenon known as hallucination [4]. A key limitation of LLMs lies in multi-step reasoning, where they often fail to correctly build upon earlier inferences, resulting in compounded logic errors and incorrect conclusions [5, 9].

Inconsistencies in output generation, such as misinterpreting conditions, ignoring constraints, or hallucinating data, raise serious concerns about the reliability and interpretability of these models [10]. Moreover, LLMs tend to show an overconfidence in their predictions, creating a dangerous mismatch between confidence and factual correctness, especially in safety-critical contexts [11].

Another core limitation lies in handling abstract and counterfactual reasoning. LLMs frequently produce contradictory or logically inconsistent explanations when faced with counterfactual prompts [12, 9]. Additionally, their outputs are sensitive to prompt phrasing and dataset biases, which complicates reproducibility and fairness [13, 14].

From a computational standpoint, despite advancements in scale and architecture, LLMs remain constrained by fundamental limitations, as scaling alone does not resolve core bottlenecks in symbolic and logical reasoning—reflecting the "curse of complexity" [15]. LLMs face persistent challenges in mathematical reasoning, often producing correct answers through flawed or superficial reasoning, particularly in tasks requiring arithmetic accuracy, spatial understanding, and structured deduction [16, 9].

Lastly, current LLMs lack the capability for online learning or dynamic adaptation. This inhibits their ability to respond to new inputs or evolving contexts in real time, limiting their usefulness in interactive and ever-changing environments [17].

Given these persistent challenges, hybrid approaches combining neural and symbolic reasoning have emerged as a promising solution, which we discuss in the next section.

2.2. Solutions with Neuro-Symbolic Integration

To address the reasoning limitations of LLMs, a range of neuro-symbolic approaches have been proposed. These methods aim to strengthen logical and mathematical reasoning by combining the perception capabilities of deep neural networks with the soundness and reliability of symbolic reasoning [18].

Controlling generation at inference time is challenging due to the inherent difficulty and general intractability of conditioning large language models on logical constraints [19, 20]. The Ctrl-G framework [19] combines production-ready LLMs with Hidden Markov Models, improving the reliability and controllability of outputs by enforcing adherence to logical constraints represented as deterministic finite automata.

The DSR-LM framework [21] enhances logical reasoning in LLMs by combining pre-trained language models for factual understanding with a differentiable symbolic module for deductive reasoning. It learns weighted rules and uses semantic loss to improve performance, providing a scalable and interpretable method for integrating prior knowledge.

Maintaining logical consistency is a persistent challenge for neural sequence models, leading to poor performance on structured reasoning tasks. To mitigate this, a training-free framework inspired by the dual-process theory of cognition has been proposed [22]. This framework filters the model's outputs through a symbolic module, enforcing logical coherence and bridging intuitive (System 1) and analytical (System 2) reasoning.

LLMs often lack precise reasoning and self-correction capabilities, which symbolic systems handle more reliably. LLM-ARC [23] addresses these gaps through a neuro-symbolic Actor-Critic framework, leveraging symbolic evaluation to iteratively refine logic generation. Reasoning is further enhanced through Answer Set Programming and self-supervised feedback.

In another line of work, Logic-LM [24] addresses LLMs' limitations in complex reasoning by translating natural language into formal logic, applying symbolic inference, and refining outputs via solver feedback. Likewise, SatLM [25] employs declarative specifications and theorem provers to solve constraint-based problems more reliably than chain-of-thought prompting.

NeSyGPT [26] combines symbolic feature extraction with Answer Set Programming using minimal labeled data, demonstrating an efficient pipeline for neural-symbolic integration with enhanced scalability.

In the domain of structured query generation, Jiao et al. [27] use unification-based grammars to ensure syntactic and schema validity in SQL outputs, illustrating the power of grammatical reasoning for improving robustness.

Further, LLM2LAS [28] tackles commonsense reasoning by coupling LLMs with ILASP to learn Answer Set Programs from minimal supervision, enabling strong generalization to unseen queries. Creswell et al. [29] improve logical reasoning through a decomposition approach using fine-tuned models for information selection and inference, producing structured and interpretable reasoning chains.

Collectively, these models underscore the importance of logic integration for elevating LLM reasoning. Techniques such as knowledge graph integration, program-guided generation, reinforcement learning, and symbolic validation represent promising steps toward more explainable and robust AI systems [9].

In summary, while existing solutions offer promising directions for mitigating LLM reasoning limitations via symbolic integration, none provide a comprehensive remedy, especially on multimodal data. As the practical effectiveness of many neuro-symbolic frameworks remains uncertain, the following section introduces the datasets used to evaluate their capabilities image analysis tasks.

