

# Neuro-Symbolic Programming in the Age of Foundation Models: Pitfalls and Opportunities

**Adam Stein**

**Aaditya Naik**

**Neelay Velingker**

**Mayur Naik**

**Eric Wong**

*University of Pennsylvania*

STEINAD@SEAS.UPENN.EDU

ASNAIK@SEAS.UPENN.EDU

NEELAY@SEAS.UPENN.EDU

MHNAIK@SEAS.UPENN.EDU

EXWONG@SEAS.UPENN.EDU

## Abstract

Neuro-Symbolic programming (NESY) was proposed to address challenges with training neural networks for complex reasoning tasks with the added benefits of interpretability, reliability, and efficiency. NESY methods train neural models in conjunction with symbolic reasoning, yet they face issues with scalability and training that limit them to simplistic problems. On the other hand, purely-neural foundation models can now reach state-of-the-art performance through prompting rather than training, but they are often unreliable and lack interpretability. Supplementing foundation models with reasoning programs, which we call Prompt-Symbolic (PRSY), provides a way to use these models for complex reasoning tasks. Doing so raises the question: What role does neuro-symbolic have in the age of foundation models? To explore this question, we highlight three pitfalls of NESY with respect to the compute, data, and programs. We then argue that PRSY can replace task-specific NESY training, offering opportunities for achieving the original goals of NESY without the downsides which come with training.

**Keywords:** Neurosymbolic, Programming, Foundation Models, Symbols, Training

## 1. Introduction

Foundation models pre-trained on general internet-scale data are now ubiquitous, bringing the benefits of deep learning to downstream applications across several domains (Bommasani et al., 2021). This is achieved primarily via prompting techniques, or finetuning for more niche use cases. Their success is driven by scaling up both the training data and model parameters, leading to predictable performance improvements (Kaplan et al., 2020). Even still, limitations on problems requiring complex reasoning and reliability remain (Dziri et al., 2023; Valmeekam et al., 2024). Further, these systems are fundamentally black-box and lack features like interpretability, vital for safety-critical domains such as medicine (Khan et al., 2025; Wu et al., 2024), autonomous driving (Sun et al., 2021), and aviation (SiyaeV and Jo, 2021), and their unpredictable nature raises safety concerns for their real-world deployment (Amodei et al., 2016).

Neuro-Symbolic programming (NESY) is a paradigm that moves towards a solution to these limitations by training deep neural models in conjunction with symbolic reasoning (Chaudhuri et al., 2021). Figure 1 on the left shows a setup common for NESY systems. Here, the task is split into two common subtasks, perception and reasoning. Perception tasks convert raw inputs (text, images, video, etc.) into symbols using deep neural models and the reasoning task uses a symbolic program (e.g. a Python function) to process the symbols from perception (Mao et al., 2019). Training the system end-to-end provides several benefits over traditional neural networks, including data

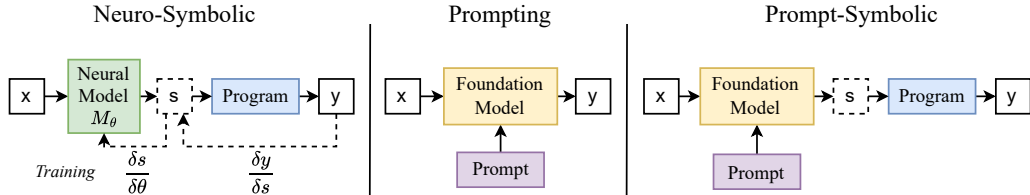


Figure 1: NESY consists of neural models whose results are fed into a program to produce the desired output (shown on the left). Training the neural models in NESY is a significant challenge due to not having supervision for the intermediate symbols  $s$ . Shown in the middle is the modern foundation model PROMPTING paradigm where a user provides a prompt and their input to a foundation model which then produces the output. On the right, we show the foundation model approach to Neuro-Symbolic which like PROMPTING does not need training and like NESY uses a symbolic component.

efficiency (using smaller datasets and less supervision), generalizability, and interpretability (the intermediate symbols can be examined and the program offers a faithful explanation of the output). Despite these benefits, NESY is often impractical due to a lack of scalability since direct supervision on the output of the neural model is not provided (Feldstein et al., 2024).

While a major challenge in NESY is learning the neural component, foundation models can now perform many tasks requiring strong input understanding without any additional training (Bommasani et al., 2021; Yue et al., 2024; Ferber et al., 2024). The common prompting setup is shown in the middle of Figure 1 where there is now just a prompt rather than the training and program used in NESY. As foundation models are trained on such large amounts of data, prompting can even offer better performance and robustness than a training-based method using less data (Bommasani et al., 2021). This raises the following question: *What role does Neuro-Symbolic have in the age of foundation models?*

To answer this, we investigate the extent to which frontier foundation models allow for achieving the benefits of NESY—program reliability and symbol interpretability—without the disadvantages that come with training. We term the replacement of neural components in NESY with foundation models as *Prompt-Symbolic* (PRSY), and show its setup in Figure 1 on the right. PRSY uses prompted foundation models to perform the perception task of extracting symbols, while a symbolic program (e.g. a Python program) is used to reason over the detected symbols.

Our experiments uncover several NESY pitfalls including unnecessarily training models when prompting is now available, overfitting to labeled datasets, and trusting a single program to provide learning signal for the correct behaviors. On the other hand, we show that PRSY provides opportunities for enabling reliability and interpretability of foundation models without the pitfalls that come with training in NESY. In light of these findings, we look towards future research on PRSY, where we highlight the problem of autonomously inferring the symbols and program as the significant remaining frontier.

