Systematic Abductive Reasoning via Diverse Relation Representations in Vector-Symbolic Architecture

Zhong-Hua Sun, Ru-Yuan Zhang, Zonglei Zhen, Da-Hui Wang, Yong-Jie Li, *Senior Member, IEEE*, Xiaohong Wan, and Hongzhi You

***Abstract*—In abstract visual reasoning, monolithic deep learning models suffer from limited interpretability and generalization, while existing neuro-symbolic approaches fall short in capturing the diversity and systematicity of attribute and relation representations. To address these challenges, we propose a systematic abductive reasoning model with diverse relation representations (Rel-SAR) in vector-symbolic architecture (VSA) to solve Raven's progressive matrices (RPM). To derive attribute representations with symbolic reasoning potential, we introduce not only various types of atomic high-dimensional (HD) encodings that capture numeric, periodic, and logical semantics, but also the structured HD representation (SHDR) for the overall grid component. For systematic reasoning, we further propose novel numerical and logical relation functions and perform rule abduction and execution in a unified framework built upon these relation representations. Experimental results demonstrate that Rel-SAR achieves significant performance improvements on RPM tasks. By synergistically combining HD attribute representations with symbolic reasoning, Rel-SAR enables systematic abductive reasoning with both interpretable and computable semantics.**

***Index Terms*—Abstract visual reasoning, relation representation, vector-symbolic architecture (VSA).**

I. Introduction

RAVEN'S progressive matrices (RPM) are a family of psychological intelligence tests widely used for the assessment of abstract reasoning [1]. From a cognitive psychology perspective, abstract visual reasoning in RPM tests involves constructing high-level representations from images and deriving potential relations from these representations [1], [2]. Endowing artificial intelligence with such capabilities is now regarded as a crucial step toward achieving human-level intelligence. Despite notable progress, many recent monolithic deep learning approaches that entangle perception and reasoning within a single neural network [3], [4], [5], [6], [7], [8] suffer from inherent limitations, such as poor interpretability, limited robustness and generalization, and difficulties in module reuse [9]. Neuro-symbolic architectures (NSAs), which integrate neural visual perception with explicit symbolic reasoning, provide a promising paradigm for addressing these challenges and achieving human-level interpretability and generalization [9], [10], [11].

In NSAs, Marcus argues that symbol manipulation in cognition relies on representing relations among variables [10]. To address the challenges of abstract reasoning, NSA must therefore be equipped with concept (or attribute) representations with meaningful semantics and relation representations built upon them. Concept or attribute representations serve as the instantiations of variables and should possess rich cognitive–computational properties, encompassing diverse categories and features [12], [13], [14], [15]. These representations may correspond to definitional concepts of objects—such as *type*, *size*, *color*, and *number*—or to graded representations encoding specific attribute values of objects [7], [16]. Building on these representations, relation representations exploit their underlying computational properties to encode structured dependencies among multiple concepts or attributes in a form that supports explicit manipulation and systematic reasoning [17].

Nevertheless, incomplete concept (or attribute) and relation representations remain a critical bottleneck for achieving systematic abduction and execution in RPM tasks. Recent NSA models such as PrAE [9], the ALANS learner [18], and neuro-vector-symbolic architecture (NVSA) [11] suffer from limited diversity in attribute representations within their neural perception frontends and in template-based relation representations within their reasoning backends, which in turn constrains the reasoning capacity required for abstract visual reasoning. Although the ALANS learner introduces learnable rule operators within an abstract algebraic structure [18], and the ARLC model adopts a more expressive VSA-based rule template operating in the rule parameter space [19], both models improve interpretability and generalizability but

Received 19 September 2025; revised 11 February 2026; accepted 12 April 2026. Date of publication 22 April 2026; date of current version 7 October 2026. This work was supported in part by the National Natural Science Foundation of China under Grant 32571280, Grant 32171094, Grant 32441102, and Grant 32571230; in part by the National Science and Technology Innovation 2030-Major Projects under Grant 2021ZD0203700-1; in part by Sichuan Science and Technology Program under Grant 2024NSFSC0505; and in part by Beijing Natural Science Foundation under Grant L247010. *(Corresponding author: Hongzhi You.)*

Zhong-Hua Sun, Yong-Jie Li, and Hongzhi You are with the MOE Key Laboratory for Neuroinformation, School of Life Science and Technology, China-Cuba Belt and Road Joint Laboratory on Neurotechnology and Brain-Apparatus Communication, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: hongzhi-you@uestc.edu.cn).

Ru-Yuan Zhang is with the School of Psychological and Cognitive Sciences, Beijing Key Laboratory of Behavior and Mental Health, IDG/McGovern Institute for Brain Research, Key Laboratory of Machine Perception (Ministry of Education), Peking University, Beijing 100871, China.

Zonglei Zhen is with the Faculty of Psychology, Beijing Normal University, Beijing 100875, China.

Da-Hui Wang is with the School of Systems Science and the State Key Laboratory of Cognitive Neuroscience and Learning, Beijing Normal University, Beijing 100875, China.

Xiaohong Wan is with the State Key Laboratory of Cognitive Neuroscience and Learning, Beijing Normal University, Beijing 100875, China.

This article has supplementary downloadable material available at https://doi.org/10.1109/TNNLS.2026.3684958, provided by the authors.

Digital Object Identifier 10.1109/TNNLS.2026.3684958

2162-237X © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

still fall short of capturing the diversity and systematicity of attribute and relation representations.

Recent theoretical and computational advances in cognitive psychology and allied fields suggest that high-dimensional (HD) vector spaces can support the rich cognitive–computational properties, and that concepts in human cognition are likely instantiated in such vector-based representations [15]. Vector symbolic architectures (VSAs), a class of HD distributed representations, provide an expressive and principled framework in this regard by supporting algebraic operations for both numerical computation and structured symbolic representations [11], [19], [20], [21], [22], [23]. Motivated by these insights, this article aims to exploit the capabilities of VSAs to construct diverse attribute and relation representations within a unified vector-symbolic space, and to leverage this diversity to improve interpretability and strengthen systematic abductive reasoning in RPM tasks.

To achieve these objectives, we introduce various types of VSA-based atomic HD encoding with distinct semantic roles, covering independent symbols as well as numeric, periodic, and logical values. Since reasoning in RPM tasks involves attributes emerging from groups of objects, we further propose a structured HD representation (SHDR) that encodes the holistic perception of the nxn grid. Together, the atomic and structured encodings provide complementary attribute representations that support abductive reasoning at both the entity level and the layout level. Meanwhile, we propose numerical and logical relation functions as relation representations that operate on multiple HD attribute representations to define their underlying relations. Building on these representations, we propose a systematic abductive reasoning model with diverse relation representations (Rel-SAR) for solving RPM, inspired by the original NVSA model [11]. In Rel-SAR, visual attribute extraction and rule inference are integrated within a unified VSA-based computational framework. The model consists of a neuro-vector frontend that leverages diverse attribute representations to encode entity-level and layout-level attributes from raw RPM images, and a generic vector-symbolic backend that utilizes diverse relation representations to perform symbolic reasoning.

The main contributions of this study are as follows.

1) Diverse attribute and relation representations in VSA are introduced. Unlike rule templates tailored to individual rules, our proposed relation functions define relations among input attributes through operator powers and function outputs, enabling enhanced relation expressiveness.
2) We propose Rel-SAR for RPM, a reasoning framework grounded in the core principle that if attributes in an RPM instance follow a numerical or logical rule, then all attribute tuples share identical parameters of relation functions. The consistency between relation functions across rule abduction and execution enhances interpretability and strengthens systematic abductive reasoning.
3) Experimental results demonstrate high perceptual accuracy and state-of-the-art reasoning performance compared to existing NSA models, with particularly strong results on configurations involving positional rules. The results further confirm strong generalization, including attribute–value generalization, scalability to increasing numbers of unseen objects, and in-distribution generalization.

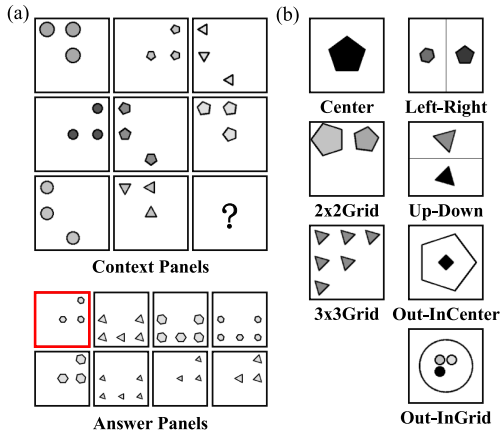


Fig. 1. Illustrations for RAVEN dataset. (a) Example of RPM test from RAVEN [16] dataset. In an RPM test, there are eight context panels and eight candidate panels. Participants are required to identify the underlying rules governing various attributes within the context panels. Subsequently, participants use these rules to infer the attributes of the missing panel (represented by "?") and choose the most appropriate option (highlighted with a red box) from the answer panels. (b) RAVEN dataset includes seven configurations: Center, 2×2 grid, 3×3 grid, left-right (L-R), up-down (U-D), Out-InCenter (O-IC) and O-IG [16]. Four types of rules, i.e., *constant*, *progression*, *arithmetic*, and *distribute three*, are applied to five attributes, i.e., *position*, *number*, *type*, *size*, and *color*, in a rowwise manner. *Constant*: the attribute values are same in a row; *progression*: the attribute value monotonically increases or decreases in a row by a value of 1 or 2; *arithmetic*: the attribute values of the first two panels and the third panel in a row follow either a rule of summation or one of subtraction, and corresponding logical rules are OR and DIFF, respectively (Supplementary material I); *distribute three* samples three values of an attribute and permutes them in different rows. The I-RAVEN dataset [7] is a variant of RAVEN, where answer sets are generated using an attribute bisection tree. RAVEN-FAIR [8] is a debiased version of RAVEN that enforces fair and statistically balanced answer choices, thereby preventing shortcut solutions.

II. Related Work

The RPM is a widely used nonverbal intelligence test designed to assess abstract reasoning. To explore the limitations of current machine learning approaches in solving abstract reasoning tasks, three automatically generated RPM-based datasets—RAVEN [16], I-RAVEN [7], and RAVEN-FAIR [8]—have been introduced (Fig. 1). Early efforts on RPM primarily employed relation network (RN) [24] and their variants [3], [6], [8], [25] to extract relations between context panels. Concurrently, CoPINet [5], MLCL [26], and DCNet [27] integrate contrastive learning in their models. Approaches like MRNet [8] and DRNet [28] aimed to enhance perception capabilities, while SRAN [7] and PredRNet [29] abstract relations using stratified models and prediction errors, respectively. In addition, several methods have focused on scene decomposition and feature disentanglement [30], [31], [32]. Although these monolithic deep learning models achieve high accuracy, they often suffer from limited interpretability and systematic generalization capabilities.

Another branch for solving RPM is based on NSAs, which explicitly distinguish between perception and reasoning. PrAE [9] employs an object CNN to generate probabilistic scene representations and uses predetermined rule templates for

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

probabilistic abduction and execution. In abstract reasoning, VSAs serve as a bridge between perception and reasoning modules by leveraging its structured distribution representations and algebraic properties. NVSA [11] projects each RPM panel into an HD vector using a trainable CNN and derives probability mass functions (PMFs) by querying an external codebook. Its reasoning backend embeds these PMFs into distributed VSA representations and performs rule abduction and execution using templates based on VSA algebraic operations. NVSA provides a differentiable and transparent implementation of probabilistic abductive reasoning by leveraging VSA representations and operators. However, its perception frontend requires searching a large external codebook, and its reasoning backend still relies on predetermined rule templates.

To address limitations in rule expressiveness, the ALANS learner utilizes learnable rule operators within an abstract algebraic structure, avoiding the need for manually defining individual rules [18]. Learn-VRF [33] focuses on reasoning by learning VSA-based rule formulations, eliminating the need for predetermined templates. ARLC [19] further enhances reasoning by incorporating context augmentation and extending rule templates to accommodate more diverse rules. However, despite these advances, both ARLC and Learn-VRF remain constrained by limited attribute representations, which prevents them from covering the full spectrum of RPM rules.

Recently, relational bottleneck methods have been proposed to promote efficient abstraction, although their ability to represent and reason over complex relations remains limited [17], [34], [35]. Nevertheless, their emphasis on explicit relation representations provides important motivation for our work. To overcome the limitations of incomplete attribute and relation representations in NSA, Rel-SAR transforms perceptual inputs into HD attribute representations with symbolic reasoning potential and abducts both numerical and logical rules within a unified framework grounded in diverse relation representations.

III. Preliminaries

A. VSA Models Utilized in This Study

VSAs are a class of computational models that utilize HD distributed representations [36]. VSA models used in this study are Fourier holographic reduced representations (FHRRs) and holographic reduced representations (HRRs) [37]. An FHRR atomic vector is a complex-valued random vector where each vector component can be considered as an angle (phasor) θ_i randomly selected independently from a uniform distribution $\mathcal{U}(-\pi, \pi]$ [36], [37]. For simplicity, FHRR atomic vectors are represented here using their phase form, $\boldsymbol{\theta} := \{\theta_i\}_{i=1}^{d}$, rather than their complex-valued representation. In this study, an HRR atomic vector $\boldsymbol{x}$ is obtained by applying the inverse fast Fourier transform (IFFT) to the complex exponential of an FHRR atomic vector $\boldsymbol{\theta}$

$$\boldsymbol{x} = \mathcal{F}^{-1}\left(e^{j\boldsymbol{\theta}}\right). \tag{1}$$

Here, $\mathcal{F}$ and $\mathcal{F}^{-1}(\cdot)$ represent the fast Fourier transform (FFT) and inverse FFT (IFFT), respectively. When the dimension d is sufficiently large, these randomly generated vectors exhibit pseudo-orthogonality, making them suitable for representing distinct symbols or concepts.

TABLE I
Basic Operations of FHRR and HRR

Operations	Impl. on FHRR	Impl. on HRR
Binding ($\boldsymbol{x} \circ \boldsymbol{y}$)	$(\boldsymbol{\theta} + \boldsymbol{\phi})\ mod\ 2\pi$	$\mathcal{F}^{-1}(\mathcal{F}(\boldsymbol{x}) \cdot \mathcal{F}(\boldsymbol{y}))$
Bundling ($\boldsymbol{x} + \boldsymbol{y}$)	$angle\left(e^{j\boldsymbol{\theta}} + e^{j\boldsymbol{\Phi}}\right)$	$\boldsymbol{x} + \boldsymbol{y}$
Inverse ($\boldsymbol{x}^{-1}$)	$(-\boldsymbol{\theta})\ mod\ 2\pi$	$\mathcal{F}^{-1}(1/\mathcal{F}(\boldsymbol{x}))$
Unbinding ($\boldsymbol{x} \oslash \boldsymbol{y}$)	$\boldsymbol{\theta^{-1}} \circ \boldsymbol{\phi}$	$\boldsymbol{x^{-1}} \circ \boldsymbol{y}$

The similarity between any two vectors is a crucial metric for evaluating the distributed representations in VSAs. In FHRR and HRR, cosine similarity is employed to measure the similarity between two vectors [36]

$$\begin{aligned} sim(\boldsymbol{\theta}, \boldsymbol{\phi}) &= \frac{1}{d}\sum_{i=1}^{d}\cos(\theta_i - \varphi_i) \\ sim(\boldsymbol{x}, \boldsymbol{y}) &= \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{|\boldsymbol{x}||\boldsymbol{y}|} \end{aligned} \tag{2}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ denote two FHRR vectors, and $\boldsymbol{x}$ and $\boldsymbol{y}$ two HRR vectors. The similarity $sim(\cdot, \cdot)$ ranges from -1 to $+1$, and the above two similarity measures are equivalent. The pseudo-orthogonality refers to the case where the similarity $sim(\cdot, \cdot) \approx 0$.

B. Basic Operations and Structured Symbolic Representations

All computations within VSAs are composed of several basic vector algebraic operations, with the primary ones being binding ($\circ$), bundling ($+$), and unbinding ($\oslash$) (Table I). The binding operation ($\circ$) is employed to form a representation of an object that contains information about the context in which it was encountered [36]. The bundling operation ($+$), also known as superposition, generates a composite HD vector that combines several lower-level representations. The unbinding operation ($\oslash$), which is the inverse of binding, extracts a constituent from the compound data structure. Binding and bundling are referred to as composition operations, while unbinding is considered a decomposition operation. All operations do not change the vector dimensionality.

Through the combination of these operations, VSAs can effectively achieve structured symbolic representations [36]. For instance, consider a scene $\boldsymbol{s}$ in which a triangle $\boldsymbol{t}$ is positioned on the left $\boldsymbol{p}_L$ and a circle $\boldsymbol{c}$ on the right $\boldsymbol{p}_R$. This scene can be represented as $\boldsymbol{s} = \boldsymbol{p}_L \circ \boldsymbol{t} + \boldsymbol{p}_R \circ \boldsymbol{c}$ by the role–filler pair [38]. In this computation, binding has a higher priority than bundling: the representations $\boldsymbol{p}_L \circ \boldsymbol{t}$ and $\boldsymbol{p}_R \circ \boldsymbol{c}$ are first computed independently, and then combined via bundling to obtain the final scene representation $\boldsymbol{s}$. By applying the inverse vector of the left position $\boldsymbol{p}_L^{-1}$ to unbind $\boldsymbol{s}$, we can retrieve an approximate vector representing the content at the left position, i.e., $\boldsymbol{p}_L^{-1} \circ \boldsymbol{s} = \boldsymbol{t} + \boldsymbol{p}_L^{-1} \circ \boldsymbol{p}_R \circ \boldsymbol{c} \approx \boldsymbol{t}$, where the term $\boldsymbol{p}_L^{-1} \circ \boldsymbol{p}_R \circ \boldsymbol{c}$ can be interpreted as crosstalk noise, caused by interference from superposed nontarget bindings [36]. Moreover, the triangle $\boldsymbol{t}$ can itself be a compositional scene, where attributes such as color and size are combined into a triangle scene in a similar manner. This decomposable, structure-sensitive, HD distributed representation has the potential to disentangle complex scenes while maintaining the advantages of traditional connectionist approaches [11].

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

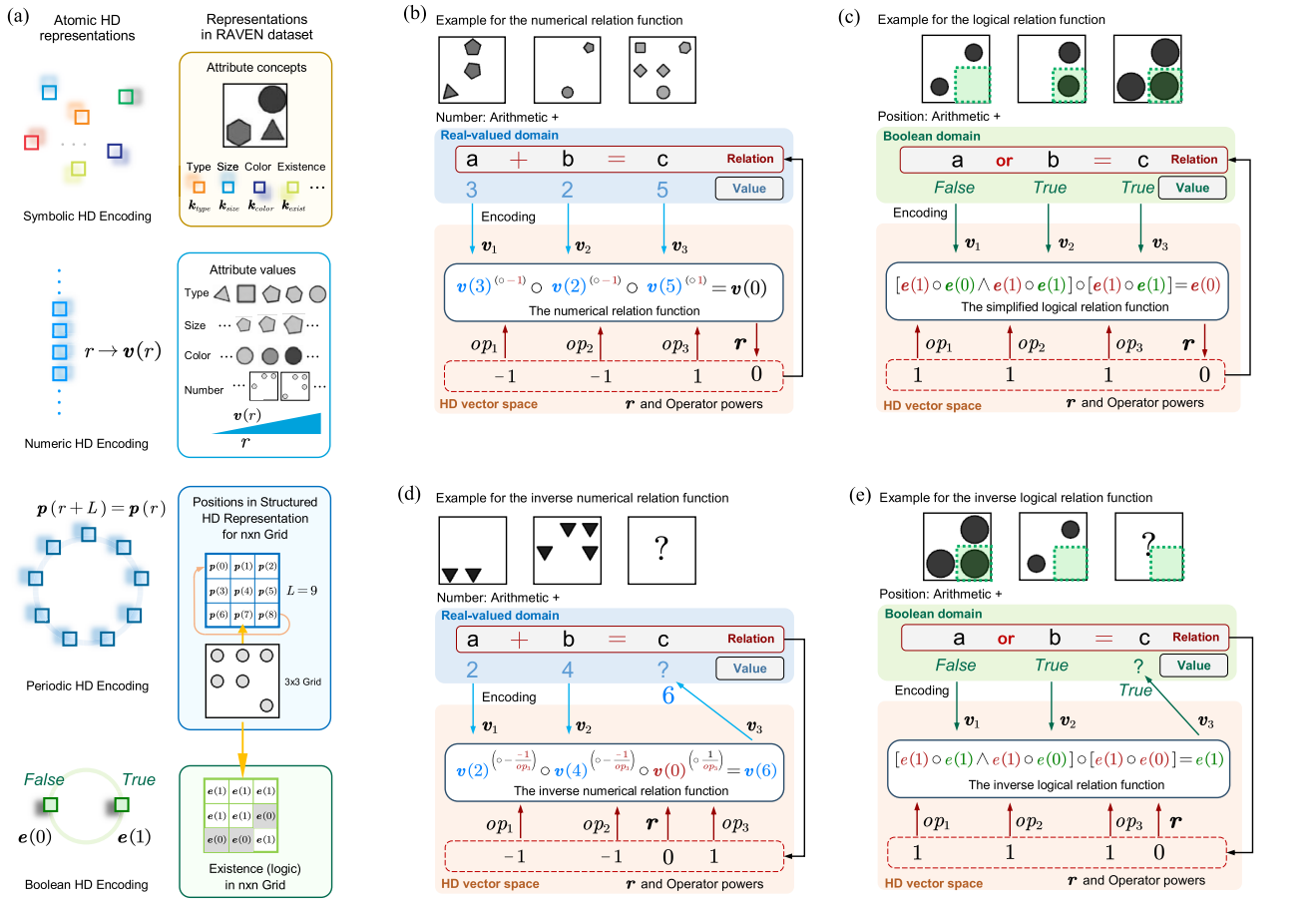


Fig. 2. Atomic HD representations and relation functions. (a) Rel-SAR model utilizes four types of atomic HD encodings. Independently sampled vectors in SE are used to represent distinct and mutually unrelated symbols or concepts, such as attribute concepts (e.g., type, size, color, and existence). NE represents real-valued quantities and supports VSA-based additive arithmetic operations, such as those applied to attribute values. PE encodes periodic values and enables additive arithmetic operations with inherent periodicity, for example, spatial positions in SHDRs for an nxn grid. BE represents logical values (*false* and *true*) and supports VSA operations for logical reasoning. (b) Example of a numerical relation function. In this example, the attribute *number* across a row of three image panels follows the Arithmetic + rule. In the real-valued domain, this relation is expressed as 3 + 2 = 5. In the HD vector space, it is represented by a ternary numerical relation function with $OP_{1:3} = [-1, -1, 1]$ and $\boldsymbol{r}_{Num} = \boldsymbol{v}(0)$. (c) Example of a logical relation function. Here, the attribute *position* across a row of three image panels follows the Arithmetic + rule. In the Boolean domain, this relation is expressed as $a \vee b = c$. In the HD vector space, it is represented by a ternary logical relation function with $OP_{1:3} = [1, 1, 1]$ and $\boldsymbol{r}_{Lgc} = \boldsymbol{e}(0)$. (d) Example of an inverse numerical relation function. In this example, the first two image panels in a row are given, and the attribute *number* is assumed to follow the Arithmetic + rule. Accordingly, in the HD vector space, the value of the *number* attribute for the third image panel can be inferred using the inverse numerical relation function with $OP_{1:3} = [-1, -1, 1]$ and $\boldsymbol{r}_{Num} = \boldsymbol{v}(0)$, yielding the result 6. (e) Example of an inverse logical relation function. In this case, the *position* attribute is assumed to follow the Arithmetic + rule. Accordingly, in the HD vector space, the truth value of object existence at each spatial position for the third image panel can be inferred using the inverse logical relation function with $OP_{1:3} = [1, 1, 1]$ and $\boldsymbol{r}_{Lgc} = \boldsymbol{e}(0)$. Additional rules and their corresponding combinations of operator powers $OP_{1:M}$ and outputs $\boldsymbol{r}$ are summarized in Table IV.

C. FPE Method

In this study, the rules in RPM are primarily numerical. We introduce the VSA representation of numerical values using the fractional power encoding method (FPE-VSA) [20], [21]. Let $x \in \mathbb{R}$ be a real number and $X \in \mathbb{R}^d$ a randomly sampled base vector. The VSA representation $\boldsymbol{v}(x) \in \mathbb{R}^d$ for any value x is obtained by repeatedly binding the base vector X with itself x times, as follows:

$$\boldsymbol{v}(x) := (X)^{(\circ x)}. \tag{3}$$

The FPE method maps arbitrary real numbers to the corresponding HD vector, and has the following properties:

$$\boldsymbol{v}(x_1 + x_2) = \boldsymbol{v}(x_1) \circ \boldsymbol{v}(x_2). \tag{4}$$

This demonstrates that addition + in the real number domain can be represented by the binding operation $\circ$ in the HD vector space.