3. Datasets

3.1. RAVEN

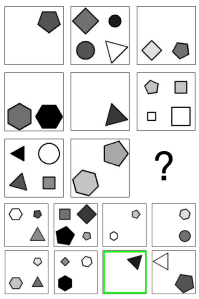
The RAVEN dataset[1] [30] has been proposed as a challenge for Computer Vision systems. RAVEN is an instance of Raven's Progressive Matrices, a cognitive test proposed by John C. Raven where one is given 9 panels organized in a 3×3 matrix. Of these, one is hidden and the aim is to select a panel, among a list of 8 other panels, which best completes the matrix. Figure 1 shows an example where the correct panel is bordered in green.

3.2. Tic Tac Toe

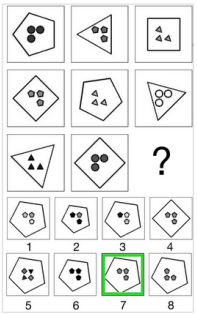
In the Tic-tac-toe game, two players take turns in placing the marks X and O on a 3x3 board. The first player who aligns three of her marks, either vertically, horizontally or diagonally, wins. The game is a draw is no more move can be made. We consider two sub-tasks:

1. Given the image of a board of a finished game, determine which player won or whether there was a draw.

[1] https://github.com/WellyZhang/RAVEN

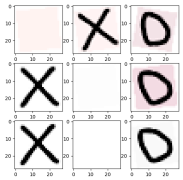


(a) Example of RAVEN problem.

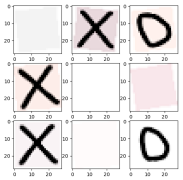


(b) Example of RAVEN problem.

Figure 1: Predict the correct panel in RAVEN



(a) Predict the winner in Tic Tac Toe



(b) Predict the next move in Tic Tac Toe

Figure 2: Examples for the Tic Tac Toe dataset using handwritten images.

2. Given the image of a board of an ongoing game, determine what move the next player should take to maximize the chances of winning.

Datasets for these sub-tasks can be generated with the code at [31].

This dataset requires both perception and reasoning capabilities: in both sub-tasks, the system must recognize marks and their positions on the board. In the first sub-tasks, the system must perform geometrical reasoning on the marks. In the second sub-task, the system must apply a game playing strategy.

The instances of these tasks are images of a board. To represent the marks, we adopt two approaches. In the first, starting from 3 handwritten images created with Inkscape representing X, O and blank, we generate variations using

- summing a random integer in the range [-13,+13] to hue, saturation and lightness (HSV)
- applying a random rotation in the range [-10,+10] degrees

In the second approach to represent marks, we use MNIST digit images: we use images of the digits 1 and 2 and 0 for X, O and blank respectively.

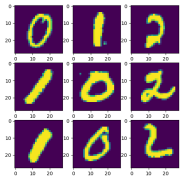
Figure 2 shows instances of the two sub-tasks using handwritten symbols: Figure 2a shows an example for sub-task 1 where the winner is player 2, while Figure 2b shows a board with label (2,3), meaning that the next symbol has to be placed in the second row, third column.

Figure 3a shows the same instances as Figure 2a but using MNIST digits.

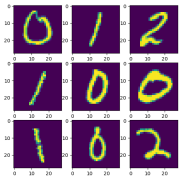
The two datasets are randomly generated using probabilistic logic programming: first a board is sampled, then ProbLog is used to provide the label of the board and finally the image of the board is generated by randomly sampling images for 1, 2 and 0. In this way, the example images are all different and pose challenges for perception.

4. Research Questions

The research questions we aim to answer are:



(a) Predict the winner in Tic Tac Toe



(b) Predict the next move in Tic Tac Toe

Figure 3: Examples for the Tic Tac Toe dataset using MNIST digits.

RQ1 Are current LLMs able to solve multimodal problems requiring reasoning such as RAVEN and Tic Tac Toe?

RQ2 Are they able to describe the images using logic programming so that downstream reasoners can be applied?

RQ3 Are there differences between closed-source and open-source models?

To answer these questions, we constructed a standardized set of prompts that serve as a consistent framework for evaluating the performance of both GPT-4o and open-source LLMs. The prompts are:

RAVEN Prompts

PR1 **Image:** Fig. 1a **Question:** What is the picture that should replace the question mark among those of the last two rows?

PR2 **Image:** Fig. 1a **Question:** Describe the picture using logic programming.

PR3 **Image:** Fig. 1b **Question:** same as PR1.

PR4 **Image:** Fig. 1b **Question:** same as PR2.

Handwritten Tic Tac Toe Prompts

PT1 **Image:** Fig. 2a **Question:** Who is the winner?

PT2 **Image:** Fig. 2a **Question:** Describe this tic tac toe board using logic programming.

PT3 **Image:** Fig. 2b **Question:** Where should player 2 ("O") move?

PT4 **Image:** Fig. 2b **Question:** same as PT2.

MNIST Tic Tac Toe Prompts

PM1 **Image:** Fig. 3a **Question:** Given this Tic Tac Toe board and the fact that 0 represents blank, 1 represents player 1 and 2 represents player 2, who is the winner?

PM2 **Image:** Fig. 3a **Question:** Given this Tic Tac Toe board and the fact that 0 represents blank, 1 represents player 1 and 2 represents player 2, describe this board using logic programming.