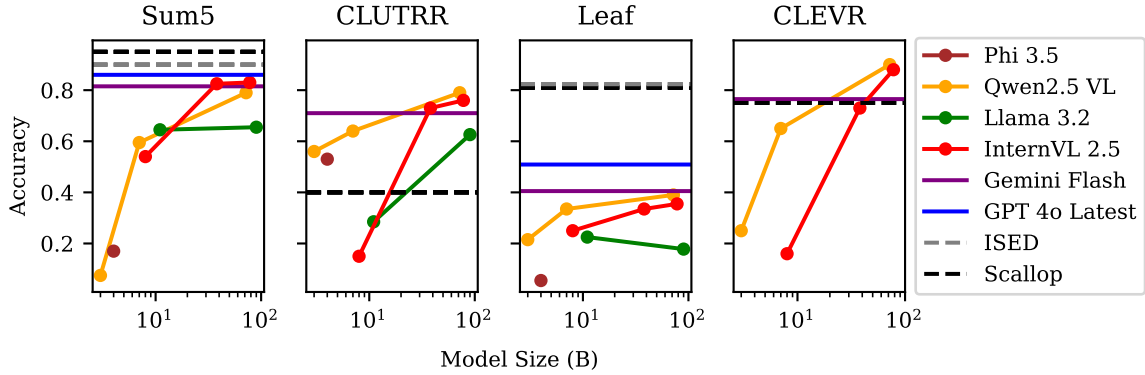


Figure 2: Performance of PRSY for four benchmarks as model size increases compared to Scallop, a baseline NESY method which trains neural models. As model size increases, the gap between NeSy training and prompting increasingly vanishes. There is still a considerable gap for the Leaf dataset which we discuss in regards to *the program pitfall* in Section 2.3. See Table 2 for full results on all five datasets.

## 2. Pitfalls

The Neuro-Symbolic paradigm enables the use of explicit programs in an end-to-end differentiable manner, which allows for learning models for reasoning tasks using less supervision, data, and compute compared to traditional deep learning techniques. However, in the age of foundation models, a well-crafted prompt often replaces the more resource-intensive training of deep neural networks. Prompt-Symbolic techniques further make many of the benefits of NESY available for foundation models. We thus observe that the NESY paradigm faces three main pitfalls in the age of foundation models: the *compute pitfall*, the *data pitfall*, and the *program pitfall*.

### 2.1. The Compute Pitfall

From its inception, NESY was proposed as a solution for tasks requiring both deep learning and complex reasoning where, formerly, knowledge that was otherwise human-specifiable was instead being learned indirectly by extensive training of large neural networks from scratch. While the NESY paradigm still required training, the models trained could be smaller and more specialized, learning concepts from less data and using less overall compute.

The proliferation of foundation models significantly changes the underlying assumptions of this paradigm. If one can replace the models from traditional NESY with foundation models that don't require additional training, they may forgo the compute required to train NESY models. It is then imperative to ask: how beneficial is it to still train NESY models?

We try and answer this question by comparing NESY techniques, Scallop (Huang et al., 2021) and ISED (Solko-Breslin et al., 2024), against PRSY instantiated with various open and proprietary foundation models. As benchmarks, we consider the Sum5 dataset (Huang et al., 2021) which asks for the sum of five MNIST (LeCun et al., 2010) handwritten digits, the HWF5 dataset (Li et al., 2020) which asks for the result of evaluating a handwritten arithmetic expression, the CLUTRR dataset (Sinha et al., 2019) which asks for the relationship between people described in text, the

CLEVR dataset (Johnson et al., 2017) which asks questions about an image containing various objects, and finally the Leaf dataset (Solko-Breslin et al., 2024; Chouhan et al., 2019) which asks for a plant’s name from an image of a leaf. For foundation models, we use the Llama 3.2 (Dubey et al., 2024), Qwen 2.5 VL (Team, 2025), InternVL 2.5 (Chen et al., 2024), and Phi 3.5 (Abdin et al., 2024) family of open models along with Gemini 2.0 Flash (Team et al., 2023) and GPT 4o (Hurst et al., 2024). See Appendix B for further information on our experimental setup.

Figure 2 shows a snapshot of four benchmarks: Sum5, CLUTRR, Leaf, and CLEVR. In each graph, Scallop’s performance is denoted by a black dashed line, whereas the solid lines indicate the performance of the foundation models using PRSY. We plot the performance of foundation models against their size (number of parameters). In all cases, the foundation models are strictly prompted, without any training or fine-tuning.

Consider the graphs for the CLUTRR and CLEVR benchmarks. In both cases, we can see that the smallest versions of the foundation models perform worse than Scallop, where the deep models are specifically trained for that particular task. However, notice that as the size of the foundation models increases, they progressively close the performance gap, with their largest versions eventually outperforming Scallop’s trained models. In the case of both Sum5 and Leaf, the largest foundation models are unable to outperform Scallop’s models, with the performance gap being more significant for Leaf. However, in both cases, the gap still narrows with scale.

These results show that foundation models have successfully learned the ability to convert raw input into a symbolic form in a general manner. Further, as the size of foundation models increases, they can replace, or come closer to matching, task-specific models trained through NESY techniques. This encapsulates our first pitfall: *spending compute on training NESY models has diminishing returns as the performance gap with PRSY shrinks with scale*. Naturally, we want to understand why there may be performance gaps for the Sum5 and Leaf benchmarks; closer examination of model behaviors over these benchmarks reveals the next two pitfalls.

## 2.2. The Data Pitfall

Examining the gap between NESY and PRSY reveals that errors in PRSY can stem from ambiguous or complex cases of identifying symbols. In several cases, the models trained via NESY predict “correct” symbols that conform to the ground truth of the dataset, even if the data contains biases or noise. This finding sheds light on the data pitfall: *NESY training on specialized datasets, as opposed to large-scale foundation model pretraining, encourages overfitting to dataset particularities*.

Consider Figure 3. We show several examples across benchmarks where the data itself is ambiguous, yet Scallop makes the correct prediction. For instance, consider the MNIST digits used in one sample from the Sum5 dataset. While the first four digits are relatively clear, the last digit is more ambiguous. Gemini predicts this digit to be a ‘1’, while Scallop predicts this as a ‘2’. While the digit appears to be closer to a ‘1’, the correct answer is ‘2’, allowing Scallop to infer the correct sum of digits. We attribute this discrepancy to the possibility that the model trained by Scallop has memorized this particular image to be a 2, while Gemini’s prediction is not biased by the peculiarities of the MNIST dataset. We see similar behavior in other benchmarks including HWF5, Leaf, and CLEVR shown in Figure 3.

We see further evidence of NESY models overfitting by seeing how it generalizes under slight distribution shifts. In Figure 4, we show NESY and PRSY performance on the Sum5 task when we replace the digits with versions from MNIST-C (Mu and Gilmer, 2019) with varying levels of noise

Dataset	Example	Prompt-symbolic (Gemini)	Neuro-symbolic (Scallop)
Sum5		8 0 3 7 1	8 0 3 7 2
HWF5		1 - 7 / 9	1 - 1 / 9
Leaf		oblong	elliptical
CLEVR	 There is a green cylinder that is the same size as the yellow metallic cube; what material is it?		

Figure 3: Examples from the five benchmarks of *the data pitfall*. All examples show cases where the PRSY method using Gemini results in an “error”. In contrast, the NESY method is “correct.” These predictions which are marked as errors from PRSY reflect cases where some of the symbols are ambiguous or hard to determine from the input. In contrast, the NESY method appears to have memorized noise and biases in the dataset to get the “correct” symbols. For example, for the leaf dataset, the leaf is folded such that it looks oblong (which is predicted by Gemini), but Scallop predicts elliptical, which is correct based on this species of leaf.

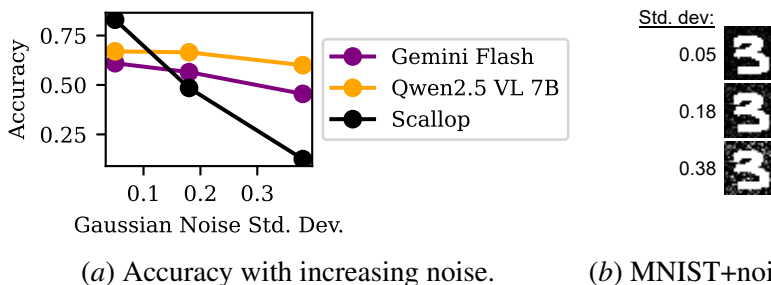


Figure 4: NeSy trained neural models memorize dataset biases rather than learn general concepts.

shown in Figure 4(b). Observe that the introduction of noise into the data in Figure 4(b) does not significantly affect the visibility of the digits in the images. Despite this, while NESY outperforms PRSY on small noise levels, adding Gaussian noise significantly drops its performance below that of PRSY. On the other hand, PRSY, which relies on general purpose foundation models, is less affected by the introduction of noise into the data.

### 2.3. The Program Pitfall

The NESY paradigm assumes the reasoning program is provided by domain experts and relies on it as a form of supervision. As such, the concepts learned by the perception models are highly






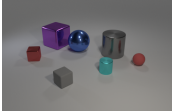
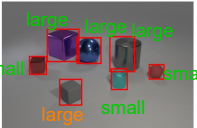
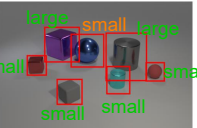
Dataset	Example	Prompt-symbolic (Gemini)	Neuro-symbolic (Scallop)
Leaf		edge: entire  shape: ovate  texture: glossy → Psidium guajava	edge: serrulate  shape: elliptical  texture: smooth → Citrus limon
CLUTRR	<div style="border: 1px solid black; padding: 5px;">[Benjamin] is the uncle of [Timothy]. [Timothy] took his son [William] out for pizza. [William] and his daughter [Christina] loved to play soccer together. Unfortunately, her sister [Connie] did not like sports. How is Benjamin related to Connie?</div>	[Benjamin] is uncle of [Timothy] [Timothy] is father of [William] [William] is father of [Christina] [William] is father of [Connie] [Christina] is sister of [Connie] → Unknown	[Benjamin] is son of [Timothy] [Timothy] is father of [William] [William] is father of [Christina] [William] is father of [Connie] [Christina] is sister of [Connie] → Uncle
CLEVR	 What shape is the small red object right of the small gray cube?	 → unknown	 → sphere

