

# Exploring Abstract Reasoning Methods Working with Disentangled Visual Attributes

**Master Thesis**

**Author(s):**

Di Stefano, Francesco

**Publication date:**

2023

**Permanent link:**

<https://doi.org/https://doi.org/10.3929/ethz-b-000629701>

**Rights / license:**

[In Copyright - Non-Commercial Use Permitted](#)



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich

# Exploring Abstract Reasoning Methods Working with Disentangled Visual Attributes

Master Thesis

Francesco Di Stefano

August 31, 2023

Advisors: Prof. Dr. T. Hofmann, Dr. A. Rahimi, Michael Hersche

Department of Computer Science, ETH Zürich

---

## Abstract

The capacity for abstract reasoning is a cornerstone of human intelligence, and replicating this in machine learning models presents a complex challenge. This study focuses on the RAVEN dataset, a standard for assessing abstract reasoning abilities. Building upon the foundational work of Hersche et al., which employed Vector Symbolic Architectures (VSA) for reasoning, we delve deeper into making the reasoning mechanism learnable. Central to our exploration is the introduction of the AxR-Learners Framework, which incorporates the Learnable Formula. This innovative model is designed to autonomously adapt, allowing for a more flexible and tailored approach to abstract reasoning tasks. Our findings highlight the efficacy of the Learnable Formula in enhancing machine reasoning, suggesting a promising avenue for the evolution of more sophisticated artificial intelligence systems.

---

# Contents

---

|  |           |
|--|-----------|
| <b>Contents</b>  | <b>ii</b> |
| <b>1 Introduction</b>  | <b>1</b>  |
| 1.1 RAVEN Dataset . . . . .  | 2         |
| 1.1.1 Historical Context and Significance . . . . .                                | 2         |
| 1.1.2 Structure of the Dataset . . . . .   | 3         |
| 1.1.3 Attributes and Characteristics . . . . .                                     | 4         |
| 1.1.4 Rules and Configurations . . . . .   | 4         |
| 1.1.5 Dataset Generation and Annotations . . . . .                                 | 5         |
| 1.1.6 Comparative Analysis with VQA . . . . .                                      | 5         |
| 1.2 Vector Symbolic Architectures (VSA) . . . . .                                  | 5         |
| 1.2.1 Origins and Evolution of VSA . . . . .                                       | 5         |
| 1.2.2 Core Principles and Utility of VSA . . . . .                                 | 5         |
| 1.2.3 Technical Foundations of VSA . . . . .                                       | 6         |
| 1.2.4 VSA’s Connection to Symbolic AI . . . . .                                    | 6         |
| 1.2.5 Integration with Neural Approaches . . . . .                                 | 6         |
| 1.2.6 VSA’s Role in Solving RAVEN . . . . .  | 7         |
| 1.3 Holographic Reduced Representations in Symbolic Vector Architectures . . . . . | 7         |
| 1.3.1 Introduction to HRR . . . . .  | 7         |
| 1.3.2 Mathematical Foundations of HRR . . . . .                                    | 8         |
| 1.3.3 Role-filling Key-value Composition . . . . .                                 | 9         |
| 1.3.4 High-Dimensionality in VSA . . . . .   | 9         |
| 1.3.5 Creation of VSA Codebooks and Encoding in RAVEN Dataset . . . . .            | 9         |
| 1.3.6 Integration with Vector Symbolic Architectures . . . . .                     | 10        |
| 1.4 Introduction to the Neuro-vector-symbolic Architecture (NVSA)                  | 11        |
| 1.4.1 Background and Motivation . . . . .  | 11        |
| 1.4.2 The Neuro-Vector-Symbolic Architecture (NVSA): An Overview . . . . .         | 11        |

|          |  |           |
|----------|--|-----------|
| 1.4.3    | Building Upon the Foundations: The Work of Hersche et al. . . . .                    | 13        |
| 1.4.4    | Setting the Stage for Further Exploration . . . . .                                  | 13        |
| <b>2</b> | <b>Methods</b>   | <b>14</b> |
| 2.1      | Perception Module . . . . .  | 15        |
| 2.2      | Recurrent Modules Utilized in the Frameworks . . . . .                               | 16        |
| 2.2.1    | VSA Conversion Module . . . . .  | 16        |
| 2.2.2    | VSA Attribute Superposition Module . . . . .   | 17        |
| 2.2.3    | VSA Context Superposition Module . . . . .   | 17        |
| 2.2.4    | Candidate Panel Selector Module . . . . .  | 18        |
| 2.3      | 1-Learner Framework . . . . .  | 18        |
| 2.3.1    | VSA Variant . . . . .  | 19        |
| 2.3.2    | VSA with Context Superposition Variant . . . . .                                     | 20        |
| 2.4      | A-Learners Framework: Disentangling Attributes . . . . .                             | 21        |
| 2.4.1    | PMF Variant . . . . .  | 22        |
| 2.4.2    | VSA Variant . . . . .  | 23        |
| 2.4.3    | VSA with Context Superposition Variant . . . . .                                     | 24        |
| 2.5      | AxR-Learners Framework with MLP . . . . .  | 25        |
| 2.5.1    | Learner Selector Module . . . . .  | 26        |
| 2.5.2    | PMF Variant . . . . .  | 29        |
| 2.5.3    | VSA Variant . . . . .  | 30        |
| 2.5.4    | VSA with Context Superposition Variant . . . . .                                     | 31        |
| 2.6      | Integration of Learnable Formula into the AxR-Learners Framework . . . . .           | 31        |
| 2.6.1    | Translating RAVEN Rules into the VSA Space . . . . .                                 | 32        |
| 2.6.2    | Introduction to the Learnable Formula Model . . . . .                                | 33        |
| 2.6.3    | Loss Curve Characteristics and Probabilistic Interpretation . . . . .                | 35        |
| 2.6.4    | Advantages of the Learnable Formula Model . . . . .                                  | 36        |
| 2.6.5    | Model Limitations . . . . .  | 36        |
| 2.6.6    | Integration of Learnable Formula within the AxR-Learners Framework . . . . .         | 37        |
| 2.6.7    | Rule Sharing via the Learnable Formula: Unveiling the R-Learners Framework . . . . . | 38        |
| 2.7      | Comprehensive Overview of Frameworks and Variants . . . . .                          | 39        |
| <b>3</b> | <b>Experimental Results and Analysis</b>   | <b>40</b> |
| 3.1      | 1-Learner Framework . . . . .  | 40        |
| 3.1.1    | VSA Variant . . . . .  | 40        |
| 3.1.2    | VSA with Context Superposition Variant . . . . .                                     | 41        |
| 3.2      | In-depth Analysis of the A-Learners Framework . . . . .                              | 41        |
| 3.2.1    | PMF Variant: A Deep Dive . . . . .   | 41        |
| 3.2.2    | Emulating Perception Uncertainty in the PMF Variant . . . . .                        | 42        |

|          |   |           |
|----------|---|-----------|
| 3.2.3    | Exploration of Loss Functions within the PMF Variant  | 44        |
| 3.2.4    | Exploring the VSA Variant . . . . .   | 45        |
| 3.2.5    | Delving into the VSA with Context Superposition Variant   | 45        |
| 3.3      | Detailed Analysis of the AxR-Learners Framework with MLP  | 46        |
| 3.3.1    | Introduction to the PMF Variant . . . . .   | 46        |
| 3.3.2    | Rationale and Exploration of the PMF Variant with Simplified MLPs . . . . .                       | 47        |
| 3.3.3    | Detailed Analysis of the VSA Variant in the AxR-Learners Framework . . . . .                      | 47        |
| 3.3.4    | Exploration of the VSA with Context Superposition Variant in the AxR-Learners Framework . . . . . | 48        |
| 3.4      | In-depth Examination of the AxR-Learners Framework with the Learnable Formula Model . . . . .     | 48        |
| 3.5      | In-depth Analysis of Rule Sharing Across Attributes in the R-Learners Framework . . . . .         | 49        |
| 3.6      | Out-Of-Distribution (OOD) Generalization Experiments . . . . .                                    | 50        |
| <b>4</b> | <b>Conclusions</b>  | <b>53</b> |
|          | <b>Bibliography</b>   | <b>55</b> |

## Chapter 1

---

# Introduction

---

Human intelligence, with its multifaceted nature, has been a focal point of scientific inquiry for centuries [1]. A standout feature of our cognitive abilities is abstract reasoning, which enables us to discern patterns, relationships, and principles beyond immediate observations [2]. The RAVEN dataset, inspired by Raven’s Progressive Matrices (RPM) [3], serves as a benchmark for this cognitive prowess, challenging machine learning models to replicate this human capability.

As computer vision has matured, we’ve seen remarkable achievements in foundational tasks such as object recognition [4], detection [5], and tracking [6]. Yet, higher-order vision problems demanding reasoning remain a frontier [7]. The RAVEN dataset stands as a challenge in this domain, pushing artificial systems towards a deeper understanding akin to human cognition.

In addressing the RAVEN dataset, two primary approaches have emerged: end-to-end models [8] and architectures that bifurcate perception and reasoning [9]. The latter, which separates sensory processing from abstract thought, offers a promising direction, allowing for a more granular exploration of reasoning mechanisms.

Vector Symbolic Architectures (VSA) have been spotlighted as a potential tool for the reasoning module [10]. By melding symbolic and distributed computational paradigms, VSA offers a distributed representation of symbolic structures, bridging the gap between traditional symbolic reasoning and neural-based approaches [11].

A pivotal reference in our journey is the work of Hersche et al. [12]. Their methodology, which distinctly partitions perception and reasoning, serves as a beacon, underscoring the significance of modular reasoning in machine learning pipelines.

Building on Hersche et al.’s foundation, this thesis ventures into the realm

of a dynamic, adaptable reasoning system. We postulate: can the reasoning module itself evolve based on the data it encounters?

Our exploration commences with the 1-Learner Framework, a rudimentary yet essential approach that encapsulates reasoning within a singular model. This framework lays the groundwork for the subsequent architectures, highlighting the challenges and nuances of abstract reasoning.

The narrative progresses to the A-Learners Framework, an ensemble-based approach where each 'learner' is dedicated to a distinct attribute. This diversification, inspired by the distributed nature of human cognition [13], aims for a more granular understanding of the RAVEN dataset.

Our exploration culminates in the AxR-Learners Framework, where we introduce the Learnable Formula, a model designed for adaptability and a deeper understanding of the arithmetic intricacies of the RAVEN dataset. This model's continuous attribute encoding offers a dynamic approach to problem-solving, marking a significant stride in our quest for machine-based abstract reasoning.

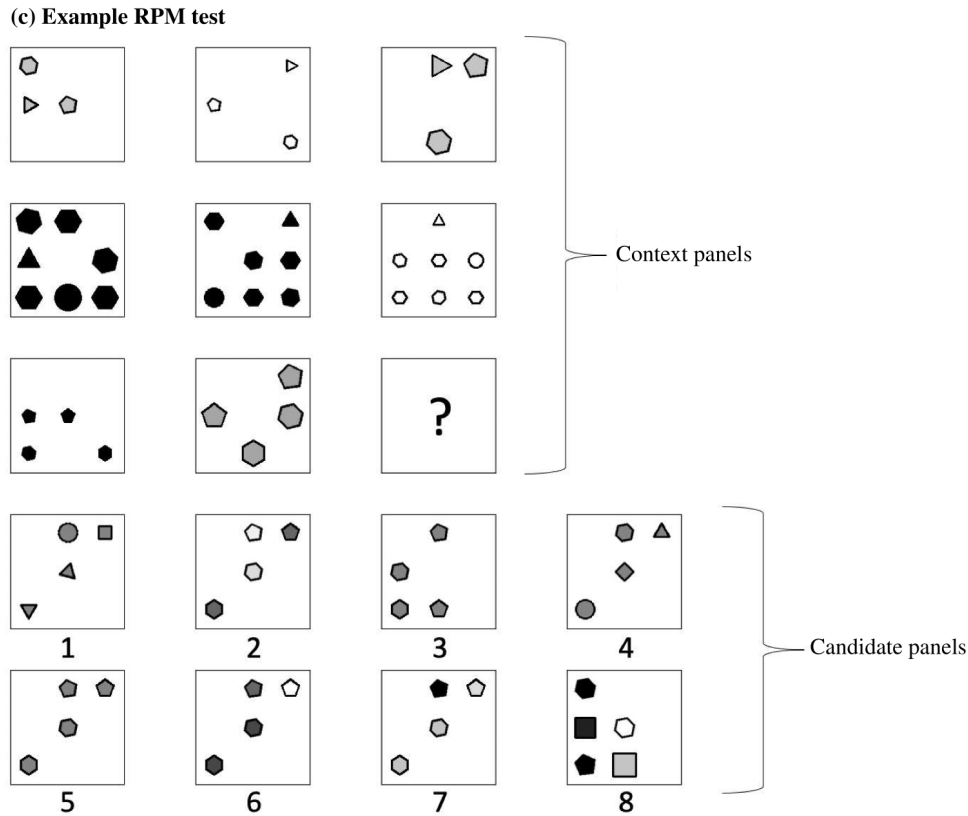
## 1.1 RAVEN Dataset

The *RAVEN* (Relational and Analogical Visual rEasoNing) dataset stands as a testament to the advancements in the domain of machine learning, specifically tailored to assess abstract reasoning capabilities. This dataset finds its roots in the principles of Raven's Progressive Matrices (RPM), a globally recognized tool for gauging fluid intelligence and abstract reasoning [3].

### 1.1.1 Historical Context and Significance

Historically, the field of computer vision has witnessed significant progress in basic vision tasks, such as object recognition, detection, and tracking [4, 5]. However, a substantial performance gap remains between artificial vision systems and human intelligence when it comes to higher-level vision problems, especially those involving reasoning. The RAVEN dataset is an endeavor to bridge this gap, aiming to challenge, debug, and improve contemporary artificial systems [14]. The dataset is inspired by John Raven's pioneering work in creating the original RPM, which is believed to be highly correlated with genuine intelligence.

## 1.1.2 Structure of the Dataset



**Figure 1.1:** An example of RPM test from the RAVEN dataset using the 3x3 grid constellation. There are eight context panels and eight answer panels. In this example, the number of objects stays constant per row. Moreover, the size values (small, medium, large) of the objects are distributed per row. Even though the shapes do not agree within a panel, they stay constant per row. The arithmetic minus rule is applied on the attribute color. Combining the detected rule leads to the correct answer panel 5.

Each RPM test in the RAVEN dataset is structured as an analogy problem presented in a  $3 \times 3$  pictorial matrix as shown in Figure 1.1. The primary objective of these tests is to discern the underlying rule set, applied row-wise, and subsequently determine the most appropriate candidate panel to complete the matrix. The dataset organizes the RPM tests into seven distinct constellations: center, 2x2 grid, 3x3 grid, left-right, up-down, out-in center, and out-in grid. Each panel within these tests has a resolution of  $160 \times 160$ . The dataset furnishes 10,000 samples for every constellation, systematically divided into six training folds, two validation folds, and two testing folds.

### 1.1.3 Attributes and Characteristics

The panels encapsulate objects characterized by multiple attributes, namely:

- **Number:** Denotes the count of objects present in a panel.
- **Position:** Refers to the spatial location of the object within the panel.
- **Type:** Distinguishes between five geometric shapes - triangle, square, pentagon, hexagon, and circle.
- **Size:** Enumerated from 1 to 6, indicating the relative size of the object.
- **Color:** Represents ten different shadings, enumerated from 1 to 10.

### 1.1.4 Rules and Configurations

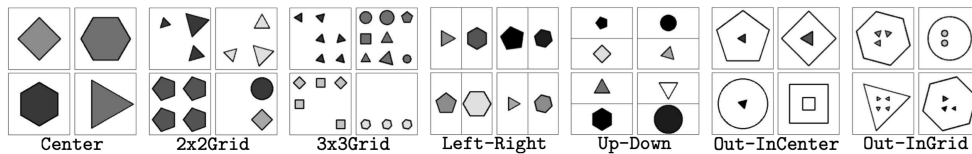


Figure 1.2: Examples of 7 different figure configurations in the RAVEN dataset.

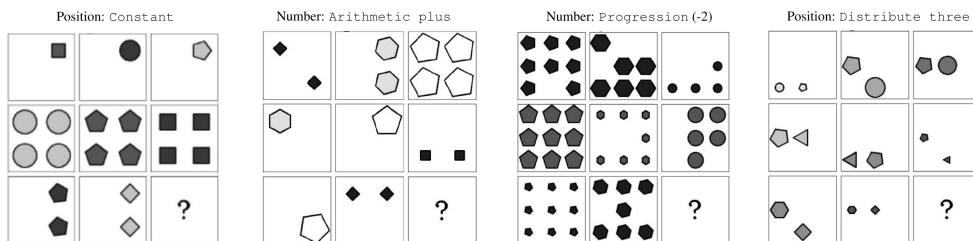


Figure 1.3: Examples for the four types of rules in RAVEN. In these examples, the rules are applied on the position attribute, or number attributes. A separate rule is applied per attribute, the displayed attribute and rule is just one of them.

The RAVEN dataset is governed by a set of rules applied to the attributes of the objects. For instance, rules might be applied to the position attribute or the number attribute of the objects. The answer choices in the RAVEN dataset are crafted such that only one randomly chosen attribute value differs from the correct answer. This introduces a shortcut solution, allowing the correct answer to be deduced by merely considering the mode of attribute values in the answer set, bypassing the need to analyze the context panels. Recognizing this limitation, the *I-RAVEN* dataset was introduced as an unbiased version, ensuring that attribute value modifications are balanced, devoid of any discernible pattern.

### 1.1.5 Dataset Generation and Annotations

The RAVEN dataset is inherently designed to be light in visual recognition but heavy in reasoning. Each image contains a limited set of simple gray-scale objects with clear-cut boundaries and no occlusion. Rules are applied row-wise, and there could be one rule for each attribute, targeting visual systems' major weaknesses in short-term memory and compositional reasoning. The dataset generation process is broken down into two stages: the first stage samples a sentence from a pre-defined Attributed Stochastic Image Grammar (A-SIG) and the second stage renders an image based on the sentence. This structured design makes the dataset diverse and easily extendable, enabling generalization tests in different figure configurations.

### 1.1.6 Comparative Analysis with VQA

Unlike Visual Question Answering (VQA) where natural language questions usually imply what to pay attention to in the image, RPM relies merely on visual clues provided in the matrix. The correspondence problem itself, i.e., finding the correct level of attributes to encode, is already a major factor distinguishing populations of different intelligence. While VQA only requires spatial and semantic understanding, RPM needs joint spatial-temporal reasoning in the problem matrix and the answer set [15, 16].

## 1.2 Vector Symbolic Architectures (VSA)

### 1.2.1 Origins and Evolution of VSA

Vector Symbolic Architectures, or VSA, emerged from the intersection of symbolic and distributed computational paradigms [10, 11]. Historically, symbolic AI, with its explicit representation of knowledge using symbols and rules, dominated the early years of artificial intelligence research [17]. However, its inability to handle uncertainty, noise, and adaptability led to the rise of distributed representations, which encode information across many dimensions, allowing for generalization and adaptability [13].

VSA was conceptualized as a bridge between these paradigms, aiming to harness the strengths of both [18]. It sought to represent symbolic structures in a distributed manner, enabling symbolic reasoning within a subsymbolic framework [19].

### 1.2.2 Core Principles and Utility of VSA

At its heart, VSA operates on high-dimensional vectors, typically in the order of thousands of dimensions [10]. These vectors are not just mere data points

but are symbolic entities that can be combined, transformed, and compared. Key features of VSA include:

- **Distributed Representation:** Symbols are represented as high-dimensional vectors, ensuring that information is spread across dimensions, leading to robustness and noise resistance [20].
- **Combinatorial Capacity:** VSA supports operations that can combine symbols, allowing the creation of composite representations. This is crucial for representing hierarchical and relational structures [11].
- **Dynamic Binding:** Symbols can be bound together to form new entities without losing their individual identities. This is achieved through operations like circular convolution [18].
- **Similarity and Dissimilarity:** VSA allows for the measurement of similarity between vectors, facilitating tasks like pattern recognition and analogy-making [19].

### 1.2.3 Technical Foundations of VSA

VSA's strength lies in its mathematical operations. The primary operations include:

- **Binding:** Typically achieved using circular convolution, it combines two vectors to produce a third, which encapsulates both parent vectors [21].
- **Bundling:** This operation accumulates vectors, often using simple vector addition, to represent sets or collections [10].

### 1.2.4 VSA's Connection to Symbolic AI

While traditional symbolic AI operates on discrete symbols and rules [17], VSA introduces fluidity by representing these symbols as vectors. This vector representation allows for operations that were challenging in a strict symbolic framework, such as generalization, analogy-making, and handling ambiguity [11]. Moreover, the high-dimensional space ensures that each symbol (or combination thereof) occupies a unique position, preserving the distinctness inherent in symbolic AI.

### 1.2.5 Integration with Neural Approaches

VSA's compatibility with neural computation is one of its standout features [20]. Neural networks, with their prowess in handling continuous data, can seamlessly operate on VSA's vector representations. This synergy allows for the training of neural models on symbolic tasks, bridging the gap between deep learning and symbolic reasoning [7, 9]. For instance, a neural network

### 1.3. Holographic Reduced Representations in Symbolic Vector Architectures

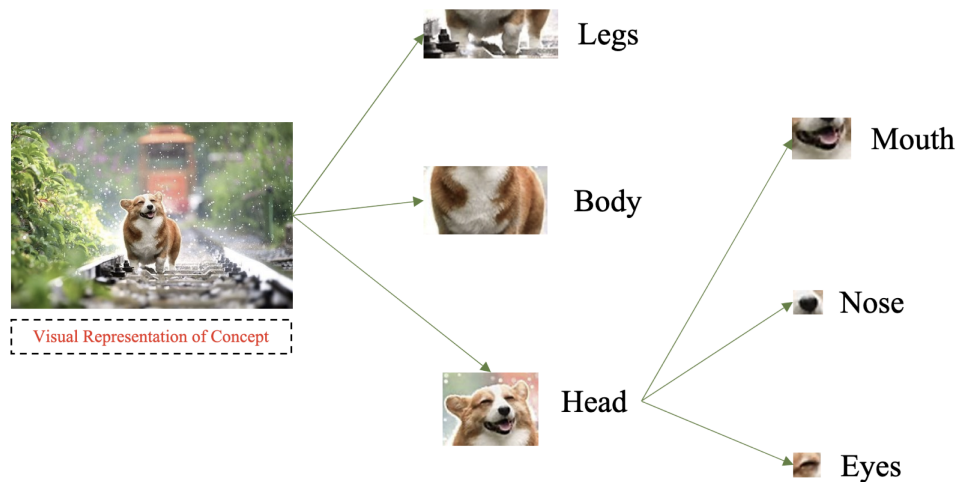
can be trained to recognize patterns in VSA-encoded data, or a recurrent network might learn sequences of symbolic operations.

#### 1.2.6 VSA's Role in Solving RAVEN

RAVEN, a benchmark that demands sophisticated visual reasoning, presented challenges that traditional neural architectures found hard to surmount [14]. VSA, with its ability to encode relational and hierarchical information as vectors, proved instrumental in addressing RAVEN [12]. By converting visual elements into VSA vectors and leveraging VSA operations to capture relationships, models could reason about intricate patterns in the RAVEN dataset. The work of Herschel et al. provides an in-depth exploration of VSA's capabilities in this context, showcasing its potential in bridging symbolic reasoning and neural computation.

### 1.3 Holographic Reduced Representations in Symbolic Vector Architectures

#### 1.3.1 Introduction to HRR



**Figure 1.4:** Sample representation of a dog, constructed from various components. This depiction follows a two-tier hierarchy, with the head further broken down into sub-elements.

Holographic Reduced Representations (HRR) are a powerful mechanism for encoding symbolic structures in distributed representations [22]. Drawing inspiration from holography, where every part of the hologram contains information about the whole, HRRs encapsulate complex structures in fixed-

### 1.3. Holographic Reduced Representations in Symbolic Vector Architectures

length vectors [10]. This allows for the representation of intricate relationships and hierarchies in a manner that's conducive to both symbolic reasoning and neural processing.

#### 1.3.2 Mathematical Foundations of HRR

##### Circular Convolution and Correlation

Circular convolution serves as the backbone of HRRs, allowing for the combination of two vectors to yield a third, composite vector [23]. For vectors  $a$  and  $b$ , their circular convolution, represented as  $a \circledast b$ , is mathematically expressed as:

$$(a \circledast b)[i] = \sum_{j=0}^{n-1} a[j] \cdot b[(i-j) \bmod n]$$

where  $n$  denotes the vector's dimensionality.

Circular correlation, on the other hand, is the inverse operation, designed to extract one vector from the convolution of two others [11]. Given  $c = a \circledast b$ , correlating  $c$  with  $a$ , symbolized as  $c \odot a$ , retrieves  $b$ . The mathematical formulation for the correlation between vectors  $a$  and  $b$  is:

$$(a \odot b)[i] = \sum_{j=0}^{n-1} a[j] \cdot b[(i+j) \bmod n]$$

The relationship between circular convolution and correlation is akin to the relationship between multiplication and division in arithmetic. While convolution binds two vectors together, correlation undoes this binding, retrieving one of the original vectors.

##### Properties of Circular Convolution and Correlation

The circular convolution operation is characterized by several pivotal properties:

- **Symmetry:**  $a \circledast b = b \circledast a$
- **Associativity:**  $(a \circledast b) \circledast c = a \circledast (b \circledast c)$
- **Neutral Element:** There exists an element  $e$  such that  $a \circledast e = a$  for every vector  $a$ .

Circular correlation, being the inverse of convolution, also has its properties:

- **Reversibility:** Given  $c = a \circledast b$ , then  $c \odot a \approx b$  and  $c \odot b \approx a$ .
- **Neutral Element:** Correlating with the neutral element  $e$  yields the original vector:  $a \odot e = a$ .

### 1.3.3 Role-filling Key-value Composition

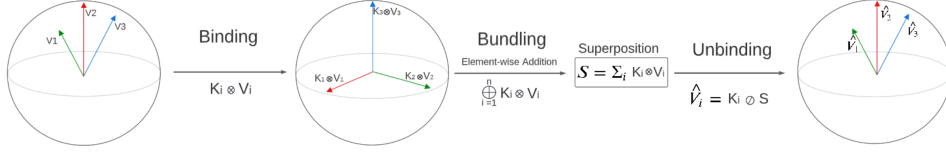


Figure 1.5: Illustration of a role-filling key-value composition.

HRRs excel in representing objects and their attributes through role-filler pairs [18]. For instance, a "car" with attributes like "color" (value "red"), "type" (value "sedan"), and "brand" (value "Toyota") can be mathematically depicted as:

$$S = \text{color} \otimes \text{red} + \text{type} \otimes \text{sedan} + \text{brand} \otimes \text{Toyota}$$

To discern the color of the car, the operation is:

$$S \otimes \text{color}^{-1} \approx \text{red}$$

### 1.3.4 High-Dimensionality in VSA

In this work, we adopted 1024-dimensional HRR vectors, divided into 4 blocks where the values within each block sum to one [10]. These vectors serve as the foundation for our VSA. The high dimensionality ensures quasi-orthogonality, which is crucial for minimizing interference during retrieval operations. Given two quasi-orthogonal vectors  $a$  and  $b$ , their superposition  $a + b$  can be deconstructed to retrieve either  $a$  or  $b$  with negligible error.

### 1.3.5 Creation of VSA Codebooks and Encoding in RAVEN Dataset

To facilitate the representation of attributes in the RAVEN dataset, we employed both discrete randomly generated VSA codebooks and continuous ones [12]. The discrete codebooks offer a set of predefined vectors, ensuring consistency across representations. In contrast, the continuous codebooks, as described by Herschel et al., allow for a more nuanced encoding of attributes, capturing the subtle variations and relationships inherent in the dataset.

#### Fractional Power Encoding for Continuous Codebooks with HRRs

The continuous codebooks, when working with HRR vectors, employ a unique approach known as fractional power encoding using circular convolution. This technique is adept at capturing continuous attributes and their nuanced variations within the HRR framework. The process can be broken down into the following steps:

### 1.3. Holographic Reduced Representations in Symbolic Vector Architectures

1. **Base Vector Selection:** For each attribute, a base vector, denoted as  $\mathbf{v}_{\text{base}}$ , is chosen. This vector serves as the foundational representation for that attribute within the HRR space.
2. **Attribute Value Transformation:** The actual value of the attribute, denoted as  $x$ , is transformed into a fractional power,  $p$ . This fractional power determines the degree to which the base vector will be modified using circular convolution.
3. **Vector Modification via Circular Convolution:** Instead of directly raising the base vector to the fractional power, the base vector is repeatedly convolved with itself  $p$  times. Mathematically, this is represented as:

$$\mathbf{v}_{\text{modified}} = \underbrace{\mathbf{v}_{\text{base}} \otimes \mathbf{v}_{\text{base}} \otimes \dots \otimes \mathbf{v}_{\text{base}}}_{p \text{ times}}$$

This operation modifies the base vector in a manner that encodes the attribute's value within the HRR framework.

By leveraging circular convolution, the fractional power encoding method ensures that HRR vectors capture continuous attributes effectively, producing vectors that are consistent in representation yet flexible enough to capture subtle variations. Through fractional power encoding, similar attribute values yield similar vectors, while distinct values produce orthogonal or near-orthogonal vectors. This method offers a nuanced encoding of attributes, capturing the subtle variations and relationships inherent in the dataset.

#### 1.3.6 Integration with Vector Symbolic Architectures

VSA emphasizes three core operations: binding (achieved through circular convolution), unbinding (realized via circular correlation), and bundling (typically through vector addition) [11]. Binding amalgamates vectors, unbinding retrieves one vector from a bound pair, while bundling facilitates the combination of multiple vectors, representing sets or collections of items.

Orthogonal key vectors play a pivotal role in this architecture [18]. Their orthogonality ensures that the binding operation yields distinct and distinguishable bound vectors. This distinctness is crucial for the unbinding operation to retrieve the correct value vector. Furthermore, orthogonality minimizes interference between vectors, ensuring that complex symbolic structures can be accurately represented and manipulated using high-dimensional vectors.

## 1.4 Introduction to the Neuro-vector-symbolic Architecture (NVSA)

### 1.4.1 Background and Motivation

The paper titled "A Neuro-vector-symbolic Architecture for Solving Raven's Progressive Matrices" [12] presents a groundbreaking approach that seamlessly integrates the capabilities of deep neural networks with the structured reasoning of vector-symbolic architectures (VSAs). This innovative architecture, termed NVSA, is a testament to the potential of marrying the strengths of both neural and symbolic AI paradigms [24]. The primary motivation behind this work is to address the inherent challenges faced by each paradigm individually [25] and to harness their combined power to tackle complex reasoning tasks, such as the Raven's Progressive Matrices (RPM) [26].

### 1.4.2 The Neuro-Vector-Symbolic Architecture (NVSA): An Overview

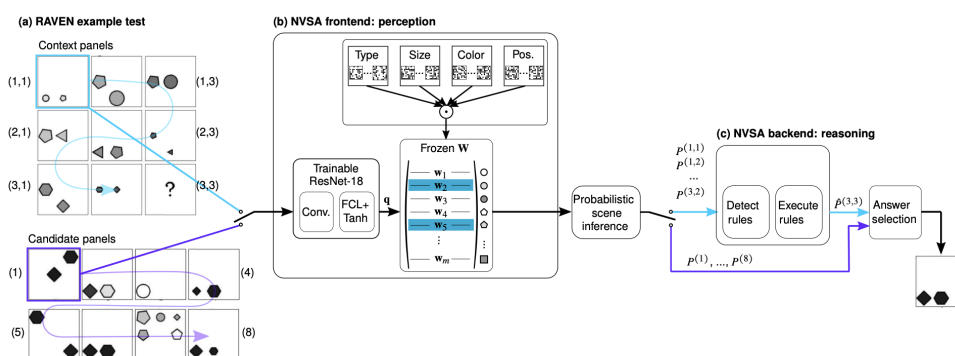


Figure 1.6: Proposed neuro-vector-symbolic architecture (NVSA).

NVSA stands as a beacon of the next generation of AI systems [27], offering a unified framework that amalgamates the computational prowess of deep learning with the logical rigor of symbolic AI [28]. By leveraging vector-symbolic architectures (VSAs) [11], NVSA provides a common representational and computational language, bridging the divide between neural processing and symbolic reasoning.

#### NVSA Frontend: The Perception Module

The perceptual frontend of NVSA is a sophisticated blend of a trainable neural network and VSA mechanisms [23]. This fusion is inspired by the unparalleled expressiveness of high-dimensional VSA representations [10]. In essence, the frontend is tasked with translating raw visual stimuli, such as

## 1.4. Introduction to the Neuro-vector-symbolic Architecture (NVSA)

images, into structured VSA representations, all the while maintaining the nuances of perceptual uncertainty.

**Translating Visual Stimuli to VSA Representations** The NVSA's modus operandi begins with the raw images undergoing processing via a neural network to distill essential features [4]. These distilled features are subsequently mapped onto VSA representations [18]. This mapping leverages a pre-defined dictionary of atomic concepts, their compositional hierarchies, and inter-relations.

### NVSA Backend: The Probabilistic Reasoning Engine

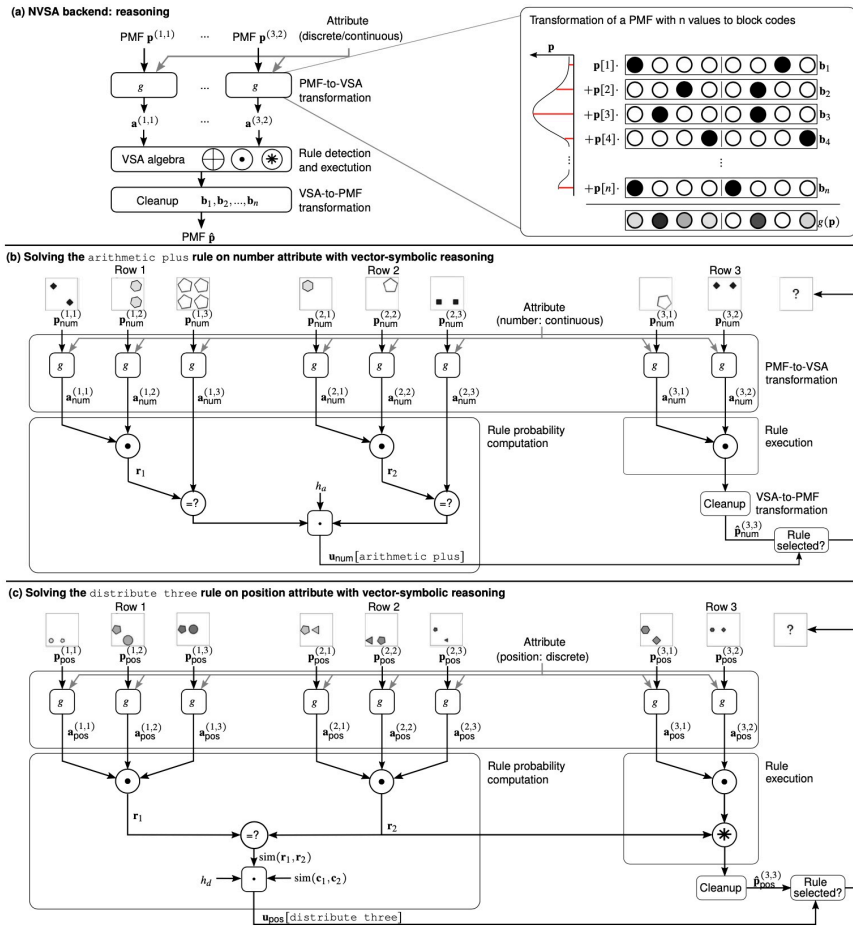


Figure 1.7: Steps involved in the NVSA reasoning.

The heart of NVSA's reasoning capabilities lies in its backend [12], meticulously designed to execute probabilistic reasoning on the VSA representations

## 1.4. Introduction to the Neuro-vector-symbolic Architecture (NVSA)

---

birthed by the frontend. The backend's genius is in its ability to transduce inferred probability mass functions into a distinct VSA vector space [11], facilitating swift and accurate probabilistic reasoning.

**The Mechanics of Probabilistic Abduction Reasoning** NVSA's symbolic logical reasoning is architected as a probabilistic abduction reasoning mechanism [29]. This form of reasoning is akin to navigating a solution space demarcated by prior background knowledge.

### 1.4.3 Building Upon the Foundations: The Work of Hersche et al.

NVSA's inception is deeply rooted in the pioneering work of Hersche et al. [12]. While Hersche and colleagues charted the initial course of integrating VSA with neural networks, NVSA refines and elevates this integration.

### 1.4.4 Setting the Stage for Further Exploration

As we embark on a journey to delve deeper into the backend system of NVSA, it's imperative to recognize the monumental significance of this architecture [12]. The subsequent sections will illuminate the intricacies of the backend, laying the groundwork for our exploration and adaptation of this system for advanced reasoning tasks.

# Methods

---

The primary objective of this research endeavor is to rigorously demonstrate that segmenting a complex problem into distinct attributes and subsequently executing parallel computations on these attributes can facilitate a higher order of abstract reasoning. To empirically validate this hypothesis, we have meticulously designed and implemented three distinct computational frameworks, each representing an incremental level of attribute disentanglement.

1. **Single-Learner Framework (1-Learner Framework):** In the inaugural framework, we initiated our experiment by harnessing the attributes as discerned by our advanced oracle perception module. However, a singular model was employed to process all these attributes collectively.
2. **Attribute-Specific Learners Framework (A-Learners Framework):** Progressing to the second framework, each attribute was paired with a unique model, thereby introducing the concept of disentanglement in reasoning. Here, the notation ' $A$ ' symbolizes the total count of attributes being processed.
3. **Attribute-Rule Specific Learners Framework (AxR-Learners Framework):** In the final and most advanced framework, we escalated the disentanglement by allocating multiple models to each attribute. Furthermore, an intelligent model selection mechanism was integrated, empowering the system to discern and select the most optimal model for a given attribute. This design was predicated on the aspiration to enable each model to master a specific rule pertinent to a particular attribute. Notably, this framework unveiled a unique category of models that, in contrast to conventional neural models, offer unparalleled interpretability and efficiency.

## 2.1 Perception Module

In the context of this research, we postulate the existence of an ‘oracle’ perception module. This module exhibits the capability to accurately segment a panel from the RAVEN suite into its inherent attributes, devoid of any uncertainty. In essence, our hypothesis revolves around a flawless perception module. The rationale behind this assumption primarily stems from our intent to underscore the potential of leveraging disentangled attribute representations to facilitate abstract reasoning. It’s noteworthy to mention that existing literature already showcases methodologies employing deep neural networks to attain optimal attribute disentanglement representations of RAVEN panels. Consequently, for the scope of this work, we presuppose a perception module adept at processing a panel image to yield five Probability Mass Function (PMF) vectors, each corresponding to a distinct attribute: Position, Number, Type, Size, and Color. Mathematically, this can be represented as:

$$f_{\text{perception}}(\mathbf{x}) = \mathbf{p}, \mathbf{n}, \mathbf{t}, \mathbf{s}, \mathbf{c} \quad (2.1)$$

Here,  $f_{\text{perception}}$  symbolizes our perception module,  $\mathbf{x}$  represents a panel in the RAVEN test, and the vectors  $\mathbf{p}$ ,  $\mathbf{n}$ ,  $\mathbf{t}$ ,  $\mathbf{s}$ , and  $\mathbf{c}$  denote the PMFs for Position, Number, Type, Size, and Color respectively. Each attribute PMF vector adheres to the following properties:

$$\mathbf{p} \in \mathbb{R}^d, \quad (2.2)$$

$$\sum_{i=0}^d p_i = 1, \quad (2.3)$$

$$\forall i p_i \in [0, 1], \quad (2.4)$$

where  $d$  signifies the dimensionality of the attribute PMF vector, which varies across attributes. It’s imperative to highlight nuances associated with the Position and Number attributes. Specifically, these attributes are pertinent solely for the 2x2 Grid and 3x3 Grid configurations, given that in other configurations, both the position and number of objects remain invariant across panels. Moreover, the dimensionality of these attributes inherently differs in these configurations due to the variable potential positions of objects within the panels and the distinct maximum object counts. For the Type, Size, and Color PMF vectors, an additional dimension is incorporated relative to their respective domains, accounting for potential attribute variations across objects within a panel. Such a scenario in an attribute is termed as the ‘Inconsistency state’. This state is feasible only when a panel houses more than one

## 2.2. Recurrent Modules Utilized in the Frameworks

object, which is exclusive to the 2x2 Grid and 3x3 Grid configurations. Tables 2.1 and 2.2 elucidate the dimensions of the attributes and their corresponding PMF vectors across different configurations, respectively.

|          | Center Single | 2x2 Grid | 3x3 Grid |
|----------|---------------|----------|----------|
| Position | 1             | 9        | 511      |
| Number   | 1             | 4        | 15       |
| Type     | 5             | 5        | 5        |
| Size     | 6             | 6        | 6        |
| Color    | 10            | 10       | 10       |

**Table 2.1:** Dimensions of each attribute in each constellation

|          | Center Single | 2x2 Grid | 3x3 Grid |
|----------|---------------|----------|----------|
| Position | 1             | 9        | 511      |
| Number   | 1             | 4        | 15       |
| Type     | 5             | 6        | 6        |
| Size     | 6             | 7        | 7        |
| Color    | 10            | 11       | 11       |

**Table 2.2:** Dimensions of each attribute PMF vector in each constellation

## 2.2 Recurrent Modules Utilized in the Frameworks

This section elucidates several pivotal components recurrently employed across the diverse frameworks under investigation.

### 2.2.1 VSA Conversion Module

The primary objective of this module is to facilitate the transformation of a PMF vector into its corresponding VSA vector, leveraging a predefined codebook. The module employs the probability values encapsulated within the PMF vector to compute a convex combination over the vectors present in the codebook. This mechanism enables the transition from the PMF space to the high-dimensional VSA space. Given an attribute PMF vector  $\mathbf{p} \in \mathbb{R}^d$ , which adheres to the properties of a PMF:

$$\sum_{i=0}^d p_i = 1, \quad (2.5)$$

$$\forall i p_i \in [0, 1], \quad (2.6)$$

and a codebook  $\mathbf{C}$  comprising  $d$  VSA vectors, the VSA Conversion Module computes the following convex combination:

$$f_{\text{VSA\_Conversion}}(\mathbf{p}, \mathbf{C}) = \sum_{i=0}^d p_i \cdot \mathbf{C}_i. \quad (2.7)$$

### 2.2.2 VSA Attribute Superposition Module

This module is designed to amalgamate the five attribute VSA vectors of a panel into a singular VSA vector, utilizing the key-value binding mechanism. Initially, five random VSA vectors are generated, each encoding a distinct attribute key. Subsequently, each attribute key undergoes binding with its corresponding attribute VSA vector. The resultant five VSA vectors are then superposed. Given a keys codebook  $\mathbf{K}$  and a panel  $\mathbf{P}$ :

$$\mathbf{k}_{\text{position}}, \mathbf{k}_{\text{number}}, \mathbf{k}_{\text{type}}, \mathbf{k}_{\text{size}}, \mathbf{k}_{\text{color}} \in \mathbf{K}, \quad (2.8)$$

$$\mathbf{a}_{\text{position}}, \mathbf{a}_{\text{number}}, \mathbf{a}_{\text{type}}, \mathbf{a}_{\text{size}}, \mathbf{a}_{\text{color}} \in \mathbf{P}, \quad (2.9)$$

the superposition is computed as:

$$f_{\text{attribute\_superposition}}(\mathbf{P}, \mathbf{K}) = (\mathbf{k}_{\text{position}} \odot \mathbf{a}_{\text{position}}) \circledast \quad (2.10)$$

$$(\mathbf{k}_{\text{number}} \odot \mathbf{a}_{\text{number}}) \circledast \quad (2.11)$$

$$(\mathbf{k}_{\text{type}} \odot \mathbf{a}_{\text{type}}) \circledast \quad (2.12)$$

$$(\mathbf{k}_{\text{size}} \odot \mathbf{a}_{\text{size}}) \circledast \quad (2.13)$$

$$(\mathbf{k}_{\text{color}} \odot \mathbf{a}_{\text{color}}). \quad (2.14)$$

From a role-filler perspective, the role is represented by the attribute, while the filler corresponds to the attribute value.

### 2.2.3 VSA Context Superposition Module

This module's primary function is to synthesize a singular VSA vector encapsulating the context, represented by all the context VSA vectors. The module can be applied to the eight context VSA vectors of a single attribute or to the eight context VSA vectors derived from the VSA Attribute Superposition Module. The operational mechanism mirrors that of the VSA Attribute Superposition Module, albeit the key-value binding is executed across VSA vectors from different context panels. Given a keys codebook  $\mathbf{K}$  and a set  $\mathbf{C}$ :

$$\mathbf{k}_i \in \mathbf{K} \forall i \in \{1, 2, 3, 4, 5, 6, 7, 8\}, \quad (2.15)$$

$$\mathbf{c}_i \in \mathbf{C} \forall i \in \{1, 2, 3, 4, 5, 6, 7, 8\}, \quad (2.16)$$

the superposition is computed as:

$$f_{\text{context\_superposition}}(\mathbf{C}, \mathbf{K}) = \bigotimes_{i=0}^8 \mathbf{k}_i \odot \mathbf{c}_i. \quad (2.17)$$

In this context, the role corresponds to the panel's position within the context, while the filler represents the value at that specific position.

#### 2.2.4 Candidate Panel Selector Module

Owing to the generative nature of the proposed frameworks, a mechanism to discern the optimal candidate panel from the proposed set, based on the generated output, becomes imperative. This module addresses this requirement by introducing a scoring function that evaluates two vectors and assigns a score. This scoring function is inversely related to the loss function employed during training:

$$f_{\text{score}}(\mathbf{o}, \mathbf{c}) = -f_{\text{loss}}(\mathbf{o}, \mathbf{c}), \quad (2.18)$$

where  $\mathbf{o}$  and  $\mathbf{c}$  can be either PMF or VSA vectors of identical dimensions. This facilitates the module's adaptability to representations using either PMF or VSA vectors. The module computes scores for all pairs  $(\mathbf{o}, \mathbf{c}_i)$ , where  $\mathbf{o}$  denotes the model's output vector and  $\mathbf{c}_i$  represents the  $i$ -th candidate vector. The panel with the highest score is selected:

$$\text{score}_i = f_{\text{score}}(\mathbf{o}, \mathbf{c}_i) = -f_{\text{loss}}(\mathbf{o}, \mathbf{c}_i), \quad (2.19)$$

$$\hat{\mathbf{y}} = \text{argmax}_i(\text{score}_i), \quad (2.20)$$

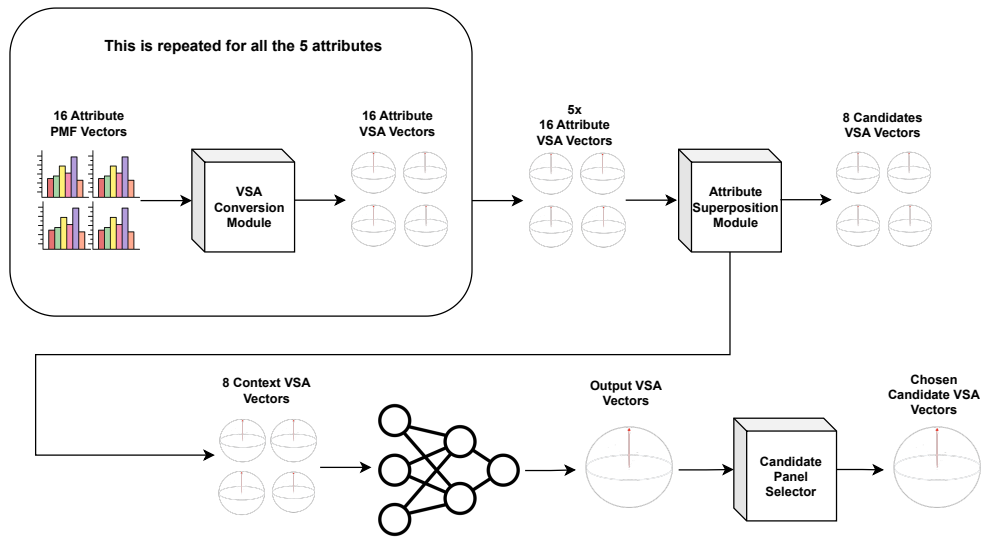
where  $\hat{\mathbf{y}}$  signifies the selected candidate panel vector. This module essentially bridges the gap between generative and discriminative models.

### 2.3 1-Learner Framework

Within the confines of this framework, we are presented with the PMF vectors corresponding to all 16 panels across the five attributes. The initial step involves the application of the VSA Conversion Module to each of

these PMF vectors, culminating in the acquisition of 16 VSA vectors for each attribute, representative of all panels. Subsequently, the VSA Attribute Superposition Module is employed on each panel's attribute VSA vectors, yielding a singular VSA vector for every panel. Post this transformation, we explored two distinct variants of this framework, with the differentiation rooted in the methodologies of feeding these eight VSA vectors into our Multi-Layer Perceptron (MLP) model.

### 2.3.1 VSA Variant



**Figure 2.1:** Schematic representation of the 1-Learner Framework, emphasizing the concatenation technique for inputting the context VSA vectors into the Multi-Layer Perceptron (MLP).

The inaugural approach, depicted in Figure 2.1, adopts a straightforward strategy of concatenating the eight context VSA vectors, as derived from the VSA Attribute Superposition Module. This methodology ensures the preservation of information integrity, as the MLP is tasked with modeling a function that comprehensively considers all eight vectors as its arguments. Subsequent to this, the MLP generates an output vector, which is then channeled through the Candidate Panel Selector Module. This module assigns a unique score to each candidate, with the highest-scoring candidate being selected. The loss computation is exclusively based on the juxtaposition of the MLP's output VSA vector with the correct candidate VSA vector, disregarding the other candidates. The Cosine Loss function is employed for this purpose:

$$\text{CosineLoss}(x, y) = 1 - \frac{x \cdot y}{\|x\| \|y\|} \quad (2.21)$$

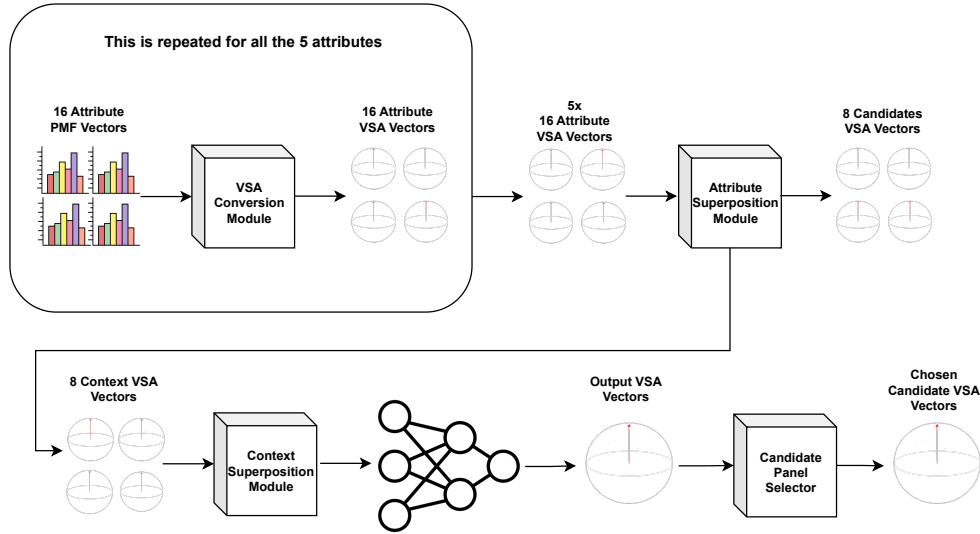
For the assignment of scores to each candidate, the Cosine Score is utilized, which is defined as the inverse of the Cosine Loss:

$$\text{CosineScore}(x, y) = 1 - \text{CosineLoss}(x, y) \quad (2.22)$$

$$= \frac{x \cdot y}{\|x\| \|y\|} \quad (2.23)$$

By computing the loss solely based on the output VSA vector and the correct candidate VSA vector, we ensure that the model and its associated framework do not depend on the candidate distribution for solution derivation.

### 2.3.2 VSA with Context Superposition Variant



**Figure 2.2:** Schematic representation of the 1-Learner Framework, emphasizing the utilization of the VSA Context Superposition Module for inputting the context VSA vectors into the Multi-Layer Perceptron (MLP).

An alternative methodology, illustrated in Figure 2.2, incorporates the VSA Context Superposition Module to derive a singular VSA vector encapsulating the entirety of the context panels. While the initial approach, as discussed previously, theoretically provides comprehensive access to the problem's information without any data loss, this secondary approach offers intriguing prospects. Specifically, it facilitates the exploration of VSA vectors and

their key-value binding mechanism, enabling the creation of fixed-dimension representations that can accommodate varying panel types within the context. This approach also presents the advantage of a consistent model structure, eliminating the need for adaptability based on input dimensions. In scenarios where VSA vector concatenation is the chosen input, the model's input layer would require adjustments to accommodate the varying dimensions resulting from the number of context panels. The remainder of the framework, particularly the candidate panel selection process, remains consistent with the previously described approach.

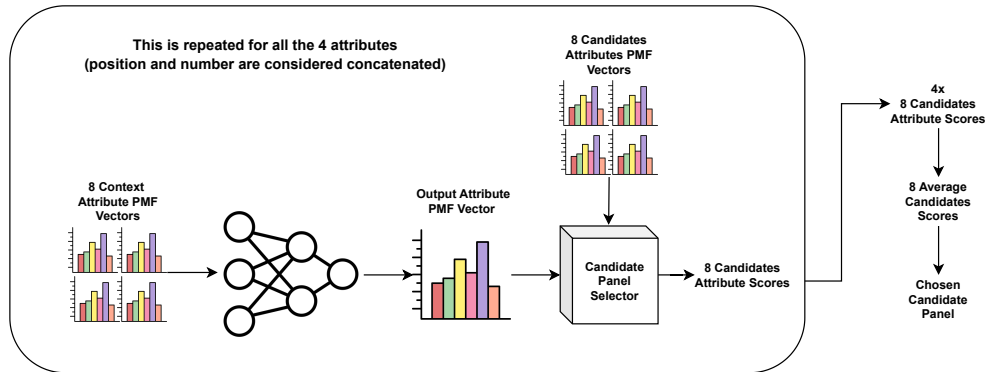
### 2.4 A-Learners Framework: Disentangling Attributes

The A-Learners Framework introduces a novel approach to attribute disentanglement. The core philosophy of this framework is to deploy distinct models for each attribute, rather than a singular model for all attributes. This approach offers several advantages:

- It facilitates modeling scenarios where different rule sets apply to individual attributes, such as the unique rules governing the Position attribute compared to other attributes.
- Complexity is distributed across multiple models, rather than being concentrated in a single model, as seen in the 1-Learner Framework.
- The framework's disentanglement allows the transformation of the VSA regression problem into four distinct multi-class classification problems. This is achieved by directly utilizing the PMF vectors for each attribute from the perception module, rather than the VSA vectors.

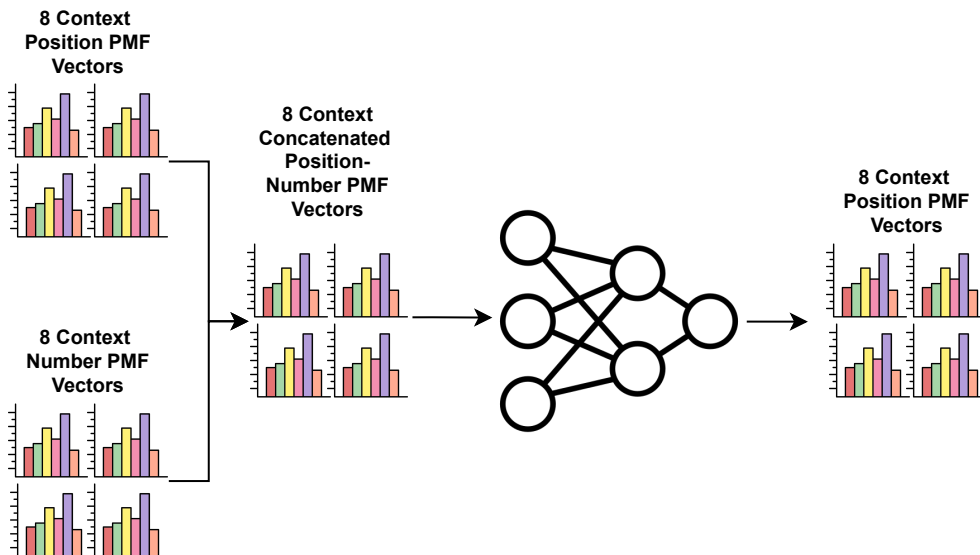
To delve deeper into the potential of this framework, we explored three distinct variants: a pure PMF-based approach, a VSA-based approach, and a hybrid approach incorporating the VSA Context Superposition Module.

### 2.4.1 PMF Variant



**Figure 2.3:** Schematic representation of the A-Learners Framework’s PMF variant. Here, each attribute’s Multi-Layer Perceptron (MLP) receives and outputs PMF vectors.

In the PMF variant, depicted in Figure 2.3, the VSA Conversion Module is bypassed. Consequently, all vectors remain as PMF vectors. Each of the four models receives PMF vectors as both input and output. A notable aspect of this variant is the combined handling of the Position and Number attributes. Their respective PMF vectors are concatenated and processed by a single MLP, which then outputs a PMF vector specific to the Position attribute domain, as illustrated in Figure 2.4.



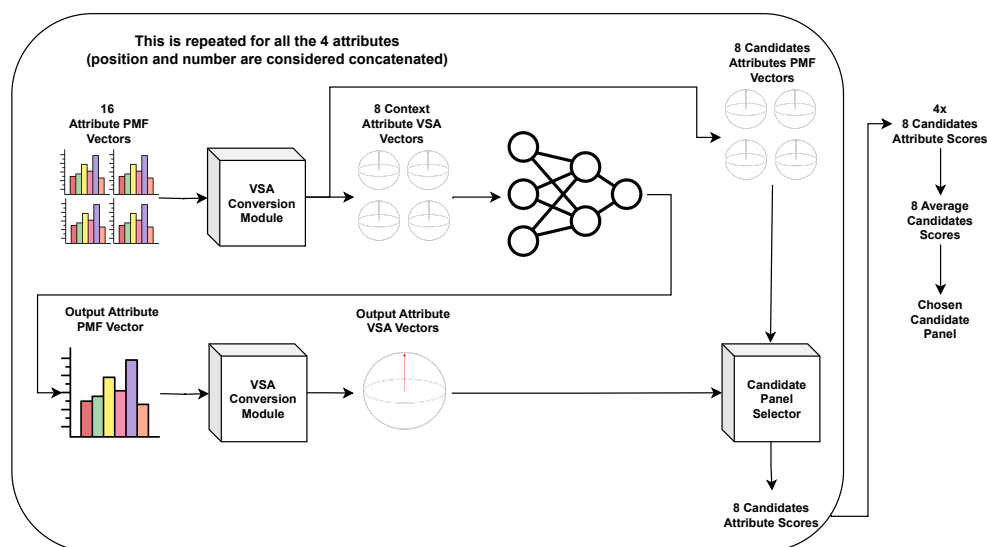
**Figure 2.4:** Methodology for inputting the Position and Number context PMF vectors in the PMF variant.

## 2.4. A-Learners Framework: Disentangling Attributes

This approach acknowledges the intertwined nature of the Position and Number attributes. In the RAVEN dataset, rules applied to the Position attribute do not affect the Number attribute, and vice versa. Moreover, the Position attribute inherently conveys information about the number of elements in a scene, obviating the need for a separate Number attribute.

The framework then proceeds to compare the candidate panel PMF vectors with the output PMF vectors for each attribute, utilizing scores computed by the Candidate Panel Selector. The final candidate is selected based on the highest average score across attributes. This multi-class classification approach allows the use of classification loss functions, specifically the Cross Entropy Loss and Kullback–Leibler Divergence Loss.

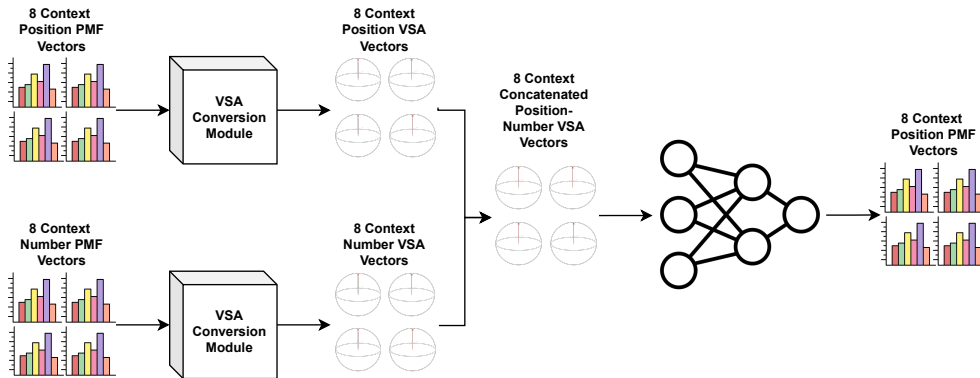
### 2.4.2 VSA Variant



**Figure 2.5:** Schematic of the A-Learners Framework's VSA variant. Here, each attribute's MLP receives VSA vectors as input, and the output PMF vectors are subsequently converted into VSA vectors.

The VSA variant, illustrated in Figure 2.5, integrates the VSA Conversion Module. Here, all panel PMF vectors are transformed into VSA vectors for each attribute. The Position and Number attributes are again jointly processed, but with a different methodology, as shown in Figure 2.6.

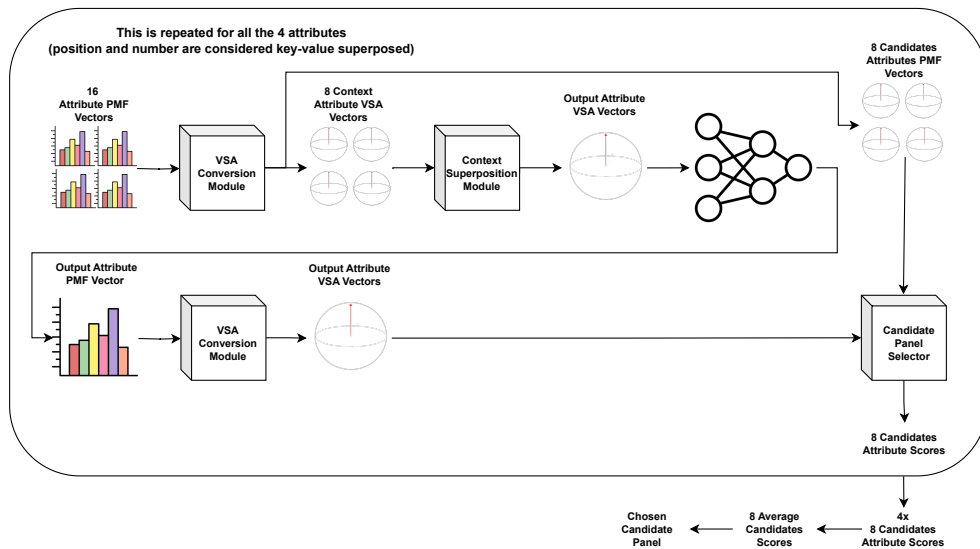
## 2.4. A-Learners Framework: Disentangling Attributes



**Figure 2.6:** Methodology for inputting the Position and Number context VSA vectors in the VSA variant.

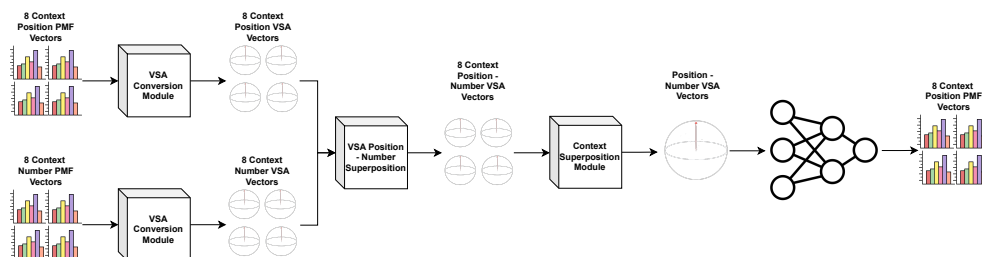
The output PMF vector from the MLP is converted into a VSA vector, facilitating comparison with the candidate VSA vectors. The remainder of the framework mirrors the PMF variant, but with the distinction of employing regression loss, specifically the Cosine Loss.

### 2.4.3 VSA with Context Superposition Variant



**Figure 2.7:** Schematic of the A-Learners Framework's VSA with Context Superposition variant. This variant employs the VSA Context Superposition Module to generate a singular VSA vector representing all context panels.

The VSA with Context Superposition variant, depicted in Figure 2.7, combines the strengths of the previous two variants. It employs VSA vectors in conjunction with the VSA Context Superposition Module. For each attribute, the context PMF vectors are first converted into VSA vectors. These vectors are then processed by the VSA Context Superposition Module to generate a singular VSA vector representing all context panels. The Position and Number attributes are again jointly processed, but using a unique approach, as illustrated in Figure 2.8.



**Figure 2.8:** Methodology for inputting the Position and Number context VSA vectors in the VSA with Context Superposition variant.

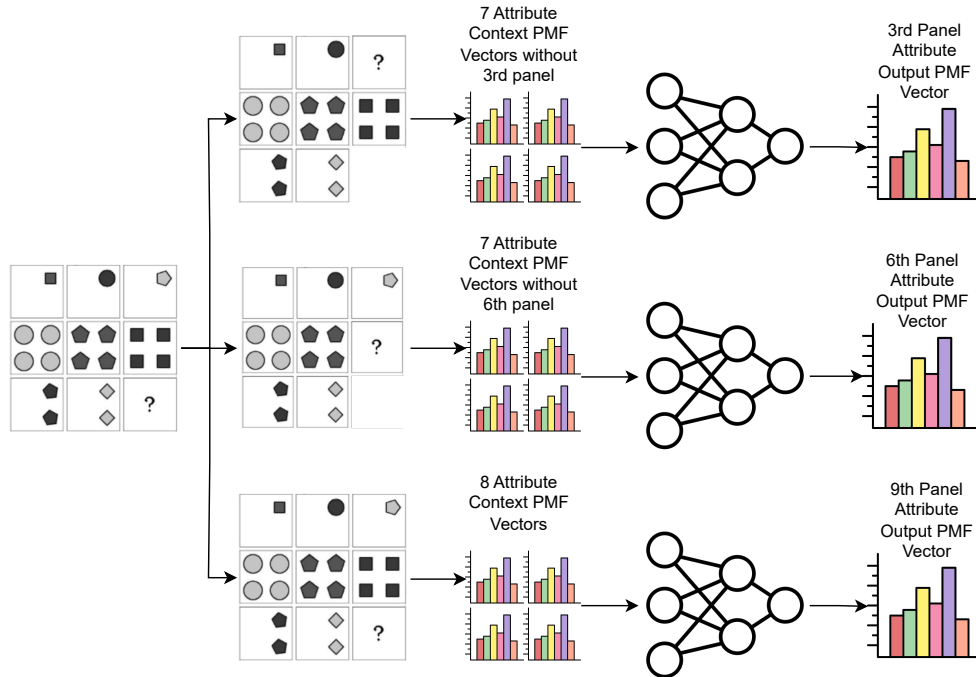
This variant emphasizes the potential of VSA vectors and their key-value binding mechanism, offering a flexible approach to representing varying panel types within the context.

## 2.5 AxR-Learners Framework with MLP

In the AxR-Learners Framework, each attribute is represented by five distinct models. However, during each training iteration, only one model per attribute is updated. Similarly, during inference, only one model per attribute is utilized. This design facilitates the bifurcation of the modeling process into two primary segments: model selection and model execution.

The model selection phase determines which among the five models will be employed for a specific attribute. Conversely, the model execution phase is responsible for the actual operation of the chosen model. This dual-phase approach is inspired by the research of Hersche et al. [hersche2023neuro]. The primary objective of this methodology is to individually model each rule, rather than encapsulating the entire transformation of an attribute. By segmenting each attribute model into multiple sub-models, the inherent complexity of each attribute model is further reduced. This segmentation introduces the Model Selector Module, which distinctly separates the intertwined processes of model selection and execution, as observed in prior attribute modeling frameworks.

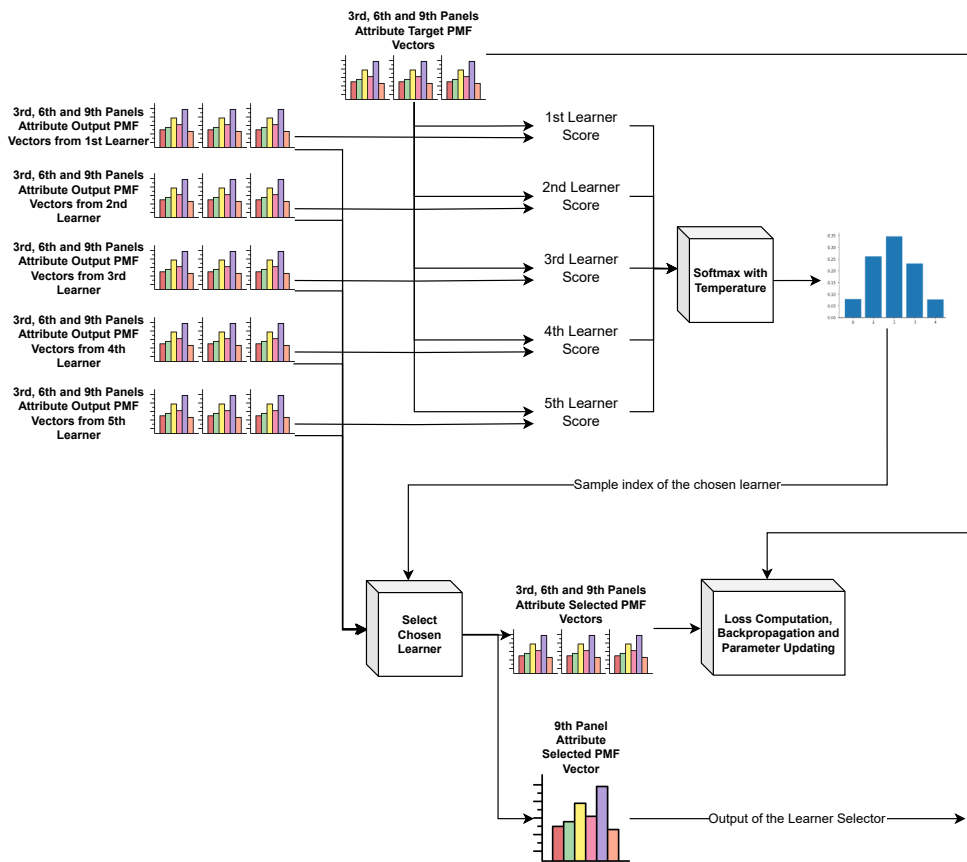
## 2.5.1 Learner Selector Module



**Figure 2.9:** Schematic representation of a learner in the AxR-Learners Framework, where each learner comprises three sub-learners.

In this framework, each primary MLP learner is further segmented into three sub-MLP learners. Each of these sub-learners predicts a distinct attribute PMF vector for a specific panel, as illustrated in Figure 2.9. The first sub-learner predicts the attribute PMF vector for the 3rd panel, excluding the 3rd panel's attribute PMF vector from its input. Similarly, the second sub-learner predicts for the 6th panel, and the third for the 9th panel. This modular structure aims to segregate the detection system, responsible for model selection, from the execution system, which operates the chosen model.

## 2.5. AxR-Learners Framework with MLP



**Figure 2.10:** Learner Selector Module's structure during the training phase.

During training, as depicted in Figure 2.10, the attribute PMF vectors generated by the sub-learners for panels 3, 6, and 9 are compared with their respective panels' attribute PMF vectors. This comparison yields scores, which, after undergoing a softmax transformation with a temperature parameter, produce a distribution over the five MLP learners. This distribution aids in sampling the learner index, which subsequently determines the learner to be updated.

## 2.5. AxR-Learners Framework with MLP

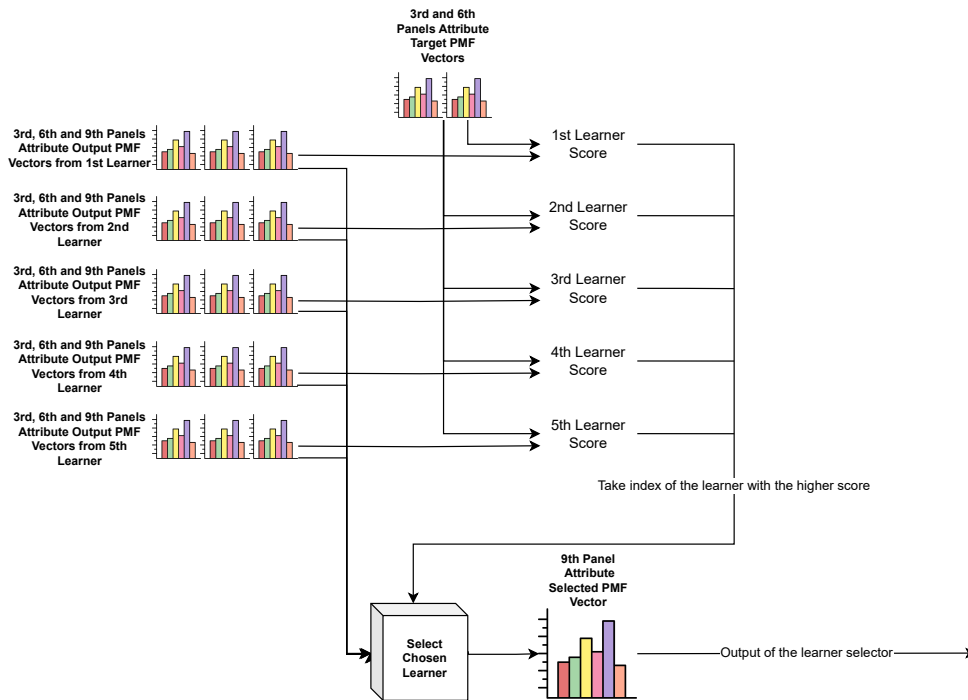


Figure 2.11: Learner Selector Module's structure during the inference phase.

Inference, as shown in Figure 2.11, slightly deviates from the training process. Since the 9th panel attribute PMF vector (the label) is inaccessible during inference, model selection relies solely on the 3rd and 6th panels. Post selection, the Learner Selector Module outputs the 9th panel attribute PMF vector from the chosen learner.

### Temperature to control Exploration-Exploitation Trade-Off

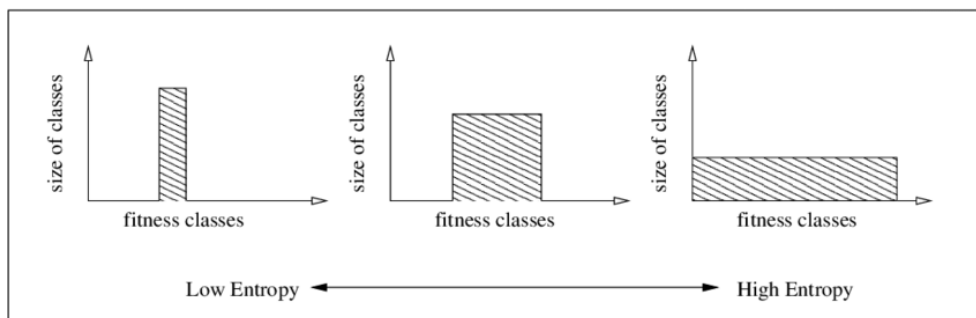
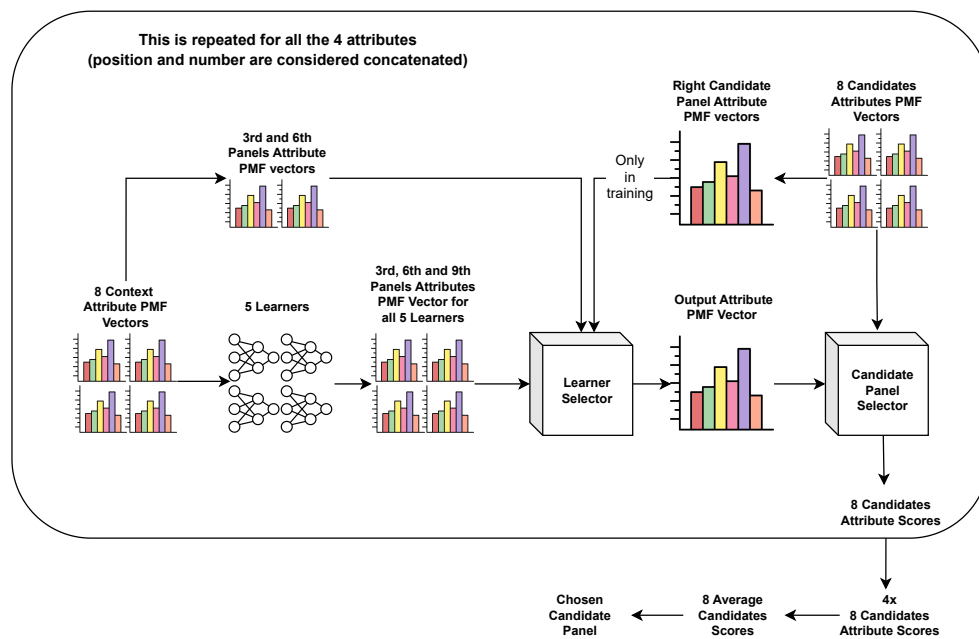


Figure 2.12: Visualization of entropy's influence on distribution shaping.

The temperature parameter in the AxR-Learners Framework serves as a control for the exploration-exploitation trade-off. Entropy, from a probabilistic perspective, gauges the expected information required to describe the outcome of a random variable  $X$ . As illustrated in Figure 2.12, entropy influences the sharpness of distributions. The softmax function, equipped with a temperature parameter  $T$ , can modulate the entropy of the generated distribution. This modulation facilitates the balance between exploration and exploitation, essential for effective rule learning in the framework.

### 2.5.2 PMF Variant

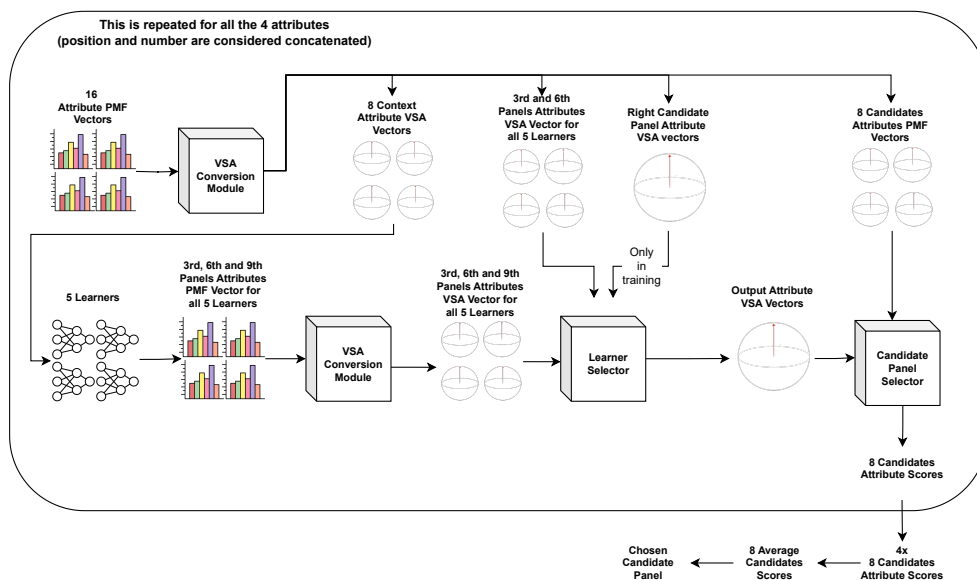


**Figure 2.13:** Schematic representation of the AxR-Learners Framework in the PMF variant. This variant utilizes PMF vectors for both input and output in the Multi-Layer Perceptron models for each attribute.

The PMF variant of the AxR-Learners Framework closely mirrors the PMF variant of the A-Learners Framework. As illustrated in Figure 2.13, each attribute begins with the PMF vectors from the eight context panels. These vectors are then fed into the five learners associated with the attribute, producing three output PMF vectors for each learner. These vectors are subsequently inputted into the Learner Selector, along with the actual PMF vectors from the 3rd and 6th panels. During training, the 9th panel's PMF vector is also included. The Learner Selector then updates and employs a chosen learner from the available five, outputting the 9th panel's PMF vector. This output is compared with the PMF vectors of all candidates through

the Candidate Panel Selector, resulting in a score for each candidate. The candidate panel with the highest score is then selected. Notably, the Position and Number attributes are treated as a singular attribute by concatenating their PMF vectors prior to input into the MLP learner, as depicted in 2.4.

### 2.5.3 VSA Variant

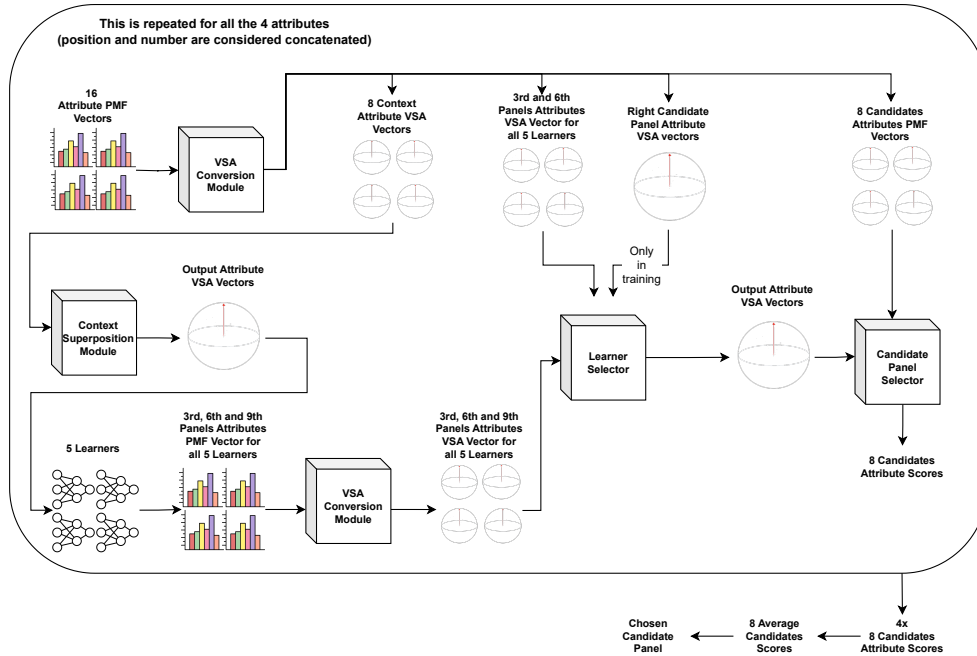


**Figure 2.14:** Schematic representation of the AxR-Learners Framework in the VSA variant. This variant employs VSA vectors as inputs to the Multi-Layer Perceptron models for each attribute, while converting the output PMF vectors into VSA vectors.

The VSA variant of the AxR-Learners Framework is analogous to the VSA variant of the A-Learners Framework. As depicted in Figure 2.14, each attribute commences with the PMF vectors from all 16 panels. These vectors undergo a transformation via the VSA Conversion Module, resulting in 16 VSA vectors for each panel. These vectors are then inputted into the five learners, producing three output PMF vectors for each learner. After converting these vectors into VSA format, they are inputted into the Learner Selector, along with the VSA vectors from the 3rd and 6th panels. During training, the VSA vector from the 9th panel is also incorporated. The Learner Selector then outputs the 9th panel's VSA vector, which is subsequently processed by the Candidate Panel Selector, similar to the PMF variant. Additionally, the Position and Number attributes are amalgamated into a single attribute by concatenating their VSA vectors before input into the MLP learners, as showcased in 2.6.

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

### 2.5.4 VSA with Context Superposition Variant



**Figure 2.15:** Schematic representation of the AxR-Learners Framework in the VSA with Context Superposition variant. This variant introduces a key-value binding of the context panels' VSA vectors, resulting in a singular VSA vector as input for the Multi-Layer Perceptron models of each attribute.

The VSA with Context Superposition variant of the AxR-Learners Framework primarily aligns with the VSA variant, with a notable distinction in the preprocessing of the context panels' VSA vectors. Prior to inputting the eight context panels' VSA vectors of each attribute into the associated five learners, a Context Superposition Module is applied, amalgamating them into a singular VSA vector. This consolidated vector is then fed into the five learners. The subsequent processes remain consistent with the VSA variant. Similarly, the Position and Number attributes are combined into a singular attribute by superposing their VSA vectors before input into the MLP learners, as demonstrated in 2.6.

## 2.6 Integration of Learnable Formula into the AxR-Learners Framework

The AxR-Learners Framework offers a unique advantage: the capability to model individual rules for each attribute. This opens the door to a novel class of models that leverage Vector Symbolic Architecture (VSA) representations.

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

In this section, we will delve into the translation of RAVEN test rules into the VSA space, construct an intuitive model based on these VSA rules, and discuss the merits and limitations of the proposed model.

### 2.6.1 Translating RAVEN Rules into the VSA Space

Hersche et al. [hersche2023neuro] demonstrated that by constructing a continuous VSA codebook using fractional power encoding, and subsequently translating the PMF vectors of attributes into VSA vectors, the rules in the RAVEN test can be reformulated as follows:

| Rule                    | VSA Formula  |
|-------------------------|--|
| Constant                | $a^{(i,3)} = a^{(i,2)} = a^{(i,1)}, i \in \{1, 2, 3\}$   |
| Arithmetic Addition     | $a^{(i,3)} = a^{(i,1)} \odot a^{(i,2)}, i \in \{1, 2, 3\}$   |
| Arithmetic Subtraction  | $a^{(i,3)} = a^{(i,2)} \otimes a^{(i,1)}, i \in \{1, 2, 3\}$   |
| Progressive Addition    | $a^{(i,3)} = a^{(i,2)} \odot (a^{(i,2)} \otimes a^{(i,1)}), i \in \{1, 2, 3\}$                                   |
| Progressive Subtraction | $a^{(i,3)} = a^{(i,2)} \otimes (a^{(i,1)} \odot a^{(i,2)}), i \in \{1, 2, 3\}$                                   |
| Triple Distribution     | $a^{(i,3)} = (a^{(1,1)} \odot a^{(2,1)} \odot a^{(3,1)}) \otimes (a^{(i,1)} \odot a^{(i,2)}), i \in \{1, 2, 3\}$ |

Table 2.3: Translation of RAVEN rules into VSA representations

Here,  $a^{(i,j)}$  represents the VSA vector of the panel situated in the  $i$ -th row and  $j$ -th column, derived using the continuous codebook. The symbols  $\odot$  and  $\otimes$  denote the binding and unbinding operations in the VSA space, which correspond to circular convolution and circular correlation, respectively.

The binding operation in the VSA space exhibits properties analogous to multiplication in the real number domain:

- **Commutativity:**

$$\mathbf{a} \odot \mathbf{b} = \mathbf{b} \odot \mathbf{a}$$

- **Associativity:**

$$(\mathbf{a} \odot \mathbf{b}) \odot \mathbf{c} = \mathbf{a} \odot (\mathbf{b} \odot \mathbf{c})$$

- **Neutral Element:** There exists an element  $\mathbf{e}$  such that:

$$\mathbf{a} \odot \mathbf{e} = \mathbf{a}$$

where  $\mathbf{e}$  typically comprises a vector with a 1 in the initial position and zeros elsewhere.

Contrastingly, the unbinding operation lacks both commutativity and associativity. However, when viewed collectively, the binding and unbinding operations in the VSA space mirror the properties of multiplication and

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

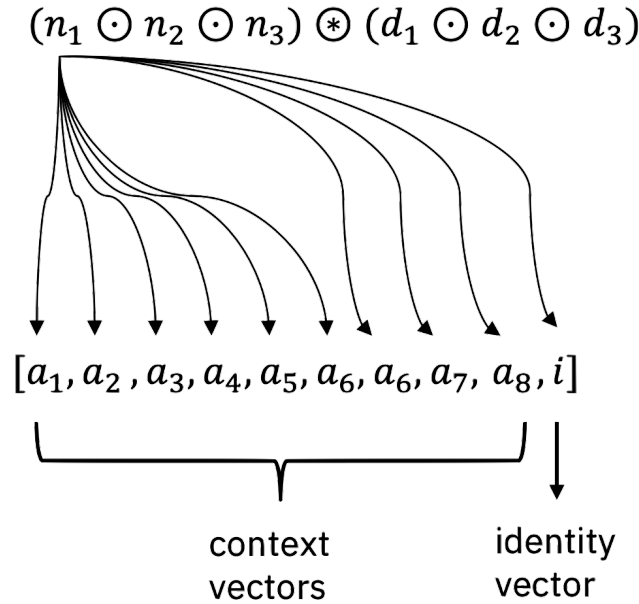
division in the realm of real numbers. This parallel allows us to represent a sequence of VSA operations in a fractional form, akin to a sequence of multiplicative and divisive operations among real numbers.

Given this analogy, a sequence involving six VSA vectors and a combination of binding and unbinding operations can be concisely represented as:

$$(\mathbf{a}_1 \odot \mathbf{a}_2 \odot \mathbf{a}_3) \circledast (\mathbf{a}_4 \odot \mathbf{a}_5 \odot \mathbf{a}_6)$$

By incorporating the neutral element for the binding operation, we can flexibly choose the number of VSA vectors to employ. This generalized fractional representation encompasses all the RAVEN rules in the VSA space as delineated in 2.3.

### 2.6.2 Introduction to the Learnable Formula Model



**Figure 2.16:** Schematic representation of the Learnable Formula model. Each term learns a distribution over the VSA vectors of the context panels' attributes, inclusive of the neutral element.

Building upon the previously introduced fractional form for VSA vectors, we present a novel model termed the "Learnable Formula." The essence of this model is to interpret each term in the VSA fractional form as a learnable

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

---

convex combination over the VSA vectors of the context panels' attributes, augmented with a neutral element  $\mathbf{e}$ . The neutral element is a VSA vector characterized by a value of 1 in all the blocks' initial positions. Formally, this can be represented as:

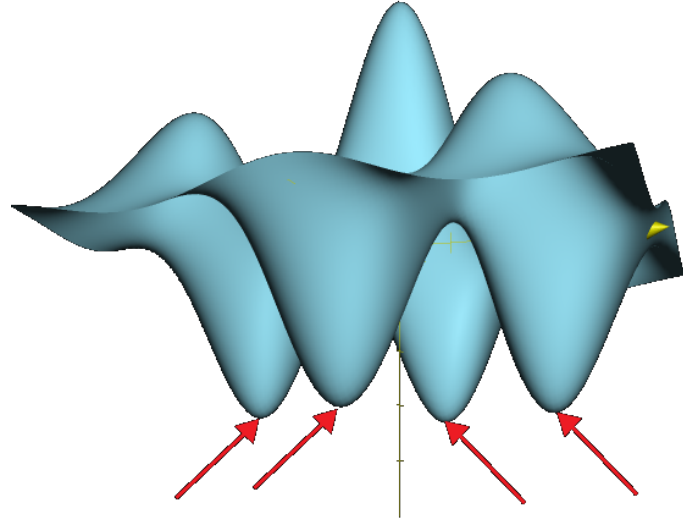
$$\mathbf{n} = \sum_{i=0}^n p_i \cdot \mathbf{a}_i + p_e \cdot \mathbf{e}$$

Given the nature of a convex combination, the following constraints apply:

$$\begin{aligned} \sum_{i=0}^n p_i + p_e &= 1, \\ 0 \leq p_i &\leq 1 \quad \forall i, \\ 0 \leq p_e &\leq 1 \end{aligned}$$

Here,  $\mathbf{n}$  denotes a term in the VSA fractional form,  $\mathbf{a}_i$  represents the VSA vector of the  $i$ -th panel's attribute, and  $p_i$  and  $p_e$  are the associated weights of the convex combination. The key insight is that each term of this model can be visualized as a distribution over the VSA vectors of the context panels' attributes, supplemented with the neutral element. The model can adaptively learn each rule in the VSA space, either by selecting the value of a specific context panel's attribute VSA vector or by opting for the neutral element, effectively excluding itself from the formula. This dynamic behavior is depicted in Figure 2.16. It's worth noting that this model is specifically tailored to represent individual rules of singular attributes, making it uniquely suited for this framework.

### 2.6.3 Loss Curve Characteristics and Probabilistic Interpretation



**Figure 2.17:** A conceptual representation of a potential loss surface for a VSA regression problem using the Learnable Formula model.

Before delving into the loss landscape, it's crucial to understand the inherent flexibility of the VSA fractional form. A single rule in the VSA space can manifest in multiple equivalent representations within this form. Consider, for instance, the Arithmetic Addition rule for the 9th panel:

$$\mathbf{a}^{(3,3)} = \mathbf{a}^{(3,1)} \odot \mathbf{a}^{(3,2)}$$

This rule can be equivalently expressed in the VSA fractional form in several ways, as demonstrated by the following representations:

$$\begin{aligned} \mathbf{a}^{(3,3)} &= (\mathbf{a}^{(3,1)} \odot \mathbf{a}^{(3,2)} \odot \mathbf{e}) \otimes (\mathbf{e} \odot \mathbf{e} \odot \mathbf{e}) \\ \mathbf{a}^{(3,3)} &= (\mathbf{a}^{(3,2)} \odot \mathbf{a}^{(3,1)} \odot \mathbf{e}) \otimes (\mathbf{e} \odot \mathbf{e} \odot \mathbf{e}) \\ \mathbf{a}^{(3,3)} &= (\mathbf{a}^{(3,1)} \odot \mathbf{a}^{(3,2)} \odot \mathbf{a}^{(3,2)}) \otimes (\mathbf{a}^{(3,2)} \odot \mathbf{e} \odot \mathbf{e}) \end{aligned}$$

The implication of this flexibility is profound. When a regression loss function is applied to the output VSA vector from the Learnable Formula and compared against the target VSA vector, the global minima of the loss curve correspond to different algebraic representations of the same rule. This phenomenon is conceptually illustrated in Figure 2.17, where the red arrows point to global minima, each representing a distinct algebraic form of the correct rule.

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

---

From a probabilistic perspective, the Learnable Formula model assigns independent probabilities to each vector in its domain. By leveraging the rule of product for independent events, we can construct a probability space encompassing all possible formulas in the VSA space with six terms:

$$P(\mathbf{f}) = P(\mathbf{n}_1) \cdot P(\mathbf{n}_2) \cdot P(\mathbf{n}_3) \cdot P(\mathbf{d}_1) \cdot P(\mathbf{d}_2) \cdot P(\mathbf{d}_3)$$

Here,  $P(\mathbf{f})$  denotes the probability mass associated with a specific formula in the VSA space, while  $P(\mathbf{n}_1), P(\mathbf{n}_2), \dots$  represent the probabilities assigned by each term in the Learnable Formula model to the respective terms of the formula  $\mathbf{f}$ .

### 2.6.4 Advantages of the Learnable Formula Model

The Learnable Formula model offers several distinct advantages:

- **Explainability:** The model’s structure inherently provides transparency. By identifying the index with the highest probability mass for each term in the Learnable Formula, we can reconstruct the VSA formula chosen by the model. Additionally, we can gauge the probability distribution assigned by each term to the VSA vectors of the context panels’ attributes.
- **Rules Hardcoding:** The model’s design facilitates the precise replication of the RAVEN rules in the VSA space, as outlined in Table 2.3. To embed a rule within the model, one can employ a Dirac distribution, concentrating all probability mass on a specific point. This is achieved by setting all convex combination coefficients to zero, except for the coefficient corresponding to the chosen VSA vector.
- **Low Parameter Count:** Each term in the Learnable Formula model comprises coefficients for each context panel’s attribute VSA vector, augmented with a coefficient for the neutral element. In scenarios utilizing all eight context panels, each term requires a mere nine parameters. Consequently, the entire Learnable Formula model necessitates learning only 54 parameters. This is remarkably efficient, especially when contrasted with a single layer of a typical MLP operating on 1024-dimensional VSA vectors, which demands  $1024 \times 1024 = 1048576$  parameters.

### 2.6.5 Model Limitations

The primary constraint of the Learnable Formula model lies in its expressiveness. To understand this, consider that in the VSA space, the sum of two attribute values is represented by the binding of their respective VSA vectors,

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

while the difference is depicted by the unbinding operation. Given the VSA fractional form underpinning the Learnable Formula:

