Dear editor and reviewers,

We want to thank you for the feedback we have received and have strived to incorporate all of it in a revised version.

We organize our response letter in the following format. We present two sections dedicated to the two reviewers. The original reviewer's comments are in black, our responses are in blue, and pointers to the changes applied in the manuscript are in red. In the revised main text and appendices, we have colored in blue significant modifications.

The main points can be summarized as follows:

- **Validation of the use of an oracle perception.** This work focuses on the evaluation of text-only LLMs assuming an oracle perception. There exist attempts[1] that leverage vision support of multi-modal LLMs (e.g., GPT-4V) directly feeding the models with visual RPMs; however, they achieve consistently lower reasoning performance than with text-only prompting. Hence, this work provides the best possible representation to date for LLMs to maximize their performance. We are aware that this approach simplifies the RPM task as it not only provides the correct attribute values but also filters irrelevant attributes. In a very recent follow-up work[2], we tested the robustness of LLMs against uncertainties in the perception by developing a new dataset to purposefully evaluate this aspect. As expected, adding confounding attributes as well as a smoothened non-one-hot distribution degraded the LLMs' performance considerably. We improved the paper by elaborating on these observations.
- **Enhancements in ARLC performance and LLM testing.** We have strived to further improve the accuracy of ARLC. An improved training strategy yielded a notable improvement on I-RAVEN-X, where it can achieve an arithmetic accuracy of 97.9% on the largest dynamic range. As requested by Reviewer #1, we added tests with instruction-tuned Llama 3 70B, which unfortunately did not lead to an improvement, neither with nor without in-context learning. We want to emphasize that our prompt engineering efforts have already improved LLMs' state-of-the-art reasoning accuracy on the RPM task from 86.4% to 93.2%. For chain-of-thought (CoT) prompting, we would like to point to our follow-up work[2] that evaluates large reasoning models (i.e., large language models tuned to solve these reasoning tasks by exploring the space of solutions through pseudo-actions using CoT) on RPMs.

[1] M. Mitchell, A.B. Palmarini and A. Moskvichev, Comparing Humans, GPT-4, and GPT-4V On Abstraction and Reasoning Tasks, in: AAAI 2024 Workshop on "Are Large Language Models Simply Causal Parrots?", 2024.

[2] G. Camposampiero, M. Hersche, R. Wattenhofer, A. Sebastian and A. Rahimi, Can Large Reasoning Models do Analogical Reasoning under Perceptual Uncertainty?, in: International Conference on Neural-Symbolic Learning and Reasoning (NeSy), 2025.

- **Better formalizations of the space of rules**. Following the suggestions, we extended Section 2 (Datasets) to more clearly describe the basic I-RAVEN dataset, as well as our new I-RAVEN-X.

Review #1

Paper summary: The present manuscript delves into the topic of learning to reason with distributed vector-symbolic architectures. Specifically, it introduces the neuro-symbolic approach Abductive Rule Learner with Context Awareness (ARLC), and compare it experimentally against LLM on a visual reasoning task: RAVEN's progressive matrices (RPMs). In ARLC, learning the rules of RPM is interpreted as an assignment problem between VSA vectors and the elements of a fixed template. Experimental results show that ARLC outperform LLMs (LLama-3 and GPT-4), and better generalize to OOD samples. Curiously, in some LLMs in context learning does not seem to increase performance. The manuscript is an extension of a previous conference submission (Towards Learning Abductive Reasoning using VSA Distributed Representations) which introduced the ARLC method, focusing on two aspects:

i) comparison against LLMs and
ii) ii) introduction of a new RPM dataset I-RAVEN-X.

In I-RAVEN-X, the grid side and the number of categories is increased to test OOD generalization. It also provides additional technical details on the implementation of ARLC.

Strengths: Although the ARLC architecture is essentially the same as in conference submission, the comparison with LLM is a particularly relevant topic at present. The experiments effectively identifies specific weaknesses within the LLM process. The manuscript is both well-written and organized.

Thank you very much for reviewing our paper and for recognizing its strengths. We agree that the evaluation of LLMs' performance on reasoning tasks is important, and hope that our insights will positively shape the design of future intelligent systems.