Figure 5: Examples of *the program pitfall*. All examples reflect errors of PRSY which the NESY method (Scallop) got correct. We see that the NESY method reaches the correct answer for the wrong reasons or even identifies seemingly undetectable symbols to reach the correct answer. For CLUTRR, the PRSY method extracts the correct symbols, but the symbolic program cannot deduce an answer, while the NESY method hallucinates incorrect symbols and reaches the correct answer. For CLEVR, the NESY method incorrectly identifies a blue sphere as “small” when it appears large while PRSY identifies the blue sphere as large but misjudges a cube as large which was vital to the question.

dependent on the program itself. Since this supervision is relatively weak, with ground truth not available for the perception subtasks, the neural network may learn to hallucinate, or mispredict, symbols that still result in the correct answer due to the reasoning program. This results in the program pitfall: *using programs as a component in NESY training can lead to the neural component hallucinating symbols.*

Figure 5 shows examples of the program pitfall in three datasets. Consider the Leaf dataset, where there is still a large gap between NESY and PRSY performance in Figure 2. The reasoning program in this case is a decision tree over the edge, shape, and texture of a leaf. This program is taken from instructions in a forestry database (Talhouk et al., 2015) developed for identifying leaves on the field. Here, we find that many of the PRSY errors correspond to cases where these features are extremely challenging to identify simply from a given image of a leaf.

For the leaf shown in Figure 5, Gemini identifies that it has an “entire” margin, meaning it is smooth, while the Scallop NESY method identifies it as “serrulate,” meaning it has a finely serrated edge. From just the image, it is difficult to determine whether the margin is serrulate or entire, even though identifying this is vital to the program outputting the correct answer. Even more ambiguous is the texture of the leaf, since this is heavily dependent on external factors such as the lighting conditions. While Gemini claims it is glossy, Scallop’s model predicts it as smooth. Again, making this prediction is crucial to getting the correct answer from the reasoning program. The Scallop

solution correctly predicts the final label, as computed by its human-specified program, even when the image itself lacks clear visual evidence supporting the neural predictions.

To validate that these concepts are not immediately apparent from just the image, we perform a human evaluation with 10 people hired from Prolific to determine if they agree with the outputs of Scallop’s trained neural model for the cases where PRSY with Gemini gets the wrong answer. Results are shown in Table 1. We see that a majority of respondents disagree with Scallop’s neural outputs for margin and shape classification even though they result in the correct answer. For texture classification, the inter-annotator agreement is so low that it indicates people cannot reliably determine leaf texture from the images.

Symbol Category	% Scallop Wrong	Agreement (Cohen’s Kappa)
Margin	58.8	0.32
Shape	60.0	0.34
Texture	46.7	0.05

Table 1: Human evaluation of Foundation Model errors on the Leaf classification dataset. We compare Foundation Model predictions leading to the wrong classification with the Scallop predictions which produce the correct answer. We find that Scallop leads to “symbol hallucination” since humans overall disagree with Scallop predictions even if they lead to the right answer.

We call this behavior *symbol hallucination*, since the neural model in NESY identifies symbols which do not appear present in the input, but their identification leads to the correct output from the program. We also see a similar behavior on the CLUTRR dataset in Figure 5 where, due to limitations of the reasoning program, PRSY gets the wrong answer even though it correctly identifies the symbols, while Scallop hallucinates the incorrect fact that “Benjamin is the son of Timothy,” resulting in a correct answer that was spuriously derived. For the CLEVR example, both Scallop’s model and PRSY mispredict some symbols due to misjudging object size, but the NESY method still results in the correct answer since the program happens to ignore the mispredicted object. In this case, NESY gets the right answer, but has not actually learned the desired distinction between small and large objects.

### 3. Opportunities

In light of the above pitfalls, where does Neuro-Symbolic programming hold merit today? In this section, we argue that NESY provides several opportunities for advancing the usefulness of foundation models.

#### 3.1. Program Reliability

Compared to pure foundation model prompting, using a symbolic program in a NESY or PRSY approach can improve accuracy and provide reliability. Recent results have shown that combining foundation models with explicit programs in various PRSY configurations improves performance for mathematical reasoning tasks while providing reliability and trustworthiness that were lacking through pure prompting (Lyu et al., 2023; Chen et al., 2023; Gao et al., 2023). Beyond mathematical reasoning, a PRSY approach is beneficial for any task involving symbolic computation, since

performing exact symbolic computation will always be more accurate and reliable than a neural approximation.

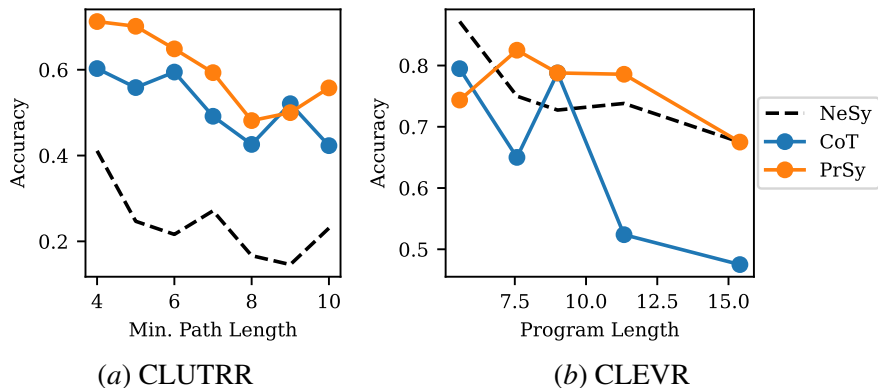


Figure 6: Performance of end-to-end prompting of Gemini-2.0-Flash compared to NeSy prompting of the same model on CLUTRR and CLEVR examples of increasing complexity. Complexity for CLUTRR is measured by the minimum number of reasoning steps needed, and complexity for CLEVR is measured by the length of the program corresponding to which answers a sample’s question.

As an example, we take the CLUTRR benchmark which asks about the relationship between two people described in a paragraph and compare the accuracy of PRSY to PROMPTING (using chain-of-thought (Wei et al., 2022)). The results shown in Figure 6(a) demonstrate that PRSY achieves consistently high accuracy with increasing question complexity while PROMPTING lags in performance. As such, PROMPTING serves as an approximation for symbolic behavior, but using a real symbolic program via PRSY yields higher accuracy. Similar behavior is shown for the CLEVR dataset in Figure 6(b) where PRSY results in higher and more stable performance than PROMPTING with chain-of-thought.

### 3.2. Symbol Interpretability

In addition to the benefits from using program execution rather than a neural approximation, the other significant benefit is that the symbol extraction step provides a means for interpretability, which was a major motivation for the emergence of NESY (Garcez et al., 2019). Now that foundation models are increasingly useful in more domains and adopted into real-world applications, this need only serves to grow. These paradigms are not orthogonal in this manner; foundation models can offer additional interpretability to NESY since large-scale pretraining is less likely to overfit to artifacts of any one dataset (Hendrycks et al., 2020).

An example of how intermediate symbols are useful for interpretability can be seen in the CLEVR example in Figure 3. In this case, PRSY results in the wrong answer of “rubber.” The nature of PRSY allows us to debug why the model output the wrong answer by investigating the intermediate symbols input to and resulting from the reasoning program. In this case, we see that the green cylinder was misidentified as rubber, since it is hard to tell the cylinder’s material. However, a prediction from pure PROMPTING would have been difficult to debug and understand due to a lack



of intermediate symbols that are interpretable and any guarantees that autoregressive explanations are faithful to themselves.

### 4. Looking Ahead

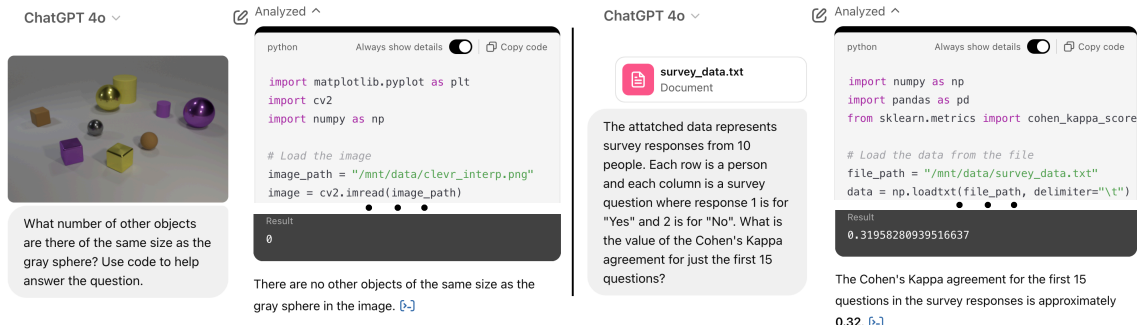


Figure 7: Example of ChatGPT code execution feature. Generated code is executed for solving the problem, but we can see in the example on the left from CLEVR that the model can produce the wrong code. For the example on the right, the model correctly calculates the Cohen’s Kappa of the attached data.

As foundation models continue to scale, the NESY pitfalls will only become more apparent, and the opportunities more important. As we demonstrate in this paper, foundation models are now highly capable for general input processing/understanding tasks which neural models were traditionally trained for in NESY. The remaining problem in NESY is no longer learning to identify symbols, but *determining which symbols and what program to use for a problem*.

We empirically demonstrate in Section 3.2 that the use of programs in a PRSY setup offers the potential for reliable and accurate symbolic reasoning, but the requirement on the program being specified by humans before inference greatly limits the practical applicability and performance of the method. As such, effectively synthesizing programs over foundation model symbols is still an open problem.

There is now growing interest in this problem with methods which mostly focus on prompting or finetuning foundation models for generating the program in a PRSY setup (Lyu et al., 2023; Chen et al., 2023; Gao et al., 2023; Pan et al., 2023). Industry has also taken interest as shown with the release of OpenAI’s Code Interpreter (Achiam et al., 2023) and Google’s Gemini code execution (Team et al., 2023) which can write and execute its generated code as well as OpenAI’s Operator (OpenAI, 2025) which writes code for performing actions on a computer. As shown in the example on the right of Figure 7, currently available PRSY tools are already useful for relatively simple data analysis tasks. However, for more complicated tasks such as CLEVR questions, the example on the left shows these methods are ineffective, potentially resulting in even worse performance than pure prompting.

## 5. Related Work

### 5.1. Neuro-Symbolic Learning Methods

For a survey on NESY methods see [Garcez et al. \(2019\)](#). NESY learning methods often consist of a neural perception component followed by symbolic reasoning ([Mao et al., 2019](#); [Yi et al., 2018](#)). There are now several frameworks for constructing such NESY setups including Deep-ProbLog ([Manhaeve et al., 2018](#)), Scallop ([Huang et al., 2021](#)), NeurASP ([Yang et al., 2020](#)), ISED ([Solko-Breslin et al., 2024](#)), and Dolphin ([Naik et al., 2024](#)). All these approaches make various assumptions regarding program differentiability, and they provide different levels of scalability.

### 5.2. Integrating Foundation Models in Neuro-Symbolic

Existing work which incorporates foundation models with NESY often uses a foundation model to generate code which is then executed ([Lyu et al., 2023](#)). These approaches either use prompting to produce explicit code ([Lyu et al., 2023](#); [Li et al., 2024b](#); [Chen et al., 2023](#); [Gao et al., 2023](#); [Hao et al., 2025](#)) or finetuning for code generation ([Gou et al., 2024](#); [Pan et al., 2023](#)). There is also work on directly finetuning foundation models for symbol extraction ([Cunnington et al., 2024](#)). Finally, NESY has also been combined with foundation models to help design new NESY datasets ([Li et al., 2020](#)) and to develop datasets for finetuning of foundation models ([Li et al., 2024a](#)).

### 5.3. Challenges in Neuro-Symbolic

Several works have recently identified challenges and misconceptions with the common NESY framing. Reasoning shortcuts, identified by [Marconato et al. \(2023\)](#), are cases where a NESY method learns to symbols with the wrong semantics, leading to poor performance on programs using the same symbols in different ways. Reasoning shortcuts are a consequence of the program pitfall. Another challenge comes from the common assumption on independence of all symbols, which often does not hold ([van Krieken et al., 2024](#)). Similarly, it is assumed that the detected symbols should display locality, or being influenced by a subset of input features ([Raman et al., 2023](#)). As observed by [Raman et al. \(2023\)](#), training in a NESY setup actually does not result in symbols with the desired locality, another instance of the program pitfall.

## 6. Conclusion

In this paper, we took a critical look at traditional NESY in the age of foundation models. NESY, as originally proposed, was meant to address the limitations of deep learning on complex reasoning problems, as well as its lack of reliability and interpretability. While addressing these problems, NESY introduced scalability and training issues which limited its effectiveness to overly simplistic domains. In the age of foundation models, where prompting alone is enough to solve many tasks without training, we highlight three pitfalls of NESY with respect to data, compute, and programs. These pitfalls are avoided by PRSY which replaces training with prompting of foundation models to offer the benefits of NESY without the downsides of training. Finally, we encourage future research on PRSY systems which infer the necessary symbols and program for solving a problem, instead of requiring them to be known in advance.

## References

- Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.
- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.
- Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.
- Swarat Chaudhuri, Kevin Ellis, Oleksandr Polozov, Rishabh Singh, Armando Solar-Lezama, Yisong Yue, et al. Neurosymbolic programming. *Foundations and Trends® in Programming Languages*, 7(3):158–243, 2021.
- Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL <https://openreview.net/forum?id=YfZ4ZPt8zd>.
- Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024.
- Siddharth Singh Chouhan, Uday Pratap Singh, Ajay Kaul, and Sanjeev Jain. A data repository of leaf images: Practice towards plant conservation with plant pathology. In *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*, pages 700–707. IEEE, 2019.
- Daniel Cunningham, Mark Law, Jorge Lobo, and Alessandra Russo. The role of foundation models in neuro-symbolic learning and reasoning. In *International Conference on Neural-Symbolic Learning and Reasoning*, pages 84–100. Springer, 2024.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36:70293–70332, 2023.

- Jonathan Feldstein, Paulius Dilkas, Vaishak Belle, and Efthymia Tsamoura. Mapping the neuro-symbolic ai landscape by architectures: A handbook on augmenting deep learning through symbolic reasoning. *arXiv preprint arXiv:2410.22077*, 2024.
- Dyke Ferber, Georg Wölflein, Isabella C Wiest, Marta Ligeró, Srividhya Sainath, Narmin Ghafari Laleh, Omar SM El Nahhas, Gustav Müller-Franzes, Dirk Jäger, Daniel Truhn, et al. In-context learning enables multimodal large language models to classify cancer pathology images. *Nature Communications*, 15(1):10104, 2024.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR, 2023.
- A Garcez, M Gori, LC Lamb, L Serafini, M Spranger, and SN Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *Journal of Applied Logics*, 6(4):611–631, 2019.
- Zhibin Gou, Zhihong Shao, Yeyun Gong, yelong shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. ToRA: A tool-integrated reasoning agent for mathematical problem solving. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=Ep0TtjVoap>.
- Yilun Hao, Yang Zhang, and Chuchu Fan. Planning anything with rigor: General-purpose zero-shot planning with LLM-based formalized programming. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=0K1OaL6XuK>.
- Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzić, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.244. URL <https://aclanthology.org/2020.acl-main.244/>.
- Jiani Huang, Ziyang Li, Binghong Chen, Karan Samel, Mayur Naik, Le Song, and Xujie Si. Scallop: From probabilistic deductive databases to scalable differentiable reasoning. *Advances in Neural Information Processing Systems*, 34:25134–25145, 2021.
- Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