TABLE II
LOGIC OPERATIONS IMPLEMENTED BY BE

Operation	Implementation on BE
NOT	$\neg \boldsymbol{a} = \boldsymbol{a} \circ \boldsymbol{e}(1)$
XOR	$\boldsymbol{a} \oplus \boldsymbol{b} = \boldsymbol{a} \circ \boldsymbol{b}$
AND	$\boldsymbol{a} \wedge \boldsymbol{b} = \boldsymbol{a}^{(\circ sim(\boldsymbol{a},\boldsymbol{b}))}$
OR	$\boldsymbol{a} \vee \boldsymbol{b} = (\boldsymbol{a} \oplus \boldsymbol{b}) \oplus (\boldsymbol{a} \wedge \boldsymbol{b}) = \boldsymbol{a} \circ \boldsymbol{b} \circ \boldsymbol{a}^{(\circ sim(\boldsymbol{a},\boldsymbol{b}))}$

IV. METHODOLOGY

A. Atomic HD Vectors With Semantic Representations

In neuro-vector-symbolic systems, atomic HD vector representations with meaningful semantics are essential for perception and reasoning. We introduce four types of atomic HD encodings used in our model [Fig. 2(a)]: Symbolic HD encoding (SE), numeric HD encoding (NE), periodic

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

TABLE III

ATTRIBUTE REPRESENTATIONS AND THE RELATION FUNCTIONS INVOLVED IN RULE ABDUCTIONS

Level	Attributes	HD representations		Rules in RAVEN and their types		Relation functions
Entity	Type	$\boldsymbol{v}^{type}$	Atomic (NE)	Constant	Numerical rules	Binary + Numerical
	Size	$\boldsymbol{v}^{size}$		Progression		
	Color	$\boldsymbol{v}^{color}$		Distribute Three		Ternary + Numerical
Layout	Number	$\boldsymbol{v}^{num}$		Arithmetic		
	Position	$\mathcal{C}^{\text{nxn}} = \sum_{j=1}^{n\times n} \boldsymbol{p}_j \circ \boldsymbol{v}_j^{exist}$	SHDR $\boldsymbol{p}_j$: PE $\boldsymbol{v}_j^{exist}$: SE	Constant		Binary + Numerical
				Progression		
				Distribute Three		Ternary + Numerical
		$\boldsymbol{v}^{exist}$	Atomic (BE)	Arithmetic	Logical rules	Ternary + logical

TABLE IV

RULES AND CORRESPONDING COMBINATIONS OF $OP_{1:M}$ AND r_{Num} IN RELATION FUNCTIONS

Rules in RAVEN		Ternary / Binary	op_1	op_2 / op_1	op_3 / op_2	r	Examples (Rule → Relation function)
Numerical rules	Constant			-1	$+1$	0	$v_1 = v_2 \rightarrow \boldsymbol{v}(0) = \boldsymbol{v}_1^{(\circ(-1))} \circ \boldsymbol{v}_2^{(\circ 1)}$
	Progression	$+$		-1	$+1$	$+1$,$+2$	$v_1 + 1 = v_2 \rightarrow \boldsymbol{v}(+1) = \boldsymbol{v}_1^{(\circ(-1))} \circ \boldsymbol{v}_2^{(\circ 1)}$
		$-$		-1	$+1$	-1,-2	$v_1 - 2 = v_2 \rightarrow \boldsymbol{v}(-2) = \boldsymbol{v}_1^{(\circ(-1))} \circ \boldsymbol{v}_2^{(\circ 1)}$
	Arithmetic	$+$	-1	-1	$+1$	0	$v_1 + v_2 = v_3 \rightarrow \boldsymbol{v}(0) = \boldsymbol{v}_1^{(\circ(-1))} \circ \boldsymbol{v}_2^{(\circ(-1))} \circ \boldsymbol{v}_3^{(\circ 1)}$
		$-$	-1	$+1$	$+1$	0	$v_1 - v_2 = v_3 \rightarrow \boldsymbol{v}(0) = \boldsymbol{v}_1^{(\circ(-1))} \circ \boldsymbol{v}_2^{(\circ 1)} \circ \boldsymbol{v}_3^{(\circ 1)}$
	Distribute Three		$+1$	$+1$	$+1$	*Any*	$v_1 + v_2 + v_3 = Any \rightarrow \boldsymbol{Any} = \boldsymbol{v}_1^{(\circ 1)} \circ \boldsymbol{v}_2^{(\circ 1)} \circ \boldsymbol{v}_3^{(\circ 1)}$
Logical rules	Arithmetic	$+$	$+1$	$+1$	$+1$	0	$\boldsymbol{e}(0) = (\boldsymbol{e}(1) \circ \boldsymbol{v}_1 \wedge \boldsymbol{e}(1) \circ \boldsymbol{v}_2) \circ (\boldsymbol{e}(1) \circ \boldsymbol{v}_3)$
		$-$	0	$+1$	0	0	$\boldsymbol{e}(0) = (\boldsymbol{e}(0) \circ \boldsymbol{v}_1 \wedge \boldsymbol{e}(1) \circ \boldsymbol{v}_2) \circ (\boldsymbol{e}(0) \circ \boldsymbol{v}_3)$

HD encoding (PE), and Boolean HD encoding (BE). Each encoding defines a parameterized family of HD vectors that represent concepts or values drawn from a specific domain.

1) Symbolic HD Encoding: Vectors in SE are sampled from specific distributions according to the VSA models, as described in the preliminary section. Because these vectors lack intrinsic numerical or logical relationships and are approximately pseudo-orthogonal in the HD vector space, SE is well suited for representing symbols and concepts that are assumed to be mutually independent and semantically dissimilar.

2) Numeric HD Encoding: NE employs FPE-VSA (3) [20] to represent real-valued quantities. Under this encoding, each real value r is mapped to an HD vector $\boldsymbol{v}(r) \in \mathbb{R}^d$. Additive arithmetic operations over numeric values can be implemented through the binding operation defined in (4) [21], enabling numerical manipulation directly in the HD vector space.

3) Periodic HD Encoding: PE is a specialization of NE designed to represent periodic quantities. Given a base vector P, whose element phases ρ_i are sampled from a discrete distribution (e.g., in FHRR, $\rho_i \sim \mathcal{U}\{2\pi j/L \ \forall j \in \{1,\ldots,L\}\}$, with L being an even integer), PE defines a family of HD vectors parameterized by a scalar value r as $\boldsymbol{p}(r) := P^{(\circ r)}$. These vectors are approximately pseudo-orthogonal for distinct values of r and exhibit periodicity with period L [21]

$$\boldsymbol{p}(r + L) = \boldsymbol{p}(r). \tag{5}$$

When L is odd, a periodic encoding with period L can be constructed by selecting every other vector from the corresponding encoding with period $2L$.

4) Boolean HD Encoding: BE is a special case of PE with period $L = 2$, designed to represent Boolean values. Following the same construction principle as in PE, BE defines two HD vectors, $\boldsymbol{e}(0)$ and $\boldsymbol{e}(1)$, corresponding to the Boolean values *False* and *True*, respectively. Basic logical operations can be implemented directly in the HD vector space, as summarized in Table II, where $\boldsymbol{a}, \boldsymbol{b} \in \{\boldsymbol{e}(0), \boldsymbol{e}(1)\}$ represent arbitrary Boolean values.

B. Relation Functions Based on Atomic HD Representations

Abductive reasoning in RPM relies on binary and ternary relations among the attributes of corresponding objects within each row of three panels [Fig. 1(a)], encompassing both numerical and logical relations. To support such rule abduction, we design general relation functions grounded in VSA algebra, which operate on the aforementioned atomic vector representations and serve as explicit relation representations for reasoning.

1) Relation Functions: Relation functions characterize relationships among multiple HD vector representations and are categorized into two types: numerical and logical. Numerical relation functions operate on vectors encoded by NE or PE, whereas logical relation functions operate on vectors encoded by BE.

The numerical relation function, R_{Num}, is defined as follows:

$$\boldsymbol{r}_{Num} = R_{Num}(\boldsymbol{v}_{1:N}, OP_{1:M}) = \circ_{i=1}^{N} \boldsymbol{v}_i^{(\circ op_i)} \tag{6}$$

where $\boldsymbol{v}_{1:N} := \{\boldsymbol{v}_i\}_{i=1}^{N}$ denotes the input set of N HD vector representations, and N corresponds to the arity of the relation function. $OP_{1:M} := \{op_i\}_{i=1}^{M}$ (with $M = N$) denotes the operator powers, which can be regarded as learnable or configurable parameters of the relation function. Each operator power op_i specifies the number of self-binding operations applied to the corresponding input vector $\boldsymbol{v}_i$ in the HD vector space. The notation $\circ_{i=1}^{N}$ denotes the sequential binding operation applied to the N HD vector representations. $\boldsymbol{r}_{Num}$ is the output HD representation. For the binary numerical relation function, $N =$

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

2 and $M = 2$, while for the ternary numerical relation function, $N = 3$ and $M = 3$.

Based on the arithmetic properties of NE and PE, R_{Num} provides a principled mechanism for translating linear arithmetic relations from the real-valued domain into compositional binding operations in a vector-symbolic representation. The combination of $OP_{1:M}$ and $\boldsymbol{r}_{Num}$ determines the specific numerical relation in this vector-symbolic method. Taking the relation $a + b = c$ in the real-valued domain as an example [Fig. 2(b)], it can be equivalently rewritten as $-a - b + c = 0$. In the HD vector space, this relation is represented by a ternary numerical relation function with $OP_{1:3} = [-1, -1, 1]$ and $\boldsymbol{r}_{Num} = \boldsymbol{v}(0)$, that is, $\boldsymbol{v}(a)^{(\circ -1)} \circ \boldsymbol{v}(b)^{(\circ -1)} \boldsymbol{v}(c)^{(\circ 1)} = \boldsymbol{v}(0)$. It shows that the operator powers serve as the vector-symbolic analogs of signed coefficients in a linear combination of $\boldsymbol{v}_{1:N}$, where the aggregate binding result equals $\boldsymbol{r}_{Num}$. Consequently, the numerical relation function can enable arithmetic reasoning to be performed directly in the HD vector space.

Similarly, *the simplified logical relation function*, R_{Lgc}, is defined as follows:

$$\boldsymbol{r}_{Lgc} = R_{Lgc}\left(\boldsymbol{v}_{1:N}, OP_{1:M}\right) = \left(op_1 \boldsymbol{v}_1 \wedge op_2 \boldsymbol{v}_2\right) \circ op_3 \boldsymbol{v}_3 \tag{7}$$

where $\boldsymbol{v}_{1:N} := \{\boldsymbol{v}_i\}_{i=1}^{N} \in \{\boldsymbol{e}(0), \boldsymbol{e}(1)\}$ denotes the input set of N HD vector representations. The full version of logical relation function is described in Supplementary material I. Here, we consider only the ternary logical relation, so $N = 3$ and $M = 3$. The parameter $OP_{1:M} := \{op_i\}_{i=1}^{M}$, where $op_i \in \{0, 1\}$ determines whether to negate $\boldsymbol{v}_i$, with negation ($\neg$) applied when $op_i = 1$ and no negation applied when $op_i = 0$ (see Supplementary material I). The symbol $\wedge$ denotes the AND operation, as shown in Table II. Based on the computational properties of BE detailed in Table II, R_{Lgc} can describe the ternary logical relations involved in RPM. The combination of the operator $OP_{1:M}$ and the output $\boldsymbol{r}_{Lgc}$ determines the specific logical relation in this vector-symbolic method. Taking the relation a or $b = c$ in the Boolean domain as an example [Fig. 2(c)], this relation in the HD vector space is represented by a ternary logical relation function with $OP_{1:3} = [1, 1, 1]$ and $\boldsymbol{r}_{Lgc} = \boldsymbol{e}(0)$.

2) Inverse Relation Functions: Rule execution in RPM involves inferring the third attribute value in a row of panels from the first two, given a known relation, and thus constitutes the inverse problem of rule abduction. In the vector-symbolic method, given the operator power $OP_{1:M}$ and the output $\boldsymbol{r}$, the last vector representation $\boldsymbol{v}_N$ can be inferred from the preceding inputs $\boldsymbol{v}_{1:N-1}$ via the inverse of the relation functions. Accordingly, based on (6), the inverse numerical relation function is defined as follows:

$$\begin{aligned} \boldsymbol{v}_N &= R_{Num}^{-1}\left(\boldsymbol{v}_{1:N-1}, OP_{1:M}, \boldsymbol{r}\right) \\ &= \left(\circ_{i=1}^{N-1} \boldsymbol{v}_i^{(\circ(-op_i/op_M))}\right) \circ \boldsymbol{r}^{(\circ(1/op_M))}. \end{aligned} \tag{8}$$

Similarly, according to (7), the inverse logical relation function is defined as follows:

$$\begin{aligned} \boldsymbol{v}_N &= R_{Lgc}^{-1}\left(\boldsymbol{v}_{1:N-1}, OP_{1:M}, \boldsymbol{r}\right) \\ &= \left(op_1 \boldsymbol{v}_1 \wedge op_2 \boldsymbol{v}_2\right) \circ op_3 \boldsymbol{r}. \end{aligned} \tag{9}$$

Examples of the inverse numerical and inverse logical relation functions are illustrated in Fig. 2(d) and (e), respectively.

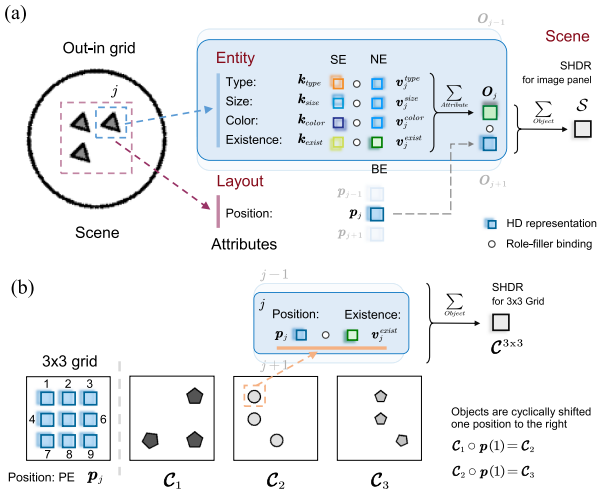


Fig. 3. SHDRs for the image panel and the nxn grid. (a) SHDR for the image panel (10). Taking the O-IG configuration as an example, an image panel contains multiple objects, each with four entity attributes: *type*, *size*, *color*, and *existence*. Through the first layer of role–filler binding, these attributes are combined to form an SHDR for each object. Additionally, at the layout level, each object has a *position* attribute. By applying a second layer of role–filler binding, the SHDR for the entire image panel is constructed. (b) SHDR for the nxn grid (12). Taking the 3×3 grid configuration as an example, the position vectors for all objects are represented using PE with a period of $3 \times 3 = 9$. The SHDR $\mathcal{C}^{3x3}$ for this 3×3 grid is obtained by performing role–filler binding between the corresponding *position* vectors $\boldsymbol{p}_j$ and *existence* vectors $\boldsymbol{v}_j$. Due to the periodic nature of the *position* vectors, as all objects shift positions cyclically, the SHDR undergoes a binding operation with the *position* vectors corresponding to the magnitude of the shift.

C. SHDR and Its Attribution Decomposition

VSAs enable the construction of structured symbolic representations from atomic HD vectors and support their direct decomposition through algebraic operations [11]. This section describes the construction of SHDRs for individual image panels and the subsequent decomposition process for retrieving attribute representations. In addition, we introduce SHDRs at the component level to encode the holistic structure of an nxn grid ($n = 2, 3$).