Weaknesses: The experimental analysis concerning LLMs could be enhanced to ensure a more balanced and interesting comparison, particularly concerning the structure of the prompts. I do not wish, by any means, to detract from the proposed NeSy approach, which I believe absolutely interesting. However, a stronger and more robust comparison would yield a greater impact, in my opinion. Additionally, there is a lack of discussion regarding the advantages and disadvantages of the two approaches beyond "mere" accuracy.

This is an interesting point. Beyond accuracy, ARLC not only inherits advantages from symbolic methods (e.g., interpretability and programmability) but also advances efficiency and trainability. Yet, it is still tailored and trained to solve the given RPM task. In contrast, LLMs are more general but lack interpretability and require more computing resources. Combining the strengths of both methods, we see great potential in integrating ARLC---as an efficient and trainable analogical reasoning tool---with LLMs, e.g., ARLC could execute or validate reasoning steps from an LLM-based neural model.

Response: We further compare the two methods beyond accuracy in Section 6.

General Remarks - I have a few doubts regarding the experimental setting in which the LLMs were compared. First, I found it curious that for GPT-4 in-context learning seems to decrease performance. Any insights or potential explanations on this phenomenon? One possible explanation is that the ability to exploit in-context learning may be tied to instruction fine-tuning, see for instance [1]. Second, I wonder if choosing a different encoding for the features, using labels instead of numbers, could potentially result in an increase in performance. Third, I see a case here to use some chain-of-thought prompting or, more generally, prompting the LLM to generate a plausible algorithm and then applying it, which would further bias the LLM to approach the task as human would do, as proposed in the paper (see page 5, lines 1-9).

Thank you very much for proposing different prompting enhancements. First, we would like to point out that our base prompts rely on the previous state-of-the-art[3] in solving RPMs with LLMs. Our work proposes further prompt enhancements and the use of better models that lead to an improvement of 6.8 percentage points (i.e., from 86.4% to 93.2%).

We would like to comment on the various prompting techniques separately:

(1) **In-context learning (ICL)**: It is indeed interesting that in-context learning was only helpful for base Llama-3 70B in our experiments, whereas it was detrimental to (instruction-tuned) GPT-4. We followed the reviewer's suggestion and performed further experiments with instruction-tuned Llama 3 70B. We found ICL to be detrimental in instruction-tuned models too, achieving 64.6% compared to 79.2% without ICL. Recent work on LLM reasoning models[4] made a similar observation, where "Few-shot prompting consistently degrades its performance".
Reference [1], which was mentioned by the reviewer, proposes an adaptive selection and ordering of the ICL examples, based on the current task. When working in the pure symbolic domain (i.e., assuming perfect perception), we do not see major variations in difficulties among the different RPM tests, despite the different rules. Applying [1] to our task would then mainly filter out ICL examples representing the corresponding rule, which we do not find appropriate.
Moreover, reference [1] mainly showed that their filtering and ordering technique is effective in combination with instruction-tuned models. There are no experimental results comparing base and instruction-tuned models in general ICL.

(2) **Different encoding of the features**: We want to highlight that our numerical encoding of the features is based on ref[3], which achieved previous state-of-the-art results on the RPM task using LLMs. Indeed, there exist attempts[5] that tested an encoding with labels (i.e., words), however, with mediocre success (18%

[3] Hu, S. Storks, R. Lewis and J. Chai, In-Context Analogical Reasoning with Pre-Trained Language Models, ACL, 2023.
[4] DeepSeek-AI, DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, arXiv preprint arXiv.2501.12948, 2025.
[5] E. Latif, Yifan Zhou, Shuchen Guo, Yizhu Gao, Lehong Shi, M. Nyaaba, Gyeonggeon Lee, L. Zhang et al., A Systematic Assessment of OpenAI o1-Preview for Higher Order Thinking in Education, arXiv preprint arXiv:2410.21287, 2024.

accuracy). Moreover, we see the use of such labels difficult in large dynamic ranges, e.g., having 1000 different labels in I-RAVEN-X.