PM3 **Image:** Fig. 3b **Question:** Given this Tic Tac Toe board and the fact that 0 represents blank, 1 represents player 1 and 2 represents player 2, where should player 2 move?

PM4 **Image:** Fig. 3b **Question:** same as PM2.

The answers of GPT-4o and open-source models are then manually assigned one of the following four classes:

- ✓ Complete and correct answers.
- ✓ Answers that are partially or could be interpreted as correct.
- × Wrong answers.
- - Malformed answers with missing or errors in token generation.

5. Performance of GPT-4o

GPT-4o is an autoregressive multimodal model capable of processing and generating text, audio, and images. Unlike previous models that rely on separate components for different modalities, GPT-4o is trained end-to-end across textual, visual, and auditory inputs, allowing it to handle any combination of these modalities using a single unified neural architecture. Notably, GPT-4o demonstrates improved performance in visual tasks compared to existing models [32]. To evaluate the performance of ChatGPT-4o, the results corresponding to the prompts of Section 4 are summarized in tables 1, 2 and 3 for RAVEN, Tic Tac Toe with handwritten images and Tic Tac Toe with MNIST images.

Prompts	ChatGPT-4o
PR1	×
PR2	×
PR3	×
PR4	×

Table 1
Raven dataset (Fig. 1).

Prompts	ChatGPT-4o
PT1	×
PT2	✓
PT3	✓
PT4	✓

Table 2
Tic Tac Toe dataset (Fig. 2).

Prompts	ChatGPT-4o
PM1	✓
PM2	✓
PM3	✓
PM4	✓

Table 3
Tic Tac Toe MNIST dataset (Fig. 3).

To clarify the operation of ChatGPT-4o, we report below part of the answer to prompt PR1:

...

```
% Format: cell(Row, Column, [shape(Type, Color, Size, Orientation)])
% Row 1
cell(1,1, [shape(pentagon, black, medium, normal)]).
cell(1,2, [shape(square, gray, medium, rotated), shape(circle, black, small, normal),
↪  shape(triangle, white, medium, rotated)]).
cell(1,3, [shape(square, gray, small, rotated), shape(diamond, gray, small, rotated),
↪  shape(pentagon, black, medium, normal)]).
% Row 2
cell(2,1, [shape(hexagon, gray, medium, normal), shape(hexagon, black, medium,
↪  normal)]).
cell(2,2, [shape(triangle, black, medium, normal)]).
cell(2,3, [shape(hexagon, gray, small, normal), shape(square, gray, small, normal),
↪  shape(square, white, medium, normal)]).
% Row 3
cell(3,1, [shape(triangle, black, small, rotated), shape(circle, white, medium,
↪  normal), shape(square, black, medium, normal)]).
cell(3,2, [shape(hexagon, gray, medium, rotated)]).
cell(3,3, ?). % Missing, to be inferred
```

... (Generated by AI tool ChatGPT-4o)

As you can see, the result is incorrect: for example, cell (1,2) is missing a circle and cell (2,3) contains a pentagon and not a hexagon. In general, some geometric figures are missing and some have the wrong number of sides, highlighting a problem in counting.

Instead, for prompt PT2, we obtain a correct answer:

...

```
% Format: cell(Row, Column, Content).
% Content can be 'x', 'o', or 'empty'
cell(0, 0, empty). cell(0, 1, x).     cell(0, 2, o).
cell(1, 0, x).     cell(1, 1, empty). cell(1, 2, o).
cell(2, 0, x).     cell(2, 1, empty). cell(2, 2, o).
```

... (Generated by AI tool ChatGPT-4o)

While ChatGPT-4o exhibited strong performance on simpler visual reasoning tasks, such as those in Tic Tac Toe, its performance was significantly lower on more complex tasks like those found in the RAVEN dataset, which can be attributed to the complexity of the visual patterns. The next section examines how open-source models perform on similar multimodal reasoning tasks.

6. Performance of Open Source Models

We tested the following open source models, capable of processing an image with a text input and returning a response based on both:

- GIT-base-textvqa[2] [33] is an image-text-to-text model, based on a decoder-only transformer that uses the CLIP [34] vision encoder, fine-tuned for Q&A on images.
- BLIP-2-flan-t5-xxl[3] is an image-text-to-text model, based on CLIP as vision encoder, a Q-Former [34] that bridges between vision and text and Flan-T5-xxl [35] as LLM.
- Deepseek-VL2[4] [36], Deepseek-VL2-small[5] [36] and Deepseek-VL2-tiny[6] [36] are image-text-to-text models, composed of a vision-encoding, a vision-language adapter and Misture of Experts (MoE) [37] architecture.
- Molmo-72B-0924[7] [38] and Molmo-7B-D-0924[8] [38] are image-text-to-text models based on CLIP as vision encoder and Qwen2 [39] as LLM.
- Molmo-72B-0924-nf4[9] is the quantized version of Molmo-72B-0924 in normalized 4 bit float.
- Molmo-7B-O-0924[10] [38]is an image-text-to-text model based on on CLIP as vision encoder and OLMo-7B-1024 [40] as LLM.
- Cogvlm-chat-hf[11][41] is an image-text-to-text model using ViT [42] as vision encoder, an MLP adapter using SwiGLU [43] as activation function, Vicuna1.5-7B [44] as LLM and a visual expert module composed of a QKV matrix and an MLP in each layer.
- MiniCPM-o-2_6 [12] [45] is a multi-modal large language model based on SigLip-400M [46], a vision encoder based on CLIP trained with a sigmoid loss function, Whisper-medium [47] a speech-to-text model, ChatTTS, [48] a text-to-speech model, and Qwen2.5-7B [49] as LLM.

We tested all models using default options and setting a high number of max sentence tokens. All of the three Deepseek versions tested, GIT-base-textVQA, CogVLM-chat-hf and BLIP-2-Flan-T5-xxl were not able to generate logic programming code. All of the four Molmo were able to, and Molmo-72B-0924

[2] https://huggingface.co/microsoft/git-base-textvqa
[3] https://huggingface.co/Salesforce/blip2-flan-t5-xxl
[4] https://huggingface.co/deepseek-ai/deepseek-VL2
[5] https://huggingface.co/deepseek-ai/deepseek-VL2-small
[6] https://huggingface.co/deepseek-ai/deepseek-VL2-tiny
[7] huggingface.co/allenai/Molmo-72B-0924
[8] https://huggingface.co/allenai/Molmo-7B-D-0924
[9] https://huggingface.co/SeanScripts/Molmo-72B-0924-nf4
[10] https://huggingface.co/allenai/Molmo-7B-O-0924
[11] https://huggingface.co/THUDM/cogvlm-chat-hf
[12] https://huggingface.co/openbmb/MiniCPM-o-2_6

was the only one able to generate code that could be interpreted and led to an answer that could be considered as correct. MiniCPM-o-2_6 was able to generate prolog code but that could not be used by an interpreter and prompted to generate logic programming sometimes returned python.

Spatial information was the main reason for model failure, even if they were able to identify items in the images, they were not able to position them correctly or reconstruct the patterns.

6.1. Results on the Raven dataset

All three versions of Deekseek-VL2 (Tab. 4) demonstrated an understanding of the task in relation to the image. Deekseek-VL2-small attempted to generate text coordinates for a bounding box in both images but failed to produce a coherent box in Fig. 1a and it partially overlapped with an incorrect answer in Fig. 1b. Deekseek-VL2 was able to describe the problem that had to be solved, reflecting an understanding of the relation between the prompt and the image, but it too failed to provide the correct answer. Deekseek-VL2-tiny came close to generating a correct response for Fig. 1b but struggled to precisely identify smaller shapes within the image.

Prompts	Deepseek-VL2	Deepseek-VL2-small	Deepseek-VL2-tiny
PR1	×	-	×
PR2	×	-	×
PR3	×	-	×
PR4	-	-	×

Table 4
Deepseek on the Raven dataset (Fig. 1)

Molmo-7B-D-0924, Molmo-7B-O-0924, and Molmo-72B-0924 (Tab. 5), despite their differences in parameters and architectures, answered similarly and wrong, failing to understand the task presented and returning a generic shape as an answer as shown. The answers given by Molmo-72B-0924-nf4 on both images were close to the correct solution but didn't distinguish all of the correct shapes to pinpoint the precise answer. In fact, on Fig. 1a the model hallucinated an extra square and on Fig. 1b it was able to correctly identify the solution disposition, but failed to distinguish the exact shapes. Looking

Prompts	Molmo-72B-0924	Molmo-72B-0924-nf4	Molmo-7B-D-0924	Molmo-72B-O-0924
PR1	×	×	×	×
PR2	✓	✓	✓	✓
PR3	×	✓	×	×
PR4	✓	✓	✓	✓

Table 5
Molmo on the Raven dataset (Fig. 1)

at Tab. 6, it is interesting to note that CogVLM-chat-hf answered similarly to Molmo-72B-0924, and BLIP-2-Flan-T5-xxl the same as Molmo-72B-0924-nf4 despite the smaller number of parameters and differences in training datasets, suggesting some form of learned logic from both datasets. Git-base-textvqa failed to answer both questions and returned an empty string on Fig. 1a and a simple "no" on Fig. 1b. MiniCPM-o-2_6 was able to give the correct answer even though it strongly hallucinated while describing the board.

Prompts	GIT-base-textVQA	BLIP2-Flan-T5-xxl	CogVLM-chat-hf	MiniCPM-o-2_6
PR1	-	×	✓	×
PR2	-	×	×	×
PR3	-	×	×	✓
PR4	-	×	×	×

Table 6
Other models on the Raven dataset (Fig. 1)

Making models describe the problem in logic programming terms, for those who were able to, resulted in the generation of prolog code that was not directly interpretable. All models that generated Prolog had the same problems as answering questions directly: hallucinations on shapes, patterns and positions. Molmo models based on Qwen were able to generate more complex code in comparison to the version based on OLMo, which generated either some simple facts or simple rules. MiniCPM-o-2_6 was the only

model able to answer both questions on Fig. 1a correctly, but without generating any logic programming code and with strong hallucinations.