- Wasif Khan, Seowung Leem, Kyle B See, Joshua K Wong, Shaoting Zhang, and Ruogu Fang. A comprehensive survey of foundation models in medicine. *IEEE Reviews in Biomedical Engineering*, 2025.
- Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, 2, 2010.
- Qing Li, Siyuan Huang, Yining Hong, Yixin Chen, Ying Nian Wu, and Song-Chun Zhu. Closed loop neural-symbolic learning via integrating neural perception, grammar parsing, and symbolic reasoning. In *International Conference on Machine Learning*, pages 5884–5894. PMLR, 2020.
- Zenan Li, Zhi Zhou, Yuan Yao, Xian Zhang, Yu-Feng Li, Chun Cao, Fan Yang, and Xiaoxing Ma. Neuro-symbolic data generation for math reasoning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a. URL <https://openreview.net/forum?id=C1cMZGLyZW>.
- Ziyang Li, Jiani Huang, Jason Liu, Felix Zhu, Eric Zhao, William Dodds, Neelay Velingker, Rajeev Alur, and Mayur Naik. Relational programming with foundational models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10635–10644, 2024b.
- Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *The 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (IJCNLP-AACL 2023)*, 2023.
- Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *Advances in neural information processing systems*, 31, 2018.
- Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=rJgMlhRctm>.
- Emanuele Marconato, Stefano Teso, Antonio Vergari, and Andrea Passerini. Not all neuro-symbolic concepts are created equal: Analysis and mitigation of reasoning shortcuts. *Advances in Neural Information Processing Systems*, 36:72507–72539, 2023.
- Norman Mu and Justin Gilmer. Mnist-c: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*, 2019.
- Aaditya Naik, Jason Liu, Claire Wang, Saikat Dutta, Mayur Naik, and Eric Wong. Dolphin: A programmable framework for scalable neurosymbolic learning. *arXiv preprint arXiv:2410.03348*, 2024.
- OpenAI. Computer-using agent: Introducing a universal interface for ai to interact with the digital world. 2025. URL <https://openai.com/index/computer-using-agent>.

- Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.248. URL <https://aclanthology.org/2023.findings-emnlp.248/>.
- Naveen Raman, Mateo Espinosa Zarlenga, Juyeon Heo, and Mateja Jamnik. Do concept bottleneck models obey locality? In *XAI in Action: Past, Present, and Future Applications*, 2023. URL <https://openreview.net/forum?id=F6RPYDUIZr>.
- Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4506–4515, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1458. URL <https://aclanthology.org/D19-1458/>.
- Aziz Siyaev and Geun-Sik Jo. Neuro-symbolic speech understanding in aircraft maintenance meta-verse. *IEEE Access*, 9:154484–154499, 2021. doi: 10.1109/ACCESS.2021.3128616.
- Alaia Solko-Breslin, Seewon Choi, Ziyang Li, Neelay Velingker, Rajeev Alur, Mayur Naik, and Eric Wong. Data-efficient learning with neural programs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=QXQY58xU25>.
- Jiankai Sun, Hao Sun, Tian Han, and Bolei Zhou. Neuro-symbolic program search for autonomous driving decision module design. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 21–30. PMLR, 16–18 Nov 2021. URL <https://proceedings.mlr.press/v155/sun21a.html>.
- S.N. Talhouk, M. Fabian, and R. Dagher. Landscape plant database, 2015. URL <https://landscapeplants.aub.edu.lb/>.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- Qwen Team. Qwen2.5-v1, January 2025. URL <https://qwenlm.github.io/blog/qwen2.5-v1/>.
- Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. Llms still can’t plan; can lrms? a preliminary evaluation of openai’s o1 on planbench. *arXiv preprint arXiv:2409.13373*, 2024.
- Emile van Krieken, Pasquale Minervini, Edoardo Ponti, and Antonio Vergari. On the independence assumption in neurosymbolic learning. In *Forty-first International Conference on Machine Learning*, 2024. URL <https://openreview.net/forum?id=S1gSrruVd4>.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yinjun Wu, Mayank Keoliya, Kan Chen, Neelay Velingker, Ziyang Li, Emily J Getzen, Qi Long, Mayur Naik, Ravi B Parikh, and Eric Wong. Discret: Synthesizing faithful explanations for treatment effect estimation. *Proceedings of machine learning research*, 235:53597, 2024.

Zhun Yang, Adam Ishay, and Joohyung Lee. Neurasp: Embracing neural networks into answer set programming. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1755–1762. ijcai.org, 2020. doi: 10.24963/IJCAI.2020/243. URL <https://doi.org/10.24963/ijcai.2020/243>.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *Advances in neural information processing systems*, 31, 2018.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multi-modal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024.

## Appendix A. Prompts

The prompt we use for the foundation models for all benchmarks takes the following form with placeholders that depend on the particular dataset.

```
System Prompt:
You are a helpful assistant.

User Prompt:
After examining the input, determine <output_description>. Here
are some examples:
Example 1:<ex1_input>This is an example of <ex1_output>.
...
<input>The input is <input_description>. Examine it and then
output just <output_description> after 'FINAL ANSWER:'. If
unsure of the answer, try to choose the best option.

Assistant:
```

For Sum5 the input description is “an image of a handwritten digit”, the output description is “the digit as an integer from 0 to 9”, and we use 5 few-shot examples.

For HWF5 the input description is “a handwritten number from 0 to 9”, the input description is “the value of the number as an integer from 0 to 9”, and we use 5 few-shot examples for the digit perception. For operator extraction the input description is “a handwritten arithmetic operator”, the input description is “the operator as a string in the set '+', '-', '\*', '/' (note that the division operator can look like a line with a dot above and below it and multiplication can look like an 'x')” and we use 4 few-shot examples.