(3) **Chain-of-thought (CoT)**: Using CoT is definitely a viable option, as modern LLM-based reasoning models have shown to improve their performance with test-time scaling. Indeed, our recent follow-up work[6] already evaluates large reasoning models (LRMs) that leverage CoT. The work shows that LRMs still feature some (yet less pronounced) weaknesses in detecting and executing the arithmetic rules.

Response:

(1) We added a discussion on ICL and the additional experimental results on Llama-3 70B Instruct in Section 5.2.
(2) We added the reasoning on the use of numerical attribute description (instead of textual ones) in Section 3.2.
(3) We point to our chain-of-thought results presented in our follow-up work in Section 3.1.

- Providing additional details on the I-RAVEN benchmark would make the paper more self-contained and clearer to follow. In particular, I would consider providing a definition for the term constellation, and the characteristics of the chosen one (center).

We want to thank the reviewer for pointing out the missing definition of the constellation.

Response: We added a description of the RPM constellations in Section 2.1.

- In the experimental settings, which version of Llama was used, with or without instruction tuning? It seems that this use case would be more suited for instruction-tuned LLMs.

Thank you for the question. We use the Llama-3 70B base model without instruction tuning. Following your request, we experimented with Llama-3 70B Instruct. We observed the instruction-tuned models to be more verbose and had to adjust the prompt, such that the model provides us with an answer. The prompt was:

"Complete the Raven's progressive matrix. **Only provide the content of the empty panel, don't give an explanation.**"

To further improve the accuracy, we performed another round of hyperparameter tuning for the temperature. Unfortunately, the instruction-tuned model generally yielded inferior results compared to its base variant:

[6] G. Camposampiero, M. Hersche, R. Wattenhofer, A. Sebastian and A. Rahimi, Can Large Reasoning Models do Analogical Reasoning under Perceptual Uncertainty?, in: International Conference on Neural-Symbolic Learning and Reasoning (NeSy), 2025.

Model	# In-context learning examples (n)	Accuracy disentangled (3x queries)
Llama-3 Base	0	84.8%
Llama-3 Base	16	85.0%
Llama-3 Instruct	0	79.2%
Llama-3 Instruct	16	64.6%

Response: We added the additional results with instruction-tuned Llama in Section 5.2.

- Regarding handling uncertainty in ARLC, the validation of multiple solutions using a single binding and similarity computation rest on the equality at page 7, lines 11-15. However, in general, the similarity metric may not be linear, and hence the equality does not hold

This is an interesting question. Indeed, the GSBC uses the non-linear cosine similarity that impacts the approximation in Equation (2). Note, however, that ARLC's rule selection only considers relative similarities between different rule probabilities, where the approximation is sufficient.

Response: We marked Equation (2) as an approximation and added a footnote in Section 4.1 that reasons about the non-linear similarity metric.

- At page 8, line 43, the sentence “During inference, the last term of the sum (i = 3) is omitted, as the ground truth for the third row is unknown” shouldn’t be During training, ….?

Thank you for asking for clarification. Note that the ground-truth value for the last row is unknown during inference, since the RPM task is to predict the content of this panel. Hence, we omit the last term of the sum (i = 3) in the inference.

Response: We added this clarification to Section 4.3.

- I found interesting to note that, in ARLC, post-training after manual programming increases the accuracy (97.6 vs 97.2), and training without initialization (ARLClearn) further improves performance (98.4). However, ARLCprogr has higher performance OOD (I-RAVEN-X) compared to ARLClearn (5 to 10% gap depending on the setting), with ARLCprog performing better on I-RAVEN-X than I-RAVEN. How are the rules manually programmed, and is it possible that training overfits to the ID distribution? Or are there differences between I-RAVEN and I-RAVEN-X that could explain this discrepancy?

ARLC’s capability of maintaining or even improving when training on top of a pre-programmed solution is indeed very interesting. Following your request, we performed some further experiments and improvements on both the programmed and learned versions.