6.2. Results on Tic Tac Toe with Handwritten Digits

Considering Tab. 7, Tab. 8 and Tab. 9 on Fig. 2a Deekseek-VL2, Deekseek-VL2-small and GIT-base-textVQA failed to generate a coherent answer while all of the other models except for MiniCPM-o-2_6 marked, wrongly, the winner as the "X" player. MiniCPM-o-2_6 instead was able to correctly identify the winner in the "O" player even though it gave the wrong reason for why that was the case.

Prompts	Deepseek-VL2	Deepseek-VL2-small	Deepseek-VL2-tiny
PT1	-	-	×
PT2	-	-	×
PT3	-	-	×
PT4	-	-	×

Table 7
Deepseek on Tic Tac Toe dataset (Fig. 2)

Prompts	Molmo-72B-0924	Molmo-72B-0924-nf4	Molmo-7B-D-0924	Molmo-72B-O-0924
PT1	×	×	×	×
PT2	✓	✓	✓	×
PT3	×	×	×	×
PT4	✓	✓	✓	×

Table 8
Molmo on Tic Tac Toe dataset (Fig. 2)

Prompts	GIT-base-textVQA	BLIP2-Flan-T5-xxl	CogVLM-chat-hf	MiniCPM-o-2_6
PT1	-	×	×	✓
PT2	-	×	-	×
PT3	-	-	×	×
PT3	-	×	-	×

Table 9
Other models on Tic Tac Toe dataset (Fig. 2)

In the generation of logic programming code, only Molmo models based on Qwen were able to return an answer with structured code with the 72B models having improved answers over the 7B one, while Molmo-72B-O-0924 was not able to generate any prolog and MiniCPM-o-2_6 returned some simple facts. The generated code featured strong hallucinations in the board positions and the failure to capture precisely item positions, giving more insight on why the models failed on direct answers, as shown in:

> Here's a Prolog representation of the tic-tac-toe board:
>
> ```
> board([[X, X, O],[X, _, O],[_, _, _]]).
> ```
>
> "..." (Generated by AI tool Molmo-72B-0924 on Fig. 2b)

> Here's a description of the tic-tac-toe board using logic programming:
>
> ```
> tictactoe_board :- grid(3,3), row(1,[X,_,O]), row(2,[X,O,]), row(3,[X,_,O]),
> ↪  column(1,[X,X,X]), column(2,[O,O,]), column(3,[X,O,X]), diagonal(1,1,[X,X,O]),
> ↪  diagonal(2,2,[O,X,X]), diagonal(3,3,[X,X,O]), all_cells(true), X=true, O=true,
> ↪  empty_cells([]).
> ```
>
> "..." (Generated by AI tool Molmo-7B-D-0924 on Fig. 2a)

Considering Tab. 7, Tab. 8 and Tab. 9 on Fig. 2b, Deekseek-VL2, Deekseek-VL2-small, GIT-base-textVQA and BLIP-2-Flan-T5-xxl failed to generate a well structured answer. Molmo models and MiniCPM-o-2_6, all answered with the best move being to put the "O" in the middle, probably due to the general knowledge of tic tac toe game in their training data. CogVLM-chat-hf and Deekseek-VL2-tiny answered suggesting an already occupied space.

Even on Fig. 2b prompts, only molmo models based on Qwen were able to return a structured logic programming representation of the game board even if displaying strong hallucinations in the board position.

6.3. Results on Tic Tac Toe with MNIST Numbers

Deekseek-VL2 and Deekseek-VL2-small on Fig. 3a and on Fig. 3b (Tab. 10) failed to generate a well defined answer or failed to generate text coordinates for bounding boxes that ended up malformed.

Prompts	Deepseek-VL2	Deepseek-VL2-small	Deepseek-VL2-tiny
PM1	-	-	×
PM1	-	-	×
PM3	-	-	×
PM4	-	-	×

Table 10
Deepseek on Tic Tac Toe MNIST dataset (Fig. 3)

By looking at Tab. 10, Tab. 11 and Tab. 12, replacing handwritten images with MNIST numbers didn't seems to improve in the ability of models to correctly identify the winner, with the exceptions of CogVLM-chat-hf that was able to correctly identify player "2" as the winner and MiniCPM-o-2_6 generating a description of the board state where the correct winner could be inferred but failing to acknowledge it. All other models hallucinated on board states calling players with "1" as the winner or a tie.