For CLUTRR the input description is “a description of a relationship between two people and a query about the two people’s relationship”, the output description is

```
the described relationship which answers the question. Use the
pronouns to determine the people’s gender. The relationship
must be one of the following: {'brother', 'sister', 'father',
'mother', 'son', 'daughter', 'grandfather', 'grandmother', '
uncle', 'aunt', 'nephew', 'niece', 'husband', 'wife', 'brother
-in-law', 'sister-in-law', 'son-in-law', 'daughter-in-law', '
father-in-law', 'mother-in-law', 'grandson', 'granddaughter',
'unknown'}. For example, for the input 'John took his sister
Mary to the store. John is Mary’s what?' the output should be
'brother.' Output just the relationship as a word.
```

and we use 2 few-shot examples.

For CLEVR the input description is “an image of geometric objects”, the output description is

```
each object’s bounding box and attributes in the form {"bbox_2d
\": (x1, y1, x2, y2), \"attributes\": (color, shape, material,
size)}. Colors can be one of ['gray', 'green', 'blue', 'red', '
brown', 'purple', 'yellow', 'cyan'], shapes can be one of ['cube
```



```
' , 'cylinder', 'sphere'], material can be one of ['rubber', 'metal'] (it is rubber if the finish is matte and metal if shiny), and size can be one of ['small', 'large'].
```

and we use 2 few-shot examples.

For Leaf, we use three different prompts for the three networks used for perception in the NESY program. For all networks, the input description is “an image of a leaf”. For the margin network, the output description is “the classification of the leaf’s margin as one of ‘entire’, ‘indented’, ‘lobed’, ‘serrate’, ‘serrulate’, ‘undulate’”, and we use 5 few-shot examples. For the shape network, the output description is “the leaf’s shape as one of ‘elliptical’, ‘lanceolate’, ‘oblong’, ‘obovate’, ‘ovate’” and we use 9 few-shot examples. Finally, the output description for the texture network is “the classification of the leaf’s texture as one of ‘glossy’, ‘leathery’, ‘smooth’, ‘rough’” and we use 3 few-shot examples.

## Appendix B. Experiment Details

For all prompting experiments, we use greedy decoding (temperature 0) so there are no error bars for PRSY methods.

### B.1. Setup

We describe the benchmark datasets, Foundation Models, and NeSy learning baseline below.

**Datasets** We use five standard NeSy benchmarks:

- Sum5 (Huang et al., 2021): Constructed from the MNIST dataset of handwritten digits (LeCun et al., 2010). The input consists of five images of digits and the expected output is the sum of the digit values.
- HWF5 (Li et al., 2020): This dataset consists of five images creating an arithmetic expression. There are three handwritten digits from zero through nine and two handwritten operators representing addition, subtraction, division, and multiplication. The expected output is the evaluation of the expression.
- CLUTRR (Sinha et al., 2019): The input consists of natural language paragraphs describing family relationships and a question about the relationship between two people mentioned.
- CLEVR (Johnson et al., 2017): The input is an image containing various objects of different shape, size, color, and texture along with a question about the image.
- Leaf (Solko-Breslin et al., 2024; Chouhan et al., 2019): The input is an image of a leaf and the expected output is the species of the leaf.

**Models** We evaluate Foundation Model prompting as a replacement for neural network training in NeSy learning using the following Foundation Models:

- Qwen2.5 VL Instruct (3B, 7B, and 72B) (Team, 2025)
- InternVL 2.5 Instruct (8B, 38B, and 78B) (Chen et al., 2024)
- Llama 3.2 Vision Instruct (11B and 90B) (Dubey et al., 2024)
- Gemini 2.0 Flash (Team et al., 2023)

**NeSy learning baselines**

- Scallop (Huang et al., 2021): We use Scallop as a representative NeSy learning method.
- ISED (Solko-Breslin et al., 2024).

**B.2. Full NeSy Learning vs. Foundation Model Prompting Performance Gap**

The performance gap between NeSy prompting and full NeSy learning is quickly diminishing. In addition, the performance gap reduces with increasing model scale. This is shown in Figure 2. Results labelled with “—” for ISED are due to an unavailable implementation for the dataset. For GPT-4o, we only evaluate on two datasets to reduce cost. Finally, the PRSY results marked “—” for the CLEVR dataset are due to those models not supporting object bounding box generation.

Method	Sum5	HWF5	CLUTRR	CLEVR	Leaf
Scallop	$0.975 \pm 0.002$	$0.966 \pm 0.005$	$0.400 \pm 0.031$	0.750	$0.811 \pm 0.035$
ISED	$0.923 \pm 0.004$	0.023	—	—	$0.823 \pm 0.041$
Phi-3.5-vision-instruct	0.17	0.01	0.53	—	0.055
Llama-3.2-11B-Vision-Instruct	0.645	0.0	0.285	—	0.255
Llama-3.2-90B-Vision-Instruct	0.655	0.180	0.626	—	0.178
Qwen2.5-VL-3B-Instruct	0.075	0.015	0.560	0.250	0.215
Qwen2.5-VL-7B-Instruct	0.595	0.03	0.640	0.650	0.335
Qwen2.5-VL-72B-Instruct	0.790	0.250	0.790	0.900	0.390
InternVL2.5 8B	0.540	0.025	0.150	0.160	0.250
InternVL2.5 38B	0.825	0.140	0.730	0.730	0.335
InternVL2.5 78B MPO	0.830	0.000	0.760	0.880	0.405
GPT-4o	0.860	—	—	—	0.509
Gemini-2.0-Flash	0.815	0.710	0.760	0.765	0.405

Table 2: All results