First, we examined the manually programmed weights for I-RAVEN and found a better programming that achieved almost perfect accuracy (99.6%). Therefore, the programmed version performs equally well on I-RAVEN and I-RAVEN-X. The rules that we program to solve I-RAVEN and I-RAVEN-X with ARLC are as follows:

I-RAVEN rules programmed in $\text{ARLC}_{\text{progr}}$.

RPM Rule	Programmed rule
constant	$(\mathbf{x}_a^1) \oslash (\mathbf{e})$
progression	$(\mathbf{x}_a^2 \otimes \mathbf{x}_a^2) \oslash (\mathbf{x}_a^1)$
arithmetic plus	$(\mathbf{x}_a^1 \otimes \mathbf{x}_a^2) \oslash (\mathbf{e})$
arithmetic minus	$(\mathbf{x}_a^1) \oslash (\mathbf{x}_a^2)$
distribute three	$(\mathbf{o}_a^1 \otimes \mathbf{o}_a^2 \otimes \mathbf{o}_a^3) \oslash (\mathbf{x}_a^1 \otimes \mathbf{x}_a^2)$

I-RAVEN-X rules programmed in $\text{ARLC}_{\text{progr}}$.

RPM Rule	Programmed rule
constant	$(\mathbf{x}_a^1) \oslash (\mathbf{e})$
progression	$(\mathbf{x}_a^9 \otimes \mathbf{x}_a^2) \oslash (\mathbf{x}_a^1)$
arithmetic plus	$(\mathbf{x}_a^1 \otimes \mathbf{x}_a^2 \otimes \mathbf{x}_a^3 \otimes \mathbf{x}_a^4 \otimes \mathbf{x}_a^5 \otimes \mathbf{x}_a^6 \otimes \mathbf{x}_a^7 \otimes \mathbf{x}_a^8 \otimes \mathbf{x}_a^9) \oslash (\mathbf{e})$
arithmetic minus	$(\mathbf{x}_a^1) \oslash (\mathbf{x}_a^2 \oslash \mathbf{x}_a^3 \oslash \mathbf{x}_a^4 \oslash \mathbf{x}_a^5 \oslash \mathbf{x}_a^6 \oslash \mathbf{x}_a^7 \oslash \mathbf{x}_a^8 \oslash \mathbf{x}_a^9)$
distribute-n	$(\mathbf{o}_a^1 \otimes \mathbf{o}_a^2 \otimes \mathbf{o}_a^3 \otimes \mathbf{o}_a^4 \otimes \mathbf{o}_a^5 \otimes \mathbf{o}_a^6 \otimes \mathbf{o}_a^7 \otimes \mathbf{o}_a^8 \otimes \mathbf{o}_a^9 \otimes \mathbf{o}_a^{10}) \oslash (\mathbf{x}_a^1 \otimes \mathbf{x}_a^2 \otimes \mathbf{x}_a^3 \otimes \mathbf{x}_a^4 \otimes \mathbf{x}_a^5 \otimes \mathbf{x}_a^6 \otimes \mathbf{x}_a^7 \otimes \mathbf{x}_a^8 \otimes \mathbf{x}_a^9)$

To program specific terms in the rule template presented in Equation (2), we simply set the corresponding assignment weight to 1 in Equation (3), while zeroing out all the others.

Second, we improved ARLC's training on I-RAVEN-X by including a more diverse training set containing all the rules. Thanks to these improvements, the best ARLC model with learned weights can now maintain an accuracy within 2% on the arithmetic task. We attribute the marginal performance degradation to the non-one-hot, continuous weights, which are amplified with larger weight spaces as tested in I-RAVEN-X.

Response: We updated the results in Tables 2, 5, and 6. Moreover, we specified the programmed weights in ARLC in Appendix D.

- Regarding future works, it would be interesting to discuss the role of recent architectures, such as TransNAR [2], that seek to promote relational reasoning without providing an explicit rule-based, interpretable representation.