1) SHDR for the Image Panel: In the RAVEN dataset, each image panel contains multiple objects, with each object characterized by several attributes. Consequently, the SHDR for each image panel can be obtained through two layers of role–filler bindings [Fig. 3(a)]. First, the bundling operation is used to construct an SHDR for each object at the entity level by combining its attributes. Then, another bundling operation aggregates these object-level representations to construct an SHDR of the image panel at the scene level. Therefore, each image panel $\mathcal{X} \in \mathbb{R}^{r \times r}$, with a resolution $r \times r$, can be represented by an SHDR $\mathcal{S} \in \mathbb{R}^{d}$ as follows:

$$\begin{aligned} \mathcal{S} &= \sum_{j=1}^{N_{pos}} \boldsymbol{p}_j \circ \boldsymbol{O}_j \\ &= \sum_{j=1}^{N_{pos}} \boldsymbol{p}_j \circ \left(\sum_{attr \in ATTR} \boldsymbol{k}_{attr} \circ \boldsymbol{v}_j^{attr}\right). \end{aligned} \tag{10}$$

Here, $\boldsymbol{O}_j$ denotes the SHDR of the jth object at the entity level, encoding its various attributes, including type, size, color, and existence. The attribute set is $ATTR = \{type, size, color, exist\}$. At the entity level, the key vector $\boldsymbol{k}_{attr}$ represents the class of

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

a specific attribute $attr \in ATTR$, while the value vector $\boldsymbol{v}_j^{attr}$ encodes the attribute's value at the position j. At the scene level, the position vector $\boldsymbol{p}_j$ specifies the location of the jth object.

2) Representation Decomposition: Given an estimated SHDR $\hat{\mathcal{S}} \in \mathbb{R}^d$ of an image panel, all SHDRs of objects $\hat{\boldsymbol{O}}_j$ at the entity level, along with the corresponding attribute representations $\hat{\boldsymbol{v}}_j^{attr}$, can be derived through a series of unbinding operations [36]. The decomposition process is shown as follows:

$$\begin{cases} \hat{\boldsymbol{O}}_j \approx \boldsymbol{p}_j \oslash \hat{\mathcal{S}} \approx \boldsymbol{p}_l^{-1} \circ \hat{\mathcal{S}} \\ \hat{\boldsymbol{v}}_j^{attr} \approx \boldsymbol{k}_{attr} \oslash \hat{\boldsymbol{O}}_j \approx \boldsymbol{k}_{attr}^{-1} \circ \hat{\boldsymbol{O}}_j. \end{cases} \tag{11}$$

It is important to note that due to inaccuracies in the estimated SHDR $\hat{\mathcal{S}}$ and the crosstalk noise introduced by the unbinding operation, the estimated attribute representations $\hat{\boldsymbol{v}}_j^{attr}$ may not fully match the original $\boldsymbol{v}_j^{attr}$ used in (10).

3) SHDR for the nxn Grid Component: In the RAVEN dataset, three figure configurations—2×2 grid, 3×3 grid, and Out-InGrid (O-IG)—include components where objects are arranged in an nxn grid pattern at the layout level [16]. Since the positions in the nxn grid involve component-level rule reasoning, the SHDR for the nxn grid component ($n = 2, 3$), focusing only on positions and object existence, is introduced as follows [Fig. 3(b)]:

$$\mathcal{C}^{\text{nxn}} = \sum_{j=1}^{n \times n} \boldsymbol{p}_j \circ \boldsymbol{v}_j^{exist}. \tag{12}$$

D. Rules From the Perspective of Relation Functions

The RAVEN dataset contains four rules—constant, progression, arithmetic, and distribute three—applied to five rule-governing attributes [16]. These include three entity-level attributes: *type*, *size*, and *color*, and two layout-level attributes: *number* and *position*. In this study, the HD representations of these attribute values for rule reasoning, along with the corresponding relations between rules and relation functions, are summarized in Table III.

For the attributes *type*, *size*, *color*, and *number*, the four associated rules follow additive arithmetic operations, meaning the attribute values $\boldsymbol{v}^{attr}$ ($attr \in \{type, size, color, number\}$) are encoded using NE. Therefore, these rules are represented using numerical relation functions (6): constant and progression are modeled as binary relation functions, whereas arithmetic and distribute three are modeled as ternary relation functions. Each rule is associated with specific combinations of $OP_{1:M}$ and $\boldsymbol{r}_{Num}$, as detailed in Table IV.

For the attribute *position*, the rules constant, progression, and distribute three operate over an nxn grid containing multiple objects, which collectively follow additive arithmetic operations. Therefore, we use the SHDR $\mathcal{C}^{\text{nxn}}$ for the nxn grid (12) to represent the attributes required by these three rules. Since progression involves a cyclic left or right shift of all objects [Fig. 3(b)], the position vectors $\boldsymbol{p}_j$ in set $\mathcal{C}^{\text{nxn}}$ during rule reasoning are encoded using PE. The object existence vectors $\boldsymbol{v}_j^{exist}$ are encoded using SE. These three rules can also be represented via numerical relation functions (6). Take the rule progression (+ 1) as an example, where the positions of objects undergo a cyclic right shift. In Fig. 3(b), the SHDRs of the 3×3 grid across a row of three panels exhibit two numerical relations: $\mathcal{C}_1 \circ \boldsymbol{p}(1) = \mathcal{C}_2$ and $\mathcal{C}_2 \circ \boldsymbol{p}(1) = \mathcal{C}_3$, which can be defined using a binary numerical relation function.

In addition, the rule arithmetic on the attribute *position* belongs to the logical rule [16]. The attribute values $\boldsymbol{v}_j^{exist}$ are encoded using BE, which supports operations as described in Table II. Therefore, the rule arithmetic for *position* corresponds to a ternary logical relation function (7), with the specific combinations of $OP_{1:M}$ and $\boldsymbol{r}_{Lgc}$ detailed in Table IV.

E. Systematic Abductive Reasoning Model

In this section, we present the systematic abductive reasoning model with diverse relation representations (Rel-SAR), inspired by the NVSA [11]. An overview of Rel-SAR is depicted in Fig. 4(a). Similar to previous neuro-symbolic models for abstract visual reasoning, Rel-SAR combines a neural visual perception frontend with a symbolic reasoning backend, both utilizing VSA representations with meaningful semantics to facilitate systematic reasoning. The perception frontend employs a neural network as the image encoder to extract the SHDR $\mathcal{S}$ of each image panel $\mathcal{X}$ in the RPM. Feature disentanglement is achieved through representation decomposition, yielding the HD attribute representations ($\boldsymbol{v}$, $\boldsymbol{p}$, and $\mathcal{C}$: Table III) required for reasoning in the backend. The reasoning backend consists of three main modules: the rule abduction module, the rule execution module, and the answer selection module. The rule abduction module extracts the corresponding rules ($OP_{1:M}$ and $\boldsymbol{r}$: Table IV) for each attribute representation according to appropriate relation function [(6) and (7), Table III]. Subsequently, the rule execution module then applies these rules to predict the attribute representations of the missing panel using the corresponding inverse relation functions (8) and (9). Finally, the answer selection module compares the predicted attribute representations of the missing panel with the candidate answer panels to select the correct response. To facilitate understanding of the subsequent detailed descriptions of the perception frontend and reasoning backend, Fig. 4(d) illustrates the perception–reasoning process of Rel-SAR using an example RPM problem.

1) Perception Frontend: The perception frontend operates independently on each of the 16 image panels to extract the HD representations of attributes required for abductive reasoning [Fig. 4(a) and (b)]. For a given image panel $\mathcal{X}^{ind} \in \mathbb{R}^{r \times r}$, where $ind \in \{(1,1),(1,2),\ldots,(3,2)\}$ for eight contexts and $ind \in \{1,2,\ldots,8\}$ for eight candidates, the frontend uses a trainable neural network (ResNet-50) as the image encoder to map the image panel to its estimated SHDR $\hat{\mathcal{S}}^{ind} \in \mathbb{R}^d$: $f_\theta : \mathcal{X} \rightarrow \hat{\mathcal{S}}$, where θ represents the trainable parameters of the network. Theoretically, the expected SHDR $\mathcal{S}$ for each panel should be derived from the corresponding attribute representations, as described by (10). Therefore, the learning objective of f_θ is to minimize the difference between its output $\hat{\mathcal{S}}$ and the theoretical SHDR $\mathcal{S}$, formulated as

$$\min_\theta \| f_\theta(\mathcal{X}; \theta) - \mathcal{S} \|. \tag{13}$$

Subsequently, the estimated SHDR $\hat{\mathcal{S}}^{ind}$ of each panel undergoes representation decomposition [Fig. 4(b)], as described in (11), to obtain the estimated HD attribute representations of individual objects. These include *type* ($\hat{\boldsymbol{v}}_j^{type}$), *size* ($\hat{\boldsymbol{v}}_j^{size}$), *color* ($\hat{\boldsymbol{v}}_j^{color}$), and *existence* ($\hat{\boldsymbol{v}}_j^{exist}$), where j denotes the position index of the corresponding object.