Prompts	Molmo-72B-0924	Molmo-72B-0924-nf4	Molmo-7B-D-0924	Molmo-72B-O-0924
PM1	×	×	×	×
PM2	✓	✓	✓	×
PM3	×	×	✓	✓
PM4	✓	✓	✓	×

Table 11
Molmo on Tic Tac Toe MNIST dataset (Fig. 3)

Prompts	GIT-base-textVQA	BLIP2-Flan-T5-xxl	CogVLM-chat-hf	MiniCPM-o-2_6
PM1	×	×	✓	×
PM2	-	-	-	✓
PM3	-	-	×	×
PM3	-	-	-	✓

Table 12
Other models on Tic Tac Toe MNIST dataset (Fig. 3)

On Fig. 3b, in reference to Tab. 10 and Tab. 12, Deekseek-VL2, Deekseek-VL2-small, GIT-base-textVQA and BLIP-2-Flan-T5-xxl failed to generate a well structured and coherent answer. Other models suggested an occupied space or center square except for both Molmo-7B versions that suggested a correct move, even with some hallucination on the board state. The move doesn't lead to immediate winning but it will lead to a forced win for player "2". The 7B version based on Qwen also suggested an incorrect move to an occupied space.

As regards the generation of logic programming code, again, only molmo models based on Qwen returned structured prolog code with clauses an rules and Molmo-72B-0924 generated interpretable prolog code that marked player "2" as the winner, while before it answered that it was player "1" having won diagonally.

Molmo-72B-O-0924 was only able to generate some simple rules on Fig. 3b and MiniCPM-o-2_6, was able to generate simple clauses on Fig. 3a and some structured rules on Fig. 3b, both presenting strong hallucinations that didn't correctly represent the game state:

...

```
cell(0,0,1). cell(0,1,0). cell(0,2,0).
cell(1,0,2). cell(1,1,1). cell(1,2,2).
cell(2,0,1). cell(2,1,2). cell(2,2,0).
```

... (Generated by AI tool MiniCPM-o-2_6 on Fig. 3a)

7. Conclusions

We have shown that general-purpose LLMs exhibit certain limitations in their logical reasoning abilities and while they can generate seemingly elaborate answers they struggle with reasoning. Furthermore this problem is amplified by processing images, showcasing a lack of ability to correctly discern and distinguish precisely shapes and patterns in complex images. To answer our initial question RQ1, for now multimodal LLMs, open source and closed source, are not able to consistently respond correctly to reasoning questions based on images and text. Regarding RQ2, while ChatGPT-4o outperforms open-source models, it still exhibit some hallucinations, and lacks the ability to correctly describe all intricacies of a complex image. Smaller open-source models, on the condition of having been trained on Prolog code, struggle with this task, producing only basic clauses or rules, while larger models, such as Molmo-72B-0924 and its quantized variant, demonstrated a stronger capability by being able to generate more structured and correct logic code. Finally with respect to RQ3 we can say that currently closed-source models still perform better than open-source ones, thanks to the larger number of parameters and more extensive training.

Acknowledgements

This work has been partially supported by Spoke 1 "FutureHPC & BigData" of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC) funded by MUR Missione 4 - Next Generation EU (NGEU), by the Italian Ministry of Industrial Development (MISE) under project EI-TWIN n. F/310168/05/X56 CUP B29J24000680005. FR is a member of the Gruppo Nazionale Calcolo Scientifico – Istituto Nazionale di Alta Matematica (GNCS-INdAM).

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT in order to: Improve writing style and Paraphrase and reword. Further, the author(s) used ChatGPT-4o, Molmo-72B-0924, Molmo-7B-D-092 and MiniCPM-o-2_6 for generating the Prolog code snippets that appear as quotations. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, A. Mian, A comprehensive overview of large language models, arXiv preprint arXiv:2307.06435 (2023).

[2] T. Gunter, Z. Wang, C. Wang, R. Pang, A. Narayanan, A. Zhang, B. Zhang, C. Chen, C.-C. Chiu, D. Qiu, et al., Apple intelligence foundation language models, arXiv preprint arXiv:2407.21075 (2024).

[3] R. Hazra, G. Venturato, P. Z. D. Martires, L. De Raedt, Can large language models reason? a characterization via 3-SAT, arXiv preprint arXiv:2408.07215 (2024). URL: https://arxiv.org/pdf/2408.07215.

[4] I. Mirzadeh, K. Alizadeh, H. Shahrokhi, O. Tuzel, S. Bengio, M. Farajtabar, GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models, arXiv preprint arXiv:2410.05229 (2024).

[5] A. Creswell, M. Shanahan, I. Higgins, Selection-inference: Exploiting large language models for interpretable logical reasoning, arXiv preprint arXiv:2205.09712 (2022).

[6] 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 (2022) 24824–24837.

[7] M. A. Ferrag, N. Tihanyi, M. Debbah, Reasoning beyond limits: Advances and open problems for LLMs, arXiv preprint arXiv:2503.22732 (2025).
[8] W. Ji, W. Yuan, E. Getzen, K. Cho, M. I. Jordan, S. Mei, J. E. Weston, W. J. Su, J. Xu, L. Zhang, An overview of large language models for statisticians, arXiv preprint arXiv:2502.17814 (2025).
[9] A. Patil, Advancing reasoning in large language models: Promising methods and approaches, arXiv preprint arXiv:2502.03671 (2025).
[10] T. Xue, Z. Wang, Z. Wang, C. Han, P. Yu, H. Ji, Rcot: Detecting and rectifying factual inconsistency in reasoning by reversing chain-of-thought, arXiv preprint arXiv:2305.11499 (2023).
[11] P. Chhikara, Mind the confidence gap: Overconfidence, calibration, and distractor effects in large language models, arXiv preprint arXiv:2502.11028 (2025).
[12] Z. Dehghanighobadi, A. Fischer, M. B. Zafar, Can LLMs explain themselves counterfactually?, arXiv preprint arXiv:2502.18156 (2025).
[13] J. Oh, M. Jeong, J. Ko, S.-Y. Yun, When debate fails: Bias reinforcement in large language models, arXiv preprint arXiv:2503.16814 (2025).
[14] D. Machlab, R. Battle, LLM in-context recall is prompt dependent, arXiv preprint arXiv:2404.08865 (2024).
[15] B. Y. Lin, R. L. Bras, K. Richardson, A. Sabharwal, R. Poovendran, P. Clark, Y. Choi, Zebralogic: On the scaling limits of LLMs for logical reasoning, arXiv preprint arXiv:2502.01100 (2025).
[16] J. Boye, B. Moell, Large language models and mathematical reasoning failures, arXiv preprint arXiv:2502.11574 (2025).
[17] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, et al., A survey on evaluation of large language models, ACM transactions on intelligent systems and technology 15 (2024) 1–45.
[18] G. Marra, S. Dumančić, R. Manhaeve, L. De Raedt, From statistical relational to neurosymbolic artificial intelligence: A survey, Artificial Intelligence 328 (2024) 104062. URL: https://www.sciencedirect.com/science/article/pii/S0004370223002084. doi:https://doi.org/10.1016/j.artint.2023.104062.
[19] H. Zhang, P.-N. Kung, M. Yoshida, G. Van den Broeck, N. Peng, Adaptable logical control for large language models, Advances in Neural Information Processing Systems 37 (2024) 115563–115587.
[20] D. Roth, On the hardness of approximate reasoning, Artificial intelligence 82 (1996) 273–302.
[21] H. Zhang, J. Huang, Z. Li, M. Naik, E. Xing, Improved logical reasoning of language models via differentiable symbolic programming, arXiv preprint arXiv:2305.03742 (2023).
[22] M. Nye, M. Tessler, J. Tenenbaum, B. M. Lake, Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning, Advances in Neural Information Processing Systems 34 (2021) 25192–25204.
[23] A. Kalyanpur, K. K. Saravanakumar, V. Barres, J. Chu-Carroll, D. Melville, D. Ferrucci, LLM-ARC: Enhancing llms with an automated reasoning critic, arXiv preprint arXiv:2406.17663 (2024).
[24] L. Pan, A. Albalak, X. Wang, W. Y. Wang, Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning, arXiv preprint arXiv:2305.12295 (2023).
[25] X. Ye, Q. Chen, I. Dillig, G. Durrett, SatLM: Satisfiability-aided language models using declarative prompting, Advances in Neural Information Processing Systems 36 (2023) 45548–45580.
[26] D. Cunnington, M. Law, J. Lobo, A. Russo, The role of foundation models in neuro-symbolic learning and reasoning, in: International Conference on Neural-Symbolic Learning and Reasoning, Springer, 2024, pp. 84–100.
[27] Y. Jiao, L. De Raedt, G. Marra, Valid text-to-SQL generation with unification-based DeepStochLog, in: International Conference on Neural-Symbolic Learning and Reasoning, Springer, 2024, pp. 312–330.
[28] I. Kareem, K. Gallagher, M. Borroto, F. Ricca, A. Russo, Using learning from answer sets for robust question answering with LLM, in: International Conference on Logic Programming and Nonmonotonic Reasoning, Springer, 2024, pp. 112–125.
[29] A. Creswell, M. Shanahan, Faithful reasoning using large language models, arXiv preprint arXiv:2208.14271 (2022).

[30] C. Zhang, F. Gao, B. Jia, Y. Zhu, S.-C. Zhu, RAVEN: A Dataset for Relational and Analogical Visual rEasoNing, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5312–5322. doi:10.1109/CVPR.2019.00546.

[31] D. Azzolini, A. Bizzarri, E. Gentili, F. Riguzzi, Tic tac toe dataset, 2024. URL: https://github.com/bizzarriA/TicTacToeDS/.

[32] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al., GPT-4o system card, arXiv preprint arXiv:2410.21276 (2024).

[33] J. Wang, Z. Yang, X. Hu, L. Li, K. Lin, Z. Gan, Z. Liu, C. Liu, L. Wang, GIT: A generative image-to-text transformer for vision and language, arXiv preprint arXiv:2205.14100 (2022).