Thanks for providing the reference. While our rule representation is designed to support all the rules in I-RAVEN-(X), ARLC learns the rules from demonstrations. Besides, we see great potential in integrating ARLC into other reasoning architectures, e.g., within a neuro-symbolic system that executes[7] or validates[8] reasoning steps from neural

[7] Pan, A. Albalak, X. Wang and W. Wang, Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning, EMNLP 2023.
[8] Kambhampati, K. Valmeekam, L. Guan, M. Verma, K. Stechly, S. Bhambri, L.P. Saldyt and A.B. Murthy, Position: LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks, ICML, 2024.

models (e.g., LLMs). Moreover, it would be interesting to have a tighter integration between the two systems at the embedding level [2].

Response: We added more details on the potential integration of ARLC in other neuro-symbolic systems in Section 6.

[1] Liu, Yinpeng, et al. "Let's Learn Step by Step: Enhancing In-Context Learning Ability with Curriculum Learning." arXiv preprint arXiv:2402.10738 (2024).
[2] Bounsi, Wilfried, et al. "Transformers meet Neural Algorithmic Reasoners." arXiv preprint arXiv:2406.09308 (2024).

Typos:
- Missing parenthesis at page 8, line 24 (O)
- In Table 2, I would consider adding a small description of the different ARLC versions in the caption to make the table more self-consistent.

Thank you for pointing out the typo and missing details in the table caption.

Response: We corrected the typo and updated the caption of Table 2.

Review #2 Submitted by d.tena@live.com
Recommendation: Major revision
Detail Comments This paper introduces a novel neuro-symbolic method for a prediction task based on a simplification of the Raven's Progressive Matrices (RPM) intelligence test. The symbolic descriptions of eight geometric figures are provided (of varying shape, colour, etc) and the goal is to predict the ninth figure by guessing the underlying pattern. The proposed method uses a differentiable rule-learning approach, where the attributes of the known figures are embedded in a high-dimensional vector space, and the system learns a vector transformation function that produces the prediction; where this transformation can be interpreted as a rule. The authors show that the prediction accuracy of the system surpasses state-of-the-art methods based on large language models. Overall, the article is written in good English, well-structured, and it appears to be technically sound.

Thank you for the summary and for taking your time to review our paper.

In my opinion, however, there are three important weaknesses.
First, the presentation of the paper, is often lacking in detail and intuitions, which makes it difficult to fully comprehend and evaluate the proposed approach. For example, the space of rules is defined in equations (2) and (3), but the definitions of x_i, o_j, are only given informally in lines 21-28 of page 8, and through an example – I found this insufficient to understand these definitions.

Thank you for raising this point. Indeed, the explicit definition of X and O was missing.

Response: We added the definition of X and O for the prediction of the last row in Section 4.2.

Furthermore, there is no intuition as to why equation (2) is chosen as the general shape of the rules. It would be important to expand on this and discuss the expressive power of this language; in particular, to know whether it is expressive enough to capture the rules used to generate the dataset.

The used GSBC framework allows us to represent additive and subtractive relationships using binding and unbinding operators, respectively. We find that the rules in I-RAVEN for the center constellation can be represented using the binding and unbinding operations.

I-RAVEN rules programmed in $\text{ARLC}_{\text{progr}}$.

RPM Rule	Programmed rule
`constant`	$(\mathbf{x}_a^1) \oslash (\mathbf{e})$
`progression`	$(\mathbf{x}_a^2 \otimes \mathbf{x}_a^2) \oslash (\mathbf{x}_a^1)$
`arithmetic plus`	$(\mathbf{x}_a^1 \otimes \mathbf{x}_a^2) \oslash (\mathbf{e})$
`arithmetic minus`	$(\mathbf{x}_a^1) \oslash (\mathbf{x}_a^2)$
`distribute three`	$(\mathbf{o}_a^1 \otimes \mathbf{o}_a^2 \otimes \mathbf{o}_a^3) \oslash (\mathbf{x}_a^1 \otimes \mathbf{x}_a^2)$

We experimentally demonstrate the completeness of this ruleset by achieving perfect accuracy with a manually programmed ARLC on I-RAVEN.

Response: We added a more elaborate discussion of the rule behavior in Section 4.2 and included a new Appendix D that describes the actual programmed rules.

For more examples of parts of the paper that could be clarified, please see the Detailed Comments below.