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

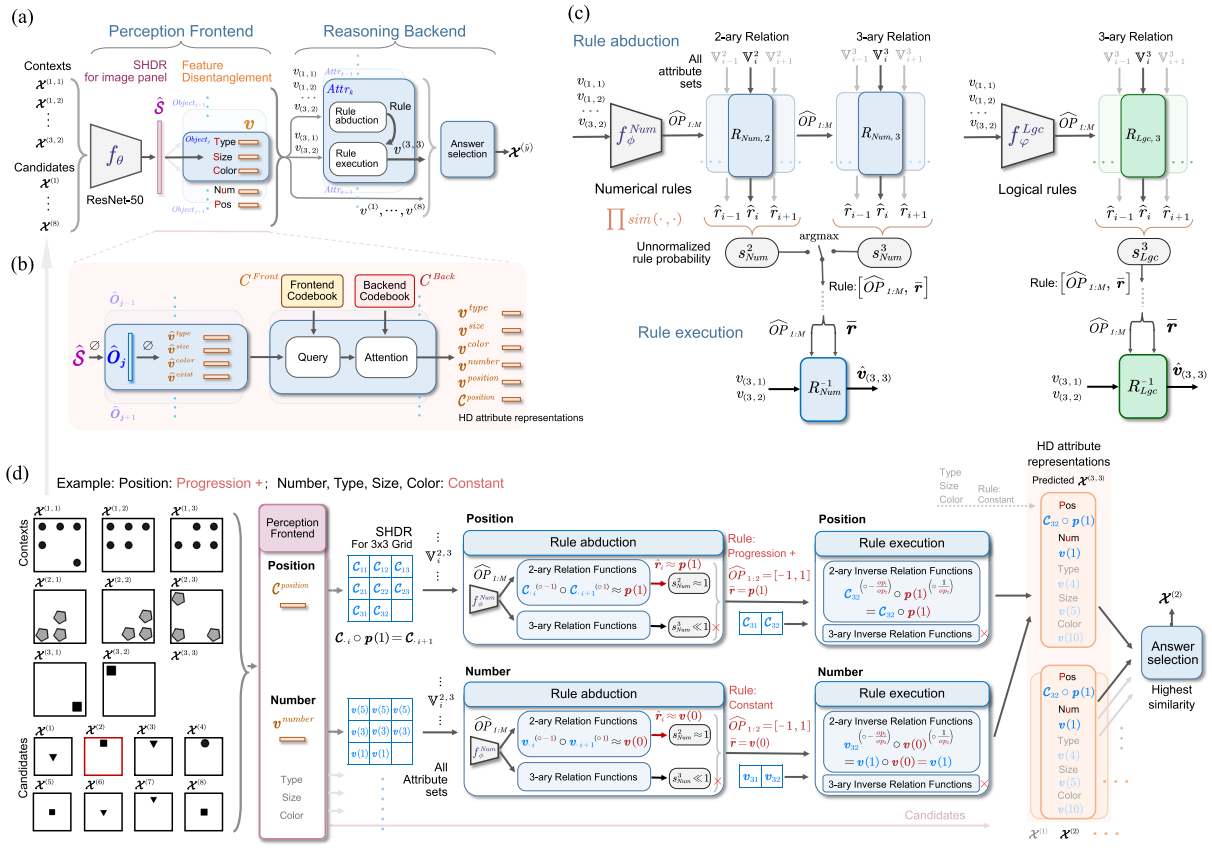


Fig. 4. Systematic abductive reasoning model with diverse relation representations (Rel-SAR). (a) Overall architecture of Rel-SAR model. Our model consists of a visual perception frontend, which processes object attributes for eight context and eight candidate image panels in an RPM test, and a reasoning backend that performs symbolic arithmetic and logical reasoning. The perception frontend utilizes a neural network as the image encoder, f_θ, to obtain the SHDR of each image panel, and then perceives attributes in the form of HD representations required by the downstream reasoning. In the reasoning backend, the rule abduction module extracts rules for each attribute representation using relation functions. The rule execution module then predicts the missing panel's attribute representations based on inverse relation functions. Finally, the answer selection module compares the predicted attributes of the missing panel with those in the candidate panels and selects the option with the highest similarity. (b) Given the predicted SHDR for each panel, the SHDR of all objects and their corresponding HD attribute representations can be obtained via representation decomposition. Subsequently, the estimated HD attribute representations are refined in two steps: querying the frontend codebook and applying attention based on the backend codebook. This process produces HD attribute representations suitable for backend reasoning, including attributes such as *type*, *size*, *color*, *number*, and *position*. (c) In the rule abduction module, two rule learners f_ϕ^{Num} and f_φ^{Lgc} predict the operator powers $\widehat{OP}_{1:M}$ for numerical and logical relation functions based on attributes in the context panels. These predicted $\widehat{OP}_{1:M}$ ensure that all binary or ternary relation input sets ($\mathbb{V}^N, N = 2, 3$) produce the same output $\hat{\boldsymbol{r}}$ when processed through their respective relation functions. Therefore, the rule defined by $\widehat{OP}_{1:M}$ and $\hat{\boldsymbol{r}}$ with the highest overall $\hat{\boldsymbol{r}}$ similarity, also viewed as unnormalized probability, is considered the underlying rule. The rule execution module then predicts the attributes of the missing panel using inverse relation functions with the estimated rules. (d) Example of an RPM problem in which the attribute *position* follows the Progression + rule, while *number*, *type*, *size*, and *color* follow the constant rule. In the reasoning backend, taking *position* as an example, adjacent panels within each row of the 3×3 grid satisfy $\mathcal{C}_i \circ \boldsymbol{p}(1) = \mathcal{C}_{i+1}$ [also see Fig. 3(b)]. When the attribute sets from all binary inputs $\mathbb{V}_i^2$ are fed into binary numerical relation functions with $\widehat{OP}_{1:2} = [-1, 1]$, all resulting relation outputs satisfy $\hat{\boldsymbol{r}}_i \approx \boldsymbol{p}(1)$. Consequently, the unnormalized rule probability for the binary relation functions is $s^2_{Num} \approx 1$. In contrast, because no consistent ternary relation holds across each row, $s^3_{Num} \ll 1$. The rule for the *position* attribute is therefore identified as $\widehat{OP}_{1:2} = [-1, 1]$ with $\bar{\boldsymbol{r}} = \boldsymbol{p}(1)$. Using the corresponding binary inverse numerical relation function, the SHDR of the missing panel is inferred as $\mathcal{C}_{33} = \mathcal{C}_{32} \circ \boldsymbol{p}(1)$. This example also illustrates the rule abduction and execution processes for the *number* attribute with the Constant rule. Finally, since the predicted HD attribute representations of the missing panel are most similar to those of the second candidate panel, the model selects $\mathcal{X}^{(2)}$ as the answer.

The resulting HD attribute representations are expected to be drawn from a set of frontend codebooks that enumerate the attributes of interest in the RAVEN dataset [Fig. 4(b)]. These frontend codebooks include $C^{Front}_{Num} := \{\boldsymbol{v}(r)\}_{r=0}^{9} \cup \{\boldsymbol{v}_{null}\}$ and $C^{Front}_{Lgc} := \{\boldsymbol{e}(r)\}_{r=0}^{1}$, which represent the numerical value and logic, respectively. Here, $\boldsymbol{v}_{null}$ represents the null attribute representation when there is no object. To improve the performance of the neural network in encoding the SHDR of an image panel, all hypervectors in the frontend codebooks are randomly and independently generated using SE, rather than using NE, PE, or BE.

However, the estimated HD attribute representations for each object, $\hat{\boldsymbol{v}}_j^{attr}$ ($attr \in \{type, size, color, exist\}$), cannot be directly applied to the reasoning backend. First, as indicated in (11), these representations are obtained by unbinding from an SHDR that superposes multiple attributes; consequently, they inevitably include crosstalk noise [36]. Second, as these representations are derived from the frontend codebooks constructed using SE, there are no intrinsic arithmetic or logical relations among attributes, which hinders effective reasoning.

To address these limitations, we adopt an approach similar to the attention mechanism [39] to transform the estimated HD attribute representations into forms suitable for the reasoning backend [Fig. 4(b)]. In the query stage, the estimated HD attribute representations ($\hat{\boldsymbol{v}}_j^{attr}$) are used as query vectors to compute their similarity with all candidate vectors of the

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

corresponding attribute in the frontend codebooks. In the attention stage, these similarity scores are then used as attention weights to perform a weighted summation of the corresponding vectors from the backend codebooks, whose hypervectors are generated according to the attribute type, as summarized in Table III. The backend codebooks consist of C^{Back}_{Num} (NE), $C^{Back}_{Lgc,BE}$ (BE), $C^{Back}_{Lgc,SE}$ (SE), and $C^{Back}_{Pos,\text{nxn}} := \{\boldsymbol{p}_r\}^{n^2}_{r=1}$ (PE for positions in nxn grid). The updated HD attribute representations obtained after the weighted summation are then used by the reasoning backend. Details of this process are provided below.

The Query Stage: For each numerical attribute $attr \in \{type, size, color\}$, we compute the cosine similarity between the estimated HD attribute representation $\hat{\boldsymbol{v}}^{attr}_j$ and all candidate vectors of C^{Front}_{Num} in the frontend codebooks

$$W^{attr}_j = sim\left(\hat{\boldsymbol{v}}^{attr}_j, C^{Front}_{Num}\right) \tag{14}$$

where $W^{attr}_j(r)$ $(r \in \{0, 1, \ldots, 9, null\})$ denotes the attention weights corresponding to the value r of attribute $attr$ at the jth position, as determined by the query similarity.

Similarly, for the logical attribute $attr \in \{exist\}$, the attention weights are obtained by querying the logic codebook C^{Front}_{Lgc} as follows:

$$W^{exist}_j = softmax\left(\beta \cdot sim\left(\hat{\boldsymbol{v}}^{exist}_j, C^{Front}_{Lgc}\right)\right) \tag{15}$$

where $W^{exist}_j(r)$ $(r \in \{0, 1\})$ corresponds to the presence and absence of an object at the jth position, respectively. The $softmax$ function normalizes the attention weights, and β denotes the inverse softmax temperature.

The Attention Stage: The reasoning backend requires HD attribute representations for entity-level attributes (*type*, *size*, and *color*) and layout-level attributes (*number* and *position*) (Table III).

For the numerical attribute $attr \in \{type, size, color\}$, the corresponding updated HD representation $\boldsymbol{v}^{attr}_j$ is obtained via a weighted summation over the numerical backend codebook C^{Back}_{Num} as follows:

$$\boldsymbol{v}^{attr}_j = \sum_{r\in\{0,\ldots,9,null\}} W^{attr}_j(r) \cdot \boldsymbol{v}(r), \boldsymbol{v}(r) \in C^{Back}_{Num}. \tag{16}$$

For the logical existence attribute $attr \in \{exist\}$, the updated HD representations $\boldsymbol{v}^{exist}_j$ are obtained through the weighted summation over the backend codebook $C^{Back}_{Lgc,SE}$ and $C^{Back}_{Lgc,BE}$, respectively, as follows:

$$\boldsymbol{v}^{exist,ET}_j = \sum_{r\in\{0,1\}} W^{exist}_j(r) \cdot \boldsymbol{e}(r), \boldsymbol{e}(r) \in C^{Back}_{Lgc,ET} \tag{17}$$

where $ET \in \{SE, BE\}$ represents the type of atomic HD encodings.

In addition, we construct overall HD attribute representations for *type*, *size*, and *color* attributes within the nxn grid. These representations $\boldsymbol{v}^{attr}_{\text{nxn}}$ $(attr \in \{type, size, color\})$ are obtained by bundling the corresponding HD attribute representations of all objects in the grid with their attention weights of *existence* as follows:

$$\boldsymbol{v}^{attr}_{\text{nxn}} = \sum_{j=1}^{n^2} W^{exist}_j(1) \cdot \boldsymbol{v}^{attr}_j. \tag{18}$$

TABLE V
ATTRIBUTE SETS FOR N-ARY RELATION

$\mathbb{V}^2$	$(\boldsymbol{v}_{(1,1)}, \boldsymbol{v}_{(1,2)})$, $(\boldsymbol{v}_{(1,2)}, \boldsymbol{v}_{(1,3)})$, $(\boldsymbol{v}_{(2,1)}, \boldsymbol{v}_{(2,2)})$, $(\boldsymbol{v}_{(2,2)}, \boldsymbol{v}_{(2,3)})$, $(\boldsymbol{v}_{(3,1)}, \boldsymbol{v}_{(3,2)})$; $(\boldsymbol{v}_{(3,2)}, \boldsymbol{v}_{(y)})$
$\mathbb{V}^3$	$(\boldsymbol{v}_{(1,1)}, \boldsymbol{v}_{(1,2)}, \boldsymbol{v}_{(1,3)})$, $(\boldsymbol{v}_{(2,1)}, \boldsymbol{v}_{(2,2)}, \boldsymbol{v}_{(2,3)})$; $(\boldsymbol{v}_{(3,1)}, \boldsymbol{v}_{(3,2)}, \boldsymbol{v}_{(y)})$

For the layout-level attribute *number*, the HD attribute representation $\boldsymbol{v}^{number}$ is obtained by projecting the sum of the attention weights of presence to FPE-VSA as follows:

$$\boldsymbol{v}^{number} = \boldsymbol{v}^{\left(\circ \sum_{j=1}^{n^2} W^{exist}_j(1)\right)} \tag{19}$$

where $\boldsymbol{v}$ denotes the base vector of the numerical backend codebook C^{Back}_{Num}.

The layout-level attribute *position* within the nxn grid involves both numerical and logical rules (Table III). Therefore, its HD attribute representations correspond to two distinct rules: (i) the logical representation of individual objects $\boldsymbol{v}^{position}_j$ and (ii) the overall HD position representation $\mathcal{C}^{position}$ of the entire nxn grid. The object-level position representation is defined as an HD existence representation with logical computational properties, i.e., $\boldsymbol{v}^{position}_j = \boldsymbol{v}^{exist,BE}_j$. Inspired by the SHDR formulation for the nxn grid (12), the overall HD position representation $\mathcal{C}^{position}$ is computed as follows:

$$\mathcal{C}^{position}_{\text{nxn}} = \sum_{j=1}^{n\times n} W^{exist}_j(1) \cdot \boldsymbol{p}_j \circ \boldsymbol{v}^{exist,SE}_j, \boldsymbol{p}_j \in C^{Back}_{Pos,\text{nxn}}. \tag{20}$$

2) Reasoning Backend: The Rel-SAR model efficiently implements systematic abductive reasoning by leveraging HD attribute representations and VSA-based relation functions. HD attribute representations from the frontend are transformed into the HD vector space, enabling VSA operations on both numerical and logical relation functions (6)–(9). Consequently, the reasoning backend can systematically perform rule abduction and rule execution directly through these relational functions, without relying on an extensive set of explicitly defined rule templates.

Rule Abduction: Attributes in the RAVEN dataset follow row-major binary or ternary relations [16]. All possible binary $\mathbb{V}^2$ and ternary $\mathbb{V}^3$ relation tuples in the RPM test are presented in Table V, where $\boldsymbol{v}_{(i,j)}$ denotes the HD attribute representation for a given attribute in the context panel at row i and column j, and $\boldsymbol{v}_{(y)}$ represents the corresponding HD attribute representation of the target answer panel. Consequently, rule abduction can be formulated as an optimization problem: for both numerical and logical rules, the rule abduction module must identify a set of operator powers $OP_{1:M}$ such that all N-ray $(N = 2, 3)$ relation tuples yield a consistent output $\boldsymbol{r}_{Num/Lgc}$ when processed through their respective relation functions $R_{Num/Lgc}$. Formally, this objective is defined as follows:

$$\max_{OP_{1:M}} s^N = \prod_{\mathbb{V}^N_i, \mathbb{V}^N_j \in \mathbb{V}^N}^{i\neq j} sim\left(R\left(\mathbb{V}^N_i, OP_{1:M}\right), R\left(\mathbb{V}^N_j, OP_{1:M}\right)\right) \tag{21}$$

where sim denotes cosine similarity, and R represents either the numerical relation function R_{Num} or the logical relation function R_{Lgc}. $\mathbb{V}^N_i, \mathbb{V}^N_j \in \mathbb{V}^N$ $(i \neq j)$ refers to any two relation tuples selected from $\mathbb{V}^N$ $(N = 2, 3)$. $s^N_{Num/Lgc}$ measures the

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

overall consistency among the outputs r of all corresponding relation tuples and can be interpreted as an unnormalized probability of the underlying rule.

Based on the above idea, in the rule abduction module [Fig. 4(c) Left], the HD attribute representations of eight context panels for a given numerical attribute are fed into a trainable neural network f_ϕ^{Num}, which serves as the numerical rule learner and predicts the operator powers $\widehat{OP}_{1:M}$. These predicted parameters are optimized to maximize the objective in (21). Subsequently, all binary and ternary relation tuples (Table V) are processed by the corresponding numerical relation functions using the predicted $\widehat{OP}_{1:M}$, yielding outputs $\hat{\boldsymbol{r}}_{Num}$. Based on these outputs, the unnormalized probability s^2_{Num} and s^3_{Num} for binary and ternary relations, respectively, are computed using (21). The operator powers $\widehat{OP}_{1:M}$ and the averaged output $\bar{\boldsymbol{r}}_{Num}$ associated with the larger s^N_{Num} are then selected as the inferred numerical rule.

Similarly, a separate trainable rule learner f_φ^{Lgc} is used to predict the operator powers $\widehat{OP}_{1:M}$ for logical rules associated with the attribute *position* [Fig. 4(c) Right]. Since the logical rules in the RAVEN dataset involve only ternary relations, all logical attribute representations are organized according to the ternary relation tuples listed in Table V and processed by the ternary logical relation functions using the predicted $\widehat{OP}_{1:M}$. Based on the outputs $\hat{\boldsymbol{r}}_{Lgc}$ from the relation functions, the unnormalized probability s^3_{Lgc} for logical relations can be computed (21). Subsequently, s^3_{Lgc} is compared with s^N_{Num}, which corresponds to numerical relations for the attribute position. If s^3_{Lgc} is larger, the operator powers $\widehat{OP}_{1:M}$ and the averaged output $\bar{\boldsymbol{r}}_{Lgc}$ for logical relations are selected as the inferred logical rule.