[34] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in: International conference on machine learning, PmLR, 2021, pp. 8748–8763.

[35] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, Q. V. Le, Finetuned language models are zero-shot learners, in: International Conference on Learning Representations, 2022. URL: https://openreview.net/forum?id=gEZrGCozdqR.

[36] Z. Wu, X. Chen, Z. Pan, X. Liu, W. Liu, D. Dai, H. Gao, Y. Ma, C. Wu, B. Wang, Z. Xie, Y. Wu, K. Hu, J. Wang, Y. Sun, Y. Li, Y. Piao, K. Guan, A. Liu, X. Xie, Y. You, K. Dong, X. Yu, H. Zhang, L. Zhao, Y. Wang, C. Ruan, DeepSeek-VL2: Mixture-of-experts vision-language models for advanced multimodal understanding, 2024. arXiv:2412.10302.

[37] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, J. Dean, Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, arXiv preprint arXiv:1701.06538 (2017).

[38] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini, J. Lu, T. Anderson, E. Bransom, K. Ehsani, H. Ngo, Y. Chen, A. Patel, M. Yatskar, C. Callison-Burch, A. Head, R. Hendrix, F. Bastani, E. VanderBilt, N. Lambert, Y. Chou, A. Chheda, J. Sparks, S. Skjonsberg, M. Schmitz, A. Sarnat, B. Bischoff, P. Walsh, C. Newell, P. Wolters, T. Gupta, K.-H. Zeng, J. Borchardt, D. Groeneveld, J. Dumas, C. Nam, S. Lebrecht, C. Wittlif, C. Schoenick, O. Michel, R. Krishna, L. Weihs, N. A. Smith, H. Hajishirzi, R. Girshick, A. Farhadi, A. Kembhavi, Molmo and PixMo: Open weights and open data for state-of-the-art multimodal models, arXiv preprint arXiv:2409.17146 (2024).

[39] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang, G. Dong, H. Wei, H. Lin, J. Tang, J. Wang, J. Yang, J. Tu, J. Zhang, J. Ma, J. Xu, J. Zhou, J. Bai, J. He, J. Lin, K. Dang, K. Lu, K. Chen, K. Yang, M. Li, M. Xue, N. Ni, P. Zhang, P. Wang, R. Peng, R. Men, R. Gao, R. Lin, S. Wang, S. Bai, S. Tan, T. Zhu, T. Li, T. Liu, W. Ge, X. Deng, X. Zhou, X. Ren, X. Zhang, X. Wei, X. Ren, Y. Fan, Y. Yao, Y. Zhang, Y. Wan, Y. Chu, Y. Liu, Z. Cui, Z. Zhang, Z. Fan, Qwen2 technical report, arXiv preprint arXiv:2407.10671 (2024).

[40] D. Groeneveld, I. Beltagy, P. Walsh, A. Bhagia, R. Kinney, O. Tafjord, A. H. Jha, H. Ivison, I. Magnusson, Y. Wang, S. Arora, D. Atkinson, R. Authur, K. Chandu, A. Cohan, J. Dumas, Y. Elazar, Y. Gu, J. Hessel, T. Khot, W. Merrill, J. Morrison, N. Muennighoff, A. Naik, C. Nam, M. E. Peters, V. Pyatkin, A. Ravichander, D. Schwenk, S. Shah, W. Smith, N. Subramani, M. Wortsman, P. Dasigi, N. Lambert, K. Richardson, J. Dodge, K. Lo, L. Soldaini, N. A. Smith, H. Hajishirzi, OLMo: Accelerating the science of language models, Preprint (2024).

[41] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang, J. Ji, Z. Yang, L. Zhao, X. Song, J. Xu, B. Xu, J. Li, Y. Dong, M. Ding, J. Tang, CogVLM: Visual expert for pretrained language models, 2023. arXiv:2311.03079.

[42] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, ICLR (2021).

[43] N. Shazeer, GLU variants improve transformer, arXiv preprint arXiv:2002.05202 (2020).

[44] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, E. P. Xing, Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality, 2023. URL: https://lmsys.org/blog/2023-03-30-vicuna/.

[45] Y. Yao, T. Yu, A. Zhang, C. Wang, J. Cui, H. Zhu, T. Cai, H. Li, W. Zhao, Z. He, et al., MiniCPM-V: A GPT-4V level MLLM on your phone, arXiv preprint arXiv:2408.01800 (2024).

[46] X. Zhai, B. Mustafa, A. Kolesnikov, L. Beyer, Sigmoid loss for language image pre-training, in: Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 11975–11986.

[47] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, I. Sutskever, Robust speech recognition via large-scale weak supervision, 2022. URL: https://arxiv.org/abs/2212.04356. doi:10.48550/ARXIV.2212.04356.

[48] 2Noise, Chattts, https://huggingface.co/2Noise/ChatTTS, 2025.

[49] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, Z. Qiu, Qwen2.5 technical report, arXiv preprint arXiv:2412.15115 (2024).