Second, the contribution of the work seems comparatively small. The abstract suggests that the main contribution of the paper is an experimental comparison between the neuro-symbolic approach ARLC and large language model-based approaches to the task described above. Reading the Aims and Scope section of the Neurosymbolic Artificial Intelligence journal, it remains uncertain whether experiment reports are regarded as a sufficient contribution.

ARLC has appeared at the International Conference on Neural-Symbolic Learning and Reasoning (NeSy) 2024. The present submission is an extended version, adding

- New comparisons to state-of-the-art LLMs on I-RAVEN
- A new reasoning benchmark (I-RAVEN-X) that tests the productivity (length generalization) and systematicity, revealing LLM's weakness in arithmetic relations
- Demonstration of ARLC's generalization from I-RAVEN to I-RAVEN-X

We believe that these contributions go clearly beyond the previously published content, as defined by the Author Guidelines of the journal.

Third, the motivation for comparing ARLC with large language models is unclear. In this context, it would appear that the space of rules to generate the patterns is known (even if the rules themselves aren't), so why not apply some standard Inductive Logic Programming (ILP) approach? In fact, it is surprising that ILP is not mentioned at all in the paper, and that no ILP systems are used in the experiments, even though it would appear that ILP methods are the most naturally suited to the given task.

Performing probabilistic reasoning in high-dimensional distributed representation is inspired by NVSA[9], which showed two orders of magnitude faster reasoning than the state-of-the-art classical symbolic approach. However, the rules in NVSA were manually programmed. ARLC addresses this limitation by learning the rules in a high-dimensional representation. We are not aware of any ILP solution in the RPM domain. Investigating its applicability and efficiency might be scope for future work.

My overall recommendation, assuming the journal is amenable to publishing work focused on experimental reporting, would be to accept a revised version of the paper that provides additional clarity and intuitions (please see list below), and develops the motivation further.

---

[9] M. Hersche, M. Zeqiri, L. Benini, A. Sebastian and A. Rahimi, A Neuro-vector-symbolic Architecture for Solving Raven's Progressive Matrices, Nature Machine Intelligence 5(4), 2023.

Thank you for your positive review. We strive to address all your concerns in the revised version of the paper.

Detailed Comments:
--I am also confused about what “constellation” means. In particular, what are 2x2, 3x3, and center constellations? Without explaining this, the task description is quite difficult to follow. It is also hard to evaluate the article’s decision to focus on the center evaluation only.

Thanks for the question. The RPM panels contain one or multiple objects, depending on the chosen constellation. For instance, panels in the center constellation contain only one object, whereas panels in the 2x2 constellation contain up to four objects. We limit ourselves to the center constellation as it already stresses the fundamental reasoning capabilities of LLMs, as empirically shown in Section 5. Going to constellations with multiple objects can make the problem easier, as shown in related arts[10].

Response: We added a more elaborate description of the RPM constellations in Section 2.1.

--I am missing a discussion about whether the original goals of the test are lost when the images are translated to symbolic descriptions. The symbolic description could be seen as a simplification of the task, since it describes explicitly the attributes of the figures that are relevant to the task (in contrast to the original task, where no such list of attributes is provided explicitly; and in fact figuring which are the relevant attributes in a non-trivial reasoning task)

This is an interesting point. This work aims to evaluate the abstract reasoning capabilities of LLMs and to compare them against neuro-symbolic methods. As mentioned in the introduction, a related work[11] has shown inferior performance of multi-modal LLMs when being queried with visual instead of textual descriptions. To provide the optimal conditions for LLMs, we hence provide textual descriptions assuming an oracle perception.

Indeed, having an oracle perception simplifies the RPM task. As the reviewer recognizes, it not only provides the correct attributes but also filters irrelevant attributes. In a follow-up work[12], we tested the robustness of LLMs against simulated uncertainties in the perception. As expected, adding confounding attributes as well as a smoothened non-one-hot distribution degraded the LLMs’ performance considerably.

---

[10] X. Hu, S. Storks, R. Lewis and J. Chai, In-Context Analogical Reasoning with Pre-Trained Language Models, ACL, 2023.
[11] M. Mitchell, A.B. Palmarini and A. Moskvichev, Comparing Humans, GPT-4, and GPT-4V On Abstraction and Reasoning Tasks, in: AAAI 2024 Workshop on ”Are Large Language Models Simply Causal Parrots?”, 2024.
[12] G. Camposampiero, M. Hersche, R. Wattenhofer, A. Sebastian and A. Rahimi, Can Large Reasoning Models do Analogical Reasoning under Perceptual Uncertainty?, in: International Conference on Neural-Symbolic Learning and Reasoning (NeSy), 2025.

Response: We added a discussion on the perception in Section 2.1.

--To make the paper self-contained and understand the prediction task, it would be important to explain the four rules used to generate the figures in the right column of the matrices. The text simply names them as “constant”, “progression”, “arithmetic”, and “distribute three”. One can later deduce them from the definition of the 3x10 extension of the RAVEN problem, but it would be very helpful to describe them explicitly.

Thanks for the input. We added an explanation of the rules already in the I-RAVEN section.

Response: We added basic I-RAVEN rule explanations in Section 2.1.

--“Page 3, line 31: what is an “attribute bisection tree”. Also, what is the “context matrix generation”?

The original RAVEN dataset had a flaw in the generation of the answer set. Each distractor in the answer set (i.e., a wrong answer candidate) was generated by randomly altering one attribute of the correct answer. As a result, one could predict the correct answer by taking the mode of the answer candidates *without looking at the context matrix,* therefore bypassing the actual reasoning task. As a remedy, I-RAVEN presented a method, the attribute bisection tree, that generates unbiased answers that are well balanced.

In I-RAVEN-X, the context matrix is of size 3xg, where g is the number of columns. The task is to predict the content of the missing element at the bottom left (i.e., at position (3, g)).

Response: We added this more elaborate explanation in Section 2.2.

--The number of attributes should be clarified, in addition to the number of possible values per attribute (dubbed “m” in the text, if I understand correctly).

Thanks for pointing this out. I-RAVEN-X maintains I-RAVEN’s four rules and three attributes.

Response: We stated explicitly that I-RAVEN-X uses four rules and three attributes.

--I do not understand why the RAVEN dataset is extended with additional columns. The article mentions that this is done to test “scalability” of the approach, but I am not sure whether this is a relevant concept. In this task, having more columns means having a larger number of examples, which could actually help discriminate better between patterns used to generate the sequence.

This is a very interesting point. Increasing the number of columns can simplify the detection and execution of some of the rules, e.g., the constant or progression rule.

However, for other rules, more columns make the task more difficult. We empirically demonstrate the increased difficulty for the arithmetic rule, where the accuracy drops below 10% for GPT-4 and Llama-3 70B.

--My lack of familiarity with "vector-symbolic architectures" may prevent a reader from understanding well the approach. It would be helpful to explain what are the binding, unbinding, and bundling operations.

Bundling ($\oplus$) is a similarity-preserving operation that creates a superposition of the operands; that is, the resulting vector will have a high similarity with the two operands. Binding ($\otimes$), on the other hand, is an operation that allows to bind a vector (value) to another vector (key) and does not preserve similarities; it usually allows an inverse operation, called unbinding ($\oslash$). The specific realizations of the bundling, binding, and vector space constitute the main difference between members of the VSA family. The specific implantation used in this work is shown in Table 1.

Response: We added the explanation of the binding, unbinding, and bundling operations in Section 4.1.

--It would also be helpful to give some intuition for why encoding into VSA is a good idea. Why use blocks?

As elaborated in the example in Section 4.1, encoding probability mass functions (PMFs) into VSA allows the validation of multiple solutions using only a single binding and similarity computation. Our previous work[13] has shown that this approach can reduce the computation time for performing probabilistic abductive reasoning by two orders of magnitude.

Moreover, the same work[8] showed that binary generalized sparse binary block codes (GSBCs) feature a better retrieval accuracy when encoding PMFs, compared to other alternatives such as Fourier holographic reduced representations (FHRR).

Response: We added the justification of using binary GSBCs in Section 4.1.

--Why have two blocks of six coefficients in Equation (2)?

Good question! The rule template with two blocks of six coefficients is sufficient to represent all the RPM rules in I-RAVEN. As described in Section 4.5, we increase the number of terms from 12 to 22 in I-RAVEN-X to account for the larger grid size. In general, one could have arbitrarily large blocks, as each of the elements can represent the identity. However, this will make the training more difficult.

Our new Appendix D shows the preprogrammed rules for I-RAVEN and I-RAVEN-X, which achieve perfect accuracy.

---

[13] M. Hersche, M. Zeqiri, L. Benini, A. Sebastian and A. Rahimi, A Neuro-vector-symbolic Architecture for Solving Raven's Progressive Matrices, Nature Machine Intelligence 5(4), 2023.

I-RAVEN rules programmed in $\text{ARLC}_{\text{progr}}$.

RPM Rule	Programmed rule
`constant`	$(\mathbf{x}_a^1) \oslash (\mathbf{e})$
`progression`	$(\mathbf{x}_a^2 \otimes \mathbf{x}_a^2) \oslash (\mathbf{x}_a^1)$
`arithmetic plus`	$(\mathbf{x}_a^1 \otimes \mathbf{x}_a^2) \oslash (\mathbf{e})$
`arithmetic minus`	$(\mathbf{x}_a^1) \oslash (\mathbf{x}_a^2)$
`distribute three`	$(\mathbf{o}_a^1 \otimes \mathbf{o}_a^2 \otimes \mathbf{o}_a^3) \oslash (\mathbf{x}_a^1 \otimes \mathbf{x}_a^2)$

I-RAVEN-X rules programmed in $\text{ARLC}_{\text{progr}}$.

RPM Rule	Programmed rule
`constant`	$(\mathbf{x}_a^1) \oslash (\mathbf{e})$
`progression`	$(\mathbf{x}_a^9 \otimes \mathbf{x}_a^2) \oslash (\mathbf{x}_a^1)$
`arithmetic plus`	$(\mathbf{x}_a^1 \otimes \mathbf{x}_a^2 \otimes \mathbf{x}_a^3 \otimes \mathbf{x}_a^4 \otimes \mathbf{x}_a^5 \otimes \mathbf{x}_a^6 \otimes \mathbf{x}_a^7 \otimes \mathbf{x}_a^8 \otimes \mathbf{x}_a^9) \oslash (\mathbf{e})$
`arithmetic minus`	$(\mathbf{x}_a^1) \oslash (\mathbf{x}_a^2 \oslash \mathbf{x}_a^3 \oslash \mathbf{x}_a^4 \oslash \mathbf{x}_a^5 \oslash \mathbf{x}_a^6 \oslash \mathbf{x}_a^7 \oslash \mathbf{x}_a^8 \oslash \mathbf{x}_a^9)$
`distribute-n`	$(\mathbf{o}_a^1 \otimes \mathbf{o}_a^2 \otimes \mathbf{o}_a^3 \otimes \mathbf{o}_a^4 \otimes \mathbf{o}_a^5 \otimes \mathbf{o}_a^6 \otimes \mathbf{o}_a^7 \otimes \mathbf{o}_a^8 \otimes \mathbf{o}_a^9 \otimes \mathbf{o}_a^{10}) \oslash (\mathbf{x}_a^1 \otimes \mathbf{x}_a^2 \otimes \mathbf{x}_a^3 \otimes \mathbf{x}_a^4 \otimes \mathbf{x}_a^5 \otimes \mathbf{x}_a^6 \otimes \mathbf{x}_a^7 \otimes \mathbf{x}_a^8 \otimes \mathbf{x}_a^9)$

Response: We added a new Appendix D with programmed ARLC formulations.

--Section 4.3 talks about “all rules” – but how is this quantified?

ARLC learns a set of R different rules with rule-specific weights. As in Learn-VRG, we set the number of rules to R=5. A single set of rules is instantiated and shared between all RPM attributes.

Response: We added the clarification regarding the rule set in Section 4.3.