Rule Execution: After obtaining the predicted operator powers $\widehat{OP}_{1:M}$ and the outputs $\bar{\boldsymbol{r}}_{Num/Lgc}$ that represent the rules, we apply these rules to infer the HD attribute representations of the missing panel. For a given attribute, the attribute representations from the first two panels in the third row of the RPM instance are provided as inputs to the inverse numerical and logical relation functions using the predicted $\widehat{OP}_{1:M}$ and $\bar{\boldsymbol{r}}_{Num/Lgc}$, resulting in the retrieval of the missing HD attribute representation $\hat{\boldsymbol{v}}_{(3,3)}$

$$\hat{\boldsymbol{v}}_{(3,3)} = R^{-1}_{Num/Lgc}\left(\mathbb{V}, \widehat{OP}_{1:M}, \bar{\boldsymbol{r}}_{Num/Lgc}\right) \tag{22}$$

where $\mathbb{V} = (\boldsymbol{v}_{(3,2)})$ for binary relation rules, and $\mathbb{V} = (\boldsymbol{v}_{(3,1)}, \boldsymbol{v}_{(3,2)})$ for ternary relation rules.

The Final Answer Selection: Finally, we calculate the similarity between all HD attribute representations of the missing panel $(\boldsymbol{v}^{position}_{(3,3)}, \boldsymbol{v}^{number}_{(3,3)}, \boldsymbol{v}^{type}_{(3,3)}, \boldsymbol{v}^{size}_{(3,3)}, \boldsymbol{v}^{color}_{(3,3)})$ and the corresponding attribute representations of each candidate panel (y). The predicted answer panel $\hat{y}$ is the one with the highest total similarity score.

F. Model Training

1) End-to-End Training: During end-to-end training, the Rel-SAR model utilizes the rule labels provided by the RAVEN dataset and the answer labels to optimize the objectives of visual perception (13) and rule abduction (21). Based on the rule labels, the corresponding ground truth $OP^{gt}_{1:M}$ and $\boldsymbol{r}^{gt}_{Num/Lgc}$, which represent the rules, can be obtained from Table IV. To facilitate the learning of $OP_{1:M}$, we design the loss function $\mathcal{L}_{op}$, which constrains the rule learners f_ϕ^{Num} and f_φ^{Lgc} to optimize the objective described in (21)

$$\mathcal{L}_{op} = MSE\left(\widehat{OP}_{1:M}, OP^{gt}_{1:M}\right) \tag{23}$$

which is a mean square error (MSE) loss between the predicted operator powers $\widehat{OP}_{1:M}$ and the corresponding ground truth. Additionally, we introduce the loss function $\mathcal{L}_{\boldsymbol{r}}$ to ensure consistent outputs when the inputs to the relation function follow a given rule. This is formulated as follows:

$$\mathcal{L}_{\boldsymbol{r}} = \sum_i \left(1 - sim\left(\hat{\boldsymbol{r}}_i, \boldsymbol{r}^{gt}\right)\right) \tag{24}$$

where $\hat{\boldsymbol{r}}_i$ denotes the output of the relation function for the ith relation tuple in $\mathbb{V}^2$ and $\mathbb{V}^3$. The overall loss function $\mathcal{L}$ for end-to-end training is constructed as follows:

$$\mathcal{L} = \sum \mathcal{L}_{op} + \sum \mathcal{L}_{\boldsymbol{r}} \tag{25}$$

where $\sum$ represents the sum of the loss functions across all attributes, for both binary and ternary relation functions, and for both numerical and logical rule types. By minimizing $\mathcal{L}_{op}$ and $\mathcal{L}_{\boldsymbol{r}}$ simultaneously during training, the optimization objective of the perception network f_θ, as described in (13), can be achieved. This is because, under the constraints of numerous ground truth rules ($OP^{gt}_{1:M}$ and $\boldsymbol{r}^{gt}_{Num/Lgc}$), the expected theoretical SHDR constructed from HD attribute representations in the codebooks will be a competitive representation, guiding the reasoning process toward optimality ($\mathcal{L}_{op} \to 0$ and $\mathcal{L}_{\boldsymbol{r}} \to 0$).

2) End-to-End Training With Auxiliary Attribute Labels: Following previous work, we assess the performance of the Rel-SAR model in end-to-end training using both auxiliary attribute labels and answer labels. Here, a cosine similarity loss is employed as the perception loss function to enhance the similarity between the estimated SHDR $\hat{\mathcal{S}}^{ind}$ of the perception frontend and the theoretical SHDR $\mathcal{S}$ (10) derived from attribute labels, thereby optimizing the trainable perception network f_θ (13). The perception loss function $\mathcal{L}_p$ is defined as follows:

$$\mathcal{L}_p = 1 - sim\left(\hat{\mathcal{S}}^{ind}, \mathcal{S}\right). \tag{26}$$

Meanwhile, to achieve rule learning through the optimization objective described in (21), we introduce the loss function $\mathcal{L}_{rs}$ to increase the overall similarity between the outputs $\boldsymbol{r}$ of all corresponding relation tuples in $\mathbb{V}^2$ and $\mathbb{V}^3$ (Table V), respectively. This is formulated as follows:

$$\mathcal{L}_{rs} = 1 - \prod_{i,j}^{i\neq j} sim\left(\hat{\boldsymbol{r}}_i, \hat{\boldsymbol{r}}_j\right) \tag{27}$$

where $\hat{\boldsymbol{r}}_i$ and $\hat{\boldsymbol{r}}_j$ denote the outputs of relation functions for any two distinct sets of relation tuples in $\mathbb{V}^2$ and $\mathbb{V}^3$, including attribute sets from the answer panel y. Therefore, the overall loss function $\mathcal{L}$ for end-to-end training with auxiliary attribute labels is constructed as follows:

$$\mathcal{L} = \sum \mathcal{L}_p + \sum \mathcal{L}_{rs} \tag{28}$$

where the former $\sum$ represents the sum of the loss functions across all context panels and the answer panel, while the latter $\sum$ represents the sum of the loss functions across all attributes.

Additionally, similar to other neuro-symbolic approaches, the perception frontend and reasoning backend in Rel-SAR

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

TABLE VI
TEST ACCURACY (%) ON THE RAVEN, I-RAVEN, AND RAVEN-FAIR DATASETS

	Method	RAVEN								I-RAVEN								RAVEN-FAIR
		Avg	Center	2X2	3X3	L-R	U-D	O-IC	O-IG	Avg	Center	2X2	3X3	L-R	U-D	O-IC	O-IG	Avg
DL	WReN [3]	14.7	13.1	28.6	28.3	7.5	6.3	8.4	10.6	23.8	29.4	26.8	23.5	21.9	21.4	22.5	21.5	30.3
	LEN [6]	72.9	80.2	57.5	62.1	73.5	81.2	84.4	71.5	41.4	56.4	31.7	29.7	44.2	44.2	52.1	31.7	50.9
	CoPINet [5]	91.4	95.1	77.5	78.9	99.1	99.7	98.5	91.4	46.1	54.4	36.8	31.9	51.9	52.5	52.2	42.8	50.6
	MXGNet [40]	84.0	94.3	60.5	64.9	96.6	96.4	94.1	81.3	33.1	40.7	27.9	24.7	35.8	34.5	36.4	31.6	35.1
	SCL [30]	91.6	98.1	91.0	82.5	96.8	96.5	96.0	80.1	95.0	99.0	96.2	89.5	97.9	97.1	97.6	87.7	90.1
	SRAN [7]	54.3	—	—	—	—	—	—	—	60.8	78.2	50.1	42.4	70.1	70.3	68.2	46.3	72.9
	Rel-Base [32]	91.7	97.6	85.9	86.9	93.5	96.5	97.6	83.8	91.1	—	—	—	—	—	—	—	93.5
	MRNet [8]	74.4	96.2	49.1	45.9	93.7	94.2	92.5	51.3	75.0	96.8	45.6	39.6	95.7	95.9	95.6	55.5	88.4
	DCNet [27]	93.6	97.8	81.7	86.7	99.8	99.8	99.0	91.5	49.4	57.8	34.1	35.5	58.5	60.0	57.0	42.9	56.1
	STSN [31]	89.7	—	—	—	—	—	—	—	95.7	98.6	96.2	88.8	98.0	98.8	97.8	92.0	95.4
	PredRNet [29]	95.8	—	—	—	—	—	—	—	96.5	—	—	—	—	—	—	—	97.1
	DRNet [28]	96.9	—	—	—	—	—	—	—	97.6	—	—	—	—	—	—	—	97.6
	ACT [41]	98.3	—	—	—	—	—	—	—	96.4	—	—	—	—	—	—	—	99.3
	RS-TRAN [42]	98.4	99.8	99.7	95.4	99.2	99.4	99.9	95.4	98.7	100	99.3	96.7	99.9	99.9	99.9	97.3	—
	CRAB [43]	94.7	96.6	95.8	90.2	93.9	95.3	94.6	96.6	95.9	97.1	93.3	91.1	96.7	97.5	98.9	96.9	—
	Valen [44]	99.5	100	99.7	98.2	99.9	99.9	99.9	98.8	99.7	100	99.8	99.2	100	100	99.9	99.2	—
NSA	PrAE [9]	65.0	76.5	78.6	28.6	90.1	90.9	48.1	42.6	77.0	90.5	85.4	45.6	96.3	97.4	63.5	60.7	—
	ALANS [18]	74.4	69.1	80.2	75.0	72.2	73.3	76.3	74.9	78.5	72.3	79.5	72.9	79.2	79.6	85.9	79.9	—
	NVSA [11]	87.7	99.7	93.5	57.1	99.8	99.7	98.6	65.4	88.1	99.8	96.2	54.3	100	99.9	99.6	67.1	—
	Rel-SAR (Ours)	96.5	99.1	95.7	96.2	99.6	99.6	99.1	86.2	98.0	99.8	97.1	98.1	100	100	99.9	90.9	98.3
NSA (Attr.)	NVSA [11]	98.5	100	99.4	96.3	100	100	100	93.9	99.0	100	99.5	97.1	100	100	100	96.4	—
	Xu et al. [45]	92.9	98.8	91.9	93.1	99.2	99.1	98.2	70.1	93.2	99.5	89.6	89.7	99.7	99.5	99.6	74.7	—
	ALANS-V [18]	94.4	98.4	91.5	87.0	97.3	96.4	97.3	93.2	93.5	98.9	85.0	83.2	90.9	98.1	99.1	89.5	—
	Rel-SAR (Ours)	96.6	97.9	94.3	96.6	99.0	98.7	97.9	92.2	98.1	98.5	96.7	97.8	99.4	99.8	99.2	95.6	97.6

Note: DL denotes deep learning models, NSA denotes neuro-symbolic approaches, and NSA (Attr.) indicates methods that use attribute labels. On the RAVEN-FAIR dataset, most methods report only the average accuracy rather than per-configuration accuracies; therefore, only the average accuracy is reported in this table.

can also be trained independently using the loss functions $\mathcal{L}_p$ and $\mathcal{L}_{rs}$, respectively.

V. EXPERIMENTS

A. Datasets and Experiment Setup

We evaluate our model on the RAVEN [16], I-RAVEN [7], and RAVEN-FAIR [8] datasets. Each dataset contains 70,000 RPM instances, split into training, validation, and test sets with a ratio of 6:2:2.

A ResNet-50 is adopted as the image encoder (f_θ) to map image panels $\mathcal{X}$ into their SHDR representations $\mathcal{S} \in \mathbb{R}^d$, where the vector dimensionality is fixed to $d = 3000$. Two five-layer fully connected networks are used as rule learners, namely the numerical rule learner (f_ϕ^{Num}) and the logical rule learner (f_φ^{Lgc}), which extract the operator powers $OP_{1:M}$ from the updated *position* representations and the remaining attribute representations (*number*, *type*, *size*, and *color*), respectively. In the backend codebook, existence vectors are implemented using HRR, as the normalization inherent to FHRR during superposition would invalidate logical reasoning. All other vectors are represented using FHRR. Overall, Rel-SAR contains 79.38 M parameters; detailed network architectures and parameter counts for each component are provided in Table S2 of the Supplementary Materials.

We train the model using the AdamW optimizer with a learning rate of 1×10^{-4} and a weight decay of 1×10^{-4}. The training is performed with a batch size of 16 for 50 epochs. All experiments are conducted on a single NVIDIA TITAN RTX GPU. During training, the peak GPU memory consumption is approximately 18 226 MB, and the total training time is about 24 h. At inference time, with a batch size of 16, evaluating 2000 RPM instances takes approximately 24 s in total, corresponding to an average inference latency of about 12 ms per instance.

B. End-to-End Learning Results

We first evaluate Rel-SAR on the RAVEN [16], I-RAVEN [7], and RAVEN-FAIR [8] datasets under an end-to-end learning setting. The results, summarized in Table VI, compare Rel-SAR with both deep neural network methods and neuro-symbolic methods. Rel-SAR achieves an average accuracy of 96.5% on RAVEN, 98.0% on I-RAVEN, and 98.3% on RAVEN-FAIR, which are slightly lower than those of the previously best-performing deep model, Valen (99.5% on RAVEN and 99.7% on I-RAVEN). In contrast, compared with the prior neuro-symbolic method NVSA, Rel-SAR demonstrates a significant accuracy improvement on configurations involving *position*-based rules, including 2×2 grid (+ 2.2%), 3×3 grid (+ 39.1%), and O-IG (+ 20.8%). These gains result in an average accuracy improvement of 8.8% on RAVEN and 9.9% on I-RAVEN. We attribute these improvements to the effectiveness of the proposed periodic and logical HD representations for the *position* attribute, as well as the numerical and logical relation functions employed for rule abduction and execution. Moreover, Rel-SAR utilizes only the answer panel during training, without leveraging information from other candidate panels. This ensures that Rel-SAR does not exploit potential defects in the RAVEN dataset for shortcut learning [7], resulting in similar accuracy on the RAVEN, I-RAVEN, and RAVEN-FAIR datasets.

In addition, since neuro-symbolic models are hybrid architectures that disentangle perception and reasoning, they are often trained with auxiliary attribute labels. We therefore evaluate Rel-SAR on three datasets under a setting with additional attribute labels. The results, presented in the bottom section of Table VI, show that Rel-SAR attains average accuracies of 96.6% on RAVEN, 98.1% on I-RAVEN, and 97.6% on RAVEN-FAIR. Among the compared neuro-symbolic approaches, ALANS learner is the most similar to Rel-SAR, ALANS learner is the most closely related to Rel-SAR, as

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

TABLE VII
VISUAL PERCEPTION ACCURACY ON RAVEN DATASET

Method	Avg.	Center	2X2	3X3	L-R	U-D	O-IC	O-IG
PrAE	85.27	88.65	93.56	73.95	100	100	94.23	46.25
Xu et al.	96.10	100	100	99.99	100	100	99.96	72.78
NVSA	99.76	100	99.83	98.61	99.97	99.96	99.97	99.95
Rel-SAR	99.99	100	100	99.99	100	100	100	99.94

both incorporate learnable parameters within their reasoning backends. Compared with ALANS, Rel-SAR achieves average accuracy improvements of 2.2% on RAVEN and 4.6% on I-RAVEN. In contrast, the reasoning backends in NVSA and Xu's model rely on predesigned rule templates or algebraic invariance modules without learnable parameters. NVSA, with its comprehensive rule template library, achieves near-perfect accuracy when trained with auxiliary attribute labels. Although Rel-SAR slightly underperforms NVSA when trained with auxiliary attribute labels, it exhibits an important advantage: when trained without attribute labels, both NVSA and ALANS learner suffer significant performance drops, whereas Rel-SAR maintains nearly identical performance. This consistency highlights the robust synergy between the perception and reasoning modules in Rel-SAR.

C. Perception Results Learned With the Attribute Labels

Similar to other neuro-symbolic methods, the perception frontend in Rel-SAR can also be independently trained with attribute labels using the perception loss function (26). By querying the estimated HD attribute representations for the highest similarity with the attribute vectors in the frontend codebooks, we can retrieve the predicted entity-level attributes. A panel's features are considered correctly extracted only if the attributes of all objects in the panel match those provided by the dataset. As shown in Table VII, the evaluation results demonstrate that the perception frontend in our model achieves an average panel accuracy of 99.99% on the RAVEN dataset, maintaining nearly perfect performance, consistent with NVSA. Notably, NVSA reports a resolution issue in the RAVEN dataset, where some objects in the inner regions of the O-InGrid configuration have a different size attribute but the same image representation [11]. To address this, we adopt a similar solution to NVSA, merging classes with different sizes but identical panel representation.

To evaluate the generalization capability of the perception frontend of our model, we adopt the experimental settings outlined in [11] to test the model on unseen combinations of attribute values. Specifically, we focus on the single-object case in the 2×2 grid configuration. As shown in Table VIII, we select two attributes (e.g., *position* and *type*) along with partial value sets for each (e.g., *position* $\in \{0, 3\}$ and *type* $\in \{0, 2\}$) as the target attributes. Panels containing these target attribute values are included in the training set, while panels lacking them are designated for the test set. This approach ensures that the attribute sets of the training and test datasets are entirely disjoint, enabling a rigorous evaluation of the module's ability to generalize beyond the observed data.

In this experiment, we observe that our perception frontend struggles to identify unseen combinations of attribute values when the value vectors $\boldsymbol{v}$ in (10) are encoded in SE. We attribute this issue to the orthogonality of vectors in SE,

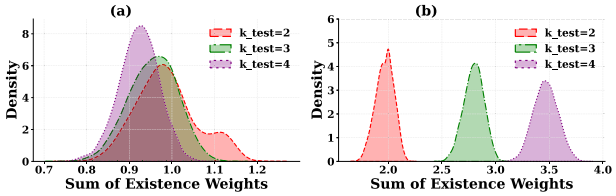


Fig. 5. Attention weights for *existence* on 2×2 grid. For the *existence* attribute, the presence weights $[W_j^{exist}(1)$ in (15)] represent the probability of an object being present at the jth position. The sum of these weights provides an approximate indication of the total number of objects in a panel. (a) When $k_{train} = 1$, the model consistently predicts the presence of only one object in the panel, regardless of the actual test scenario. (b) When $k_{train} = 2$, the model learns to distinguish between panels with varying numbers of objects, effectively adapting its feature extraction process.

which hinders the model's ability to recognize adjacency concepts. To address this, we replace the SE vectors with FPE vectors based on a Gaussian kernel, which introduces progressive similarity between vectors [20], [21]. As shown in Table VIII, our model demonstrates better generalization on unseen attribute combinations compared to NVSA [11] when using FPE vectors with progressive similarity (NE). However, the type–size combination still shows low accuracy, likely because there is no clear continuous progression between adjacent attribute values of type (e.g., triangle and square).

We also conduct an experiment to evaluate the generalization ability of our perception frontend when applied to an unseen number of objects [11]. The training set consists of panels with a finite number of objects (e.g., $k_{train} = 1$), while the test set consists of panels with a larger number of objects (e.g., $k_{test} = 2$). As shown in Table IX, when $k_{train} = 1$, our model fails to accurately predict the attributes of panels containing more objects. By analyzing the predicted attribute values, we observe that our model exhibits a consistent distribution of predicted attribute attention weights across different test sets, attributable to the limited diversity [Fig. 5(a)]. However, as the number of objects in the training panels increases, the model learns more complex patterns and can correctly allocate attention to different positions [Fig. 5(b)]. Our model achieves perfect generalization performance when $k_{train} = 2$ for 2×2 grid and when $k_{train} = 3$ for 3×3 grid.

D. Reasoning Results Utilizing the Attribute Labels

We analyze the performance of the reasoning backend of our model on the I-RAVEN dataset. Following the assumption of perfect perception in prior works [19], [33], we also utilize the ground truth attribute labels provided by the I-RAVEN dataset to generate HD attribute representations. The model is trained using the reasoning loss function (27), and the corresponding evaluation results are shown in Table X. We compare our model with the LLM-based GPT-3 [46], the deep neural network SCL [30], neuro-symbolic-based methods PrAE [9] and NVSA [11], and VSA-based methods Learn-VRF [33] and ARLC [19]. The performance of GPT-3 is reported in [33]. Our method achieves an accuracy of 99.2%, representing a 6.8% improvement over the previous state-of-the-art model, ARLC. Notably, our approach significantly outperforms prior VSA-based method ARLC on the 2×2 grid (+ 14.9%), 3×3 grid (+ 17.7%), and O-InGrid (+ 9.4%) configurations, which involve rules on the *position* attribute. This improvement demonstrates that the periodic and logical HD representations

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

TABLE VIII

ACCURACY OF ATTRIBUTE–VALUE GENERALIZATION ON 2×2 GRID CONTAINING $k = 1$ OBJECT

Training combinations	Testing combinations	Train sample	Test Sample	NVSA	Rel-SAR (SE)	Rel-SAR (NE)
Position $\in \{0,3\}$ OR Type $\in \{0,2\}$	Position $\notin \{0,3\}$ AND Type $\notin \{0,2\}$	6720	2880	0.0	0.0	26.8
Position $\in \{0,3\}$ OR Size $\in \{1,5\}$	Position $\notin \{0,3\}$ AND Size $\notin \{1,5\}$	6400	3200	15.1	0.0	31.4
Position $\in \{0,3\}$ OR Color $\in \{0,3,6,8\}$	Position $\notin \{0,3\}$ AND Color $\notin \{0,3,6,8\}$	6720	2880	34.8	0.0	73.5
Type $\in \{0,2\}$ OR Size $\in \{1,5\}$	Type $\notin \{0,2\}$ AND Size $\notin \{1,5\}$	5760	3840	0.0	0.0	13.9
Type $\in \{0,2\}$ OR Color $\in \{0,3,6,8\}$	Type $\notin \{0,2\}$ AND Color $\notin \{0,3,6,8\}$	6144	3456	72.0	0.0	89.5
Size $\in \{1,5\}$ OR Color $\in \{0,3,6,8\}$	Size $\notin \{1,5\}$ AND Color $\notin \{0,3,6,8\}$	5760	3840	29.3	0.0	82.6

TABLE IX

GENERALIZATION TO A GROWING NUMBER OF UNSEEN OBJECTS IN THE RAVEN PANEL

Training Configuration	Training Samples	k_{train}	k_{test}								Avg	ODD Avg
			2	3	4	5	6	7	8	9		
2x2Grid	9600	1	0.3	0.0	0.0	—	—	—	—	—	0.1	0.1
2x2Grid	19200	2	100.0	100.0	100.0	—	—	—	—	—	100.0	100.0
3x3Grid	21600	1	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3x3Grid	43200	2	100.0	100.0	99.4	70.3	21.6	0.4	0.0	0.0	49.0	41.7
3x3Grid	64800	3	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
3x3Grid	86400	4	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

TABLE X

IN-DISTRIBUTION ACCURACY ON THE I-RAVEN DATASET

Method	Avg	Center	2X2	3X3	L-R	U-D	O-IC	O-IG
GPT-3 [46]	86.5	86.4	83.2	81.8	83.4	84.6	92.8	93.0
SCL [30]	84.3	99.9	68.9	43.0	98.5	99.1	97.7	82.6
PrAE [9]	71.1	83.8	82.9	47.4	94.8	94.8	56.6	37.4
NVSA [11]	88.1	99.8	96.2	54.3	100	99.9	99.6	67.1
Learn-VRF [33]	79.5	97.7	56.3	49.9	94.0	95.6	98.3	64.8
ARLC [19]	92.4	98.4	83.4	80.0	98.7	98.4	98.8	89.4
Rel-SAR	99.2	99.9	98.3	97.7	100.0	99.9	100.0	98.8

TABLE XI

OOD ACCURACY ON UNSEEN RULE–ATTRIBUTE PAIRS ON I-RAVEN

	Type			Size				Color			
	Const.	Progr.	Dist.3	Const.	Progr.	Arth.	Dist.3	Const.	Progr.	Arth.	Dist.3
GPT-3 [46]	88.5	86.0	88.6	93.6	93.2	92.6	71.6	94.2	94.7	94.3	65.8
Learn-VRF [33]	100	100	99.7	100	100	99.8	99.8	100	98.8	100	100
ARLC [19]	100	98.6	99.7	100	100	99.6	99.6	100	100	100	99.8
Rel-SAR	99.7	99.6	100	99.3	99.6	97.9	100	96.8	89.5	86.4	70.7

of *position*, along with the corresponding relation functions, effectively handle the reasoning of *arithmetic* and *progression* rules of *position*.

We further evaluate the out-of-distribution (OOD) generalization ability of our reasoning backend, following the experimental setup described in [33]. In this evaluation, a specific rule (e.g., *type*) for an attribute (e.g., *const.*) is designated as the target rule. The model is trained and validated using samples that exclude the target rule and is then tested on samples containing only the target rule. All experiments are conducted on the center configuration, while training, validation, and test sets are filtered from the I-RAVEN dataset. The corresponding evaluation results are shown in Table XI. For attributes *type* and *size*, our model—consistent with Learn-VRF [33] and ARLC [19]—exhibits a near-perfect ability to generalize to unseen attribute rules. This capability arises from the use of unified HD vectors to represent values across different attributes, facilitating rule transfer between attributes. However, our model exhibits relatively limited performance on the *color* attribute. This could be due to the fact that *color* has a broader range of attribute values, making it difficult to transfer rules learned from attributes with narrower value ranges, such as *type* (0–4) and *size* (0–5), to *color* (0–9) attribute.

VI. CONCLUSION AND FUTURE DIRECTIONS

In this work, we propose Rel-SAR, a novel model that leverages VSA algebra to facilitate systematic rule abduction and execution. Rel-SAR adopts an NSA, where the perception frontend extracts diverse HD attribute representations with intrinsic algebraic properties, and the reasoning backend systematically derives a variety of rules based on numerical and logical relation functions. Extensive experiments demonstrate that Rel-SAR achieves superior performance while offers better interpretability and transparency on RPM tasks.

The perception frontend of Rel-SAR effectively extracts object-level attributes while preserving the structure information of the image through the binding and bundling operations of VSA. By utilizing FPE vectors with progressive similarity, our model demonstrates its ability to generalize beyond the observed data. However, Rel-SAR exhibits relatively low accuracy on the O-IG configuration, which may be attributed to the small object size and resolution issue. As noted in [29], CNNs with large kernel sizes or more stacked blocks are less effective at extracting features from RPM images with smaller objects. NVSA further shows that reducing the stride from 2 to 1 in the first convolutional block and removing the max-pooling layer in ResNet-18 can improve accuracy [11]. We believe that using CNNs with smaller kernel sizes and strides in the shallow layers will enhance the accuracy on the O-IG configuration.

Consistent with other neuro-symbolic methods, we also introduce auxiliary rule labels during training. This is because, without precise rule-driven guidance, the model struggles to learn meaningful SHDR of attributes. We note that slot attention [47] enables unsupervised scene decomposition, while VQ-VAE [48] learns discrete latent representations, disentangling different concepts (i.e., attributes in RPM) from raw images. Therefore, we suggest employing a learnable frontend codebook, combined with slot attention to enable unsupervised extraction of SHDR from raw images in the future work.

The reasoning backend of Rel-SAR implements systematic abductive reasoning based on diverse HD attribute

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

representation and relation representations. Our model has a significant improvement on configurations with rules on *position*. This demonstrates the effectiveness of PE and BE for *position* attribute. Although experimental results indicate that Rel-SAR demonstrates strong attribute–value generalization, scalability to increasing numbers of unseen objects, and robust in-distribution generalization, its reasoning backend exhibits limited OOD generalization compared with Learn-VRF [33] and ARLC [19]. This limitation likely stems from the rule learner, which is implemented as a multilayer fully connected neural network and lacks the capacity to generalize across attribute domains with substantially different value ranges. In particular, it struggles to transfer relational rules learned from attributes with relatively narrow ranges (e.g., size and type) to attributes with much wider ranges, such as color. Recent studies increasingly emphasize reasoning over relations among perceptual inputs, rather than over the absolute features of individual inputs [34], [35]. This perspective is formalized by the relational bottleneck principle [17], which seeks to attenuate the dependence of relational reasoning on the absolute magnitude of attribute values. Future work may therefore explore integrating relational bottlenecks with VSA algebra, while drawing on novel ideas from Learn-VRF [33] and ARLC [19], to model more complex relations and simultaneously maintain strong robustness to OOD generalization.

REFERENCES

[1] P. A. Carpenter, M. A. Just, and P. Shell, "What one intelligence test measures: A theoretical account of the processing in the raven progressive matrices test," *Psychol. Rev.*, vol. 97, no. 3, p. 404, 1990.

[2] M. Mitchell, "Abstraction and analogy-making in artificial intelligence," *Ann. New York Acad. Sci.*, vol. 1505, no. 1, pp. 79–101, 2021.

[3] D. G. T. Barrett, F. Hill, A. Santoro, A. S. Morcos, and T. Lillicrap, "Measuring abstract reasoning in neural networks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 511–520.

[4] F. Hill, A. Santoro, D. G. T. Barrett, A. S. Morcos, and T. Lillicrap, "Learning to make analogies by contrasting abstract relational structure," 2019, *arXiv:1902.00120*.

[5] C. Zhang, B. Jia, F. Gao, Y. Zhu, H. Lu, and S.-C. Zhu, "Learning perceptual inference by contrasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[6] K. Zheng, Z.-J. Zha, and W. Wei, "Abstract reasoning with distracting features," *Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[7] S. Hu, Y. Ma, X. Liu, Y. Wei, and S. Bai, "Stratified rule-aware network for abstract visual reasoning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 2, pp. 1567–1574.

[8] Y. Benny, N. Pekar, and L. Wolf, "Scale-localized abstract reasoning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12552–12560.

[9] C. Zhang, B. Jia, S.-C. Zhu, and Y. Zhu, "Abstract spatial–temporal reasoning via probabilistic abduction and execution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 9736–9746.

[10] G. F. Marcus, *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. Cambridge, MA, USA: MIT Press, 2003.

[11] M. Hersche, M. Zeqiri, L. Benini, A. Sebastian, and A. Rahimi, "A neuro-vector-symbolic architecture for solving Raven's progressive matrices," *Nature Mach. Intell.*, vol. 5, no. 4, pp. 363–375, Mar. 2023.

[12] E. E. Margolis and S. E. Laurence, *Concepts: Core Readings*. Cambridge, MA, USA: MIT Press, 1999.

[13] F. A. Mansouri, D. J. Freedman, and M. J. Buckley, "Emergence of abstract rules in the primate brain," *Nature Rev. Neurosci.*, vol. 21, no. 11, pp. 595–610, Nov. 2020.

[14] G. Marcus and E. Davis, "Insights for AI from the human mind," *Commun. ACM*, vol. 64, no. 1, pp. 38–41, Jan. 2021.

[15] S. T. Piantadosi et al., "Why concepts are (probably) vectors," *Trends Cognit. Sci.*, vol. 28, no. 9, pp. 844–856, Sep. 2024.

[16] C. Zhang, F. Gao, B. Jia, Y. Zhu, and S.-C. Zhu, "RAVEN: A dataset for relational and analogical visual REasoNing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5312–5322.

[17] T. W. Webb et al., "The relational bottleneck as an inductive bias for efficient abstraction," *Trends Cognit. Sci.*, vol. 28, no. 9, pp. 829–843, Sep. 2024.

[18] C. Zhang, S. Xie, B. Jia, Y. N. Wu, S.-C. Zhu, and Y. Zhu, "Learning algebraic representation for systematic generalization in abstract reasoning," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 692–709.

[19] G. Camposampiero, M. Hersche, A. Terzić, R. Wattenhofer, A. Sebastian, and A. Rahimi, "Towards learning abductive reasoning using vsa distributed representations," in *Proc. Int. Conf. Neural-Symbolic Learn. Reasoning*, 2024, pp. 370–385.

[20] T. A. Plate, *Distributed Representations and Nested Compositional Structure*. Toronto, ON, Canada: University of Toronto, Department of Computer Science, 1994.

[21] E. Frady, D. Kleyko, C. Kymn, B. Olshausen, and F. Sommer, "Computing on functions using randomized vector representations," 2021, *arXiv:2109.03429*.

[22] M. Mejri, C. Amarnath, and A. Chatterjee, "RESOLVE: Relational reasoning with symbolic and object-level features using vector symbolic processing," 2024, *arXiv:2411.08290*.

[23] M. Mejri, C. Amarnath, and A. Chatterjee, "LARS-VSA: A vector symbolic architecture for learning with abstract rules," 2024, *arXiv:2405.14436*.

[24] A. Santoro et al., "A simple neural network module for relational reasoning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–10.

[25] M. Jahrens and T. Martinetz, "Solving Raven's progressive matrices with multi-layer relation networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–6.

[26] M. Małkiński and J. Mańdziuk, "Multi-label contrastive learning for abstract visual reasoning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 2, pp. 1941–1953, Feb. 2024.

[27] T. Zhuo and M. Kankanhalli, "Effective abstract reasoning with dual-contrast network," 2022, *arXiv:2205.13720*.

[28] K. Zhao, C. Xu, and B. Si, "Learning visual abstract reasoning through dual-stream networks," in *Proc. AAAI Conf. Artif. Intell.*, 2024, vol. 38, no. 15, pp. 16979–16988.

[29] L. Yang et al., "Neural prediction errors enable analogical visual reasoning in human standard intelligence tests," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 39572–39583.

[30] Y. Wu, H. Dong, R. Grosse, and J. Ba, "The scattering compositional learner: Discovering objects, attributes, relationships in analogical reasoning," 2020, *arXiv:2007.04212*.

[31] S. S. Mondal, T. Webb, and J. D. Cohen, "Learning to reason over visual objects," 2023, *arXiv:2303.02260*.

[32] S. Spratley, K. Ehinger, and T. Miller, "A closer look at generalisation in raven," in *Proc. 16th Eur. Conf. Comput. Vis.-ECCV*, 2020, pp. 601–616.

[33] M. Hersche, F. Di Stefano, T. Hofmann, A. Sebastian, and A. Rahimi, "Probabilistic abduction for visual abstract reasoning via learning rules in vector-symbolic architectures," 2024, *arXiv:2401.16024*.

[34] A. Altabaa, T. Webb, J. Cohen, and J. Lafferty, "Abstractors and relational cross-attention: An inductive bias for explicit relational reasoning in transformers," 2023, *arXiv:2304.00195*.

[35] G. Kerg, S. Mittal, D. Rolnick, Y. Bengio, B. Richards, and G. Lajoie, "On neural architecture inductive biases for relational tasks," 2022, *arXiv:2206.05056*.

[36] D. Kleyko, D. A. Rachkovskij, E. Osipov, and A. Rahimi, "A survey on hyperdimensional computing aka vector symbolic architectures, Part I: Models and data transformations," *ACM Comput. Surv.*, vol. 55, no. 6, pp. 1–40, Jul. 2023.

[37] T. A. Plate, "Holographic reduced representations," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 623–641, May 1995.

[38] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognit. Comput.*, vol. 1, no. 2, pp. 139–159, Jun. 2009.

[39] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[40] D. Wang, M. Jamnik, and P. Lio, "Abstract diagrammatic reasoning with multiplex graph networks," 2020, *arXiv:2006.11197*.

[41] J. Kwiatkowski and K. Krawiec, "Staged self-supervised learning for raven progressive matrices," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 9, pp. 16564–16576, Sep. 2025.

[42] Q. Wei, D. Chen, and B. Yuan, "Multi-viewpoint and multi-evaluation with felicitous inductive bias boost machine abstract reasoning ability," *IEEE Trans. Image Process.*, vol. 34, pp. 667–677, 2025.

[43] F. Shi, B. Li, and X. Xue, "Abstracting concept-changing rules for solving Raven's progressive matrix problems," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jan. 21, 2026, doi: 10.1109/TPAMI.2026.3656670.

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.

[44] R. Song and B. Yuan, "Funny-Valen-tine: Planning solution distribution enhances machine abstract reasoning ability," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 6, 2026, doi: 10.1109/TNNLS.2025.3647282.
[45] J. Xu, T. Vaidya, Y. Wu, S. Chandra, Z. Lai, and K. F. E. Chong, "Abstract visual reasoning: An algebraic approach for solving Raven's progressive matrices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 6715–6724.
[46] T. B. Brown et al., "Language models are few-shot learners," 2020, *arXiv:2005.14165*.
[47] F. Locatello et al., "Object-centric learning with slot attention," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 11525–11538.
[48] A. Van Den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–10.



Zhong-Hua Sun received the B.S. degree in biomedical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2023, where he is currently pursuing the M.S. degree with the School of Life Science and Technology.

His research interests include brain-inspired computing, cognitive science, and symbolic AI.



Ru-Yuan Zhang received the B.S. degree in psychology from the Peking University, Beijing, China, in 2010, and the Ph.D. degree in brain and cognitive sciences from the University of Rochester, Rochester, NY, USA, in 2016.

He is currently a tenure-track Associate Professor with the School of Psychological and Cognitive Sciences, Peking University. He has authored or co-authored more than 50 papers in prestigious international journals and conferences in both neuroscience and machine learning, including *Nature Human Behavior*, *Proceedings of the National Academy of Sciences* (PNAS), and IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE. His current interests include human visual neuroscience, human decision making and learning, computer vision, and machine reasoning.



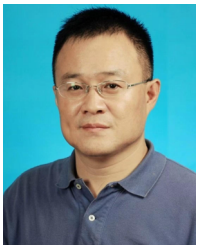
Zonglei Zhen received the B.S. degree (Hons.) in mechatronic engineering and the M.S. degree in mechatronic engineering from Tianjin University, Tianjin, China, in 2001 and 2004, respectively, and the Ph.D. degree in computer applied technology from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2007.

He is currently a Professor with the Faculty of Psychology, Beijing Normal University, Beijing. His research interests focus on the cognitive neural mechanisms and computations underlying visual perception and action, with investigations spanning both humans and artificial intelligence systems.



Da-Hui Wang received the B.S. degree in physics and the Ph.D. degree in system theory from Beijing Normal University, Beijing, China, in 1997 and 2002, respectively.

He is currently a Professor with the School of Systems Science, Beijing Normal University. His research interests focus on the fundamental theory of complex systems and its applications. In particular, he develops computational models based on neural circuits to investigate the dynamical mechanisms of neural systems and to elucidate how specific functions are achieved. His current research emphasizes whole-brain neural network modeling for cognitive tasks, aiming to uncover the global mechanisms underlying cognitive functions and the neural basis of intelligence.



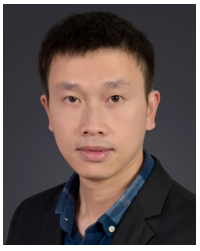
Yong-Jie Li (Senior Member, IEEE) received the Ph.D. degree in biomedical engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2004.

He is currently a Professor with the Key Laboratory for NeuroInformation of Ministry of Education, School of Life Science and Technology, UESTC. His research interests include building of biologically inspired computational models of visual perception and the applications in image processing and computer vision.



Xiaohong Wan received the B.S. degree in chemical physics and electric engineering from the University of Science and Technology of China, Hefei, China, in 1999, and the M.S. degree in applied chemistry and the Ph.D. degree in energy engineering from Tohoku University, Sendai, Japan, in 2002 and 2006, respectively.

He is currently a Full Professor with the School of Psychology, Beijing Normal University, Beijing, China. His research interests include cognitive neuroscience, cognitive computation, and cognitive intelligence.



Hongzhi You received the B.S. degree in mechanical engineering and automation from Beijing University of Posts and Telecommunications, Beijing, China, in 2008, and the Ph.D. degree in systems theory from Beijing Normal University, Beijing, in 2014.

He was a Visiting Student with the Institute of Neuroinformatics, University of Zurich and ETH Zurich, Zürich, Switzerland, from 2011 to 2013. He is currently an Associate Professor with the Biomedical Engineering Department, University of Electronic Science and Technology of China, Chengdu, China. His research interests include cognitive computation and brain-inspired computational models of visual perception and reasoning.

Authorized licensed use limited to: Peking University. Downloaded on October 25,2026 at 12:38:58 UTC from IEEE Xplore. Restrictions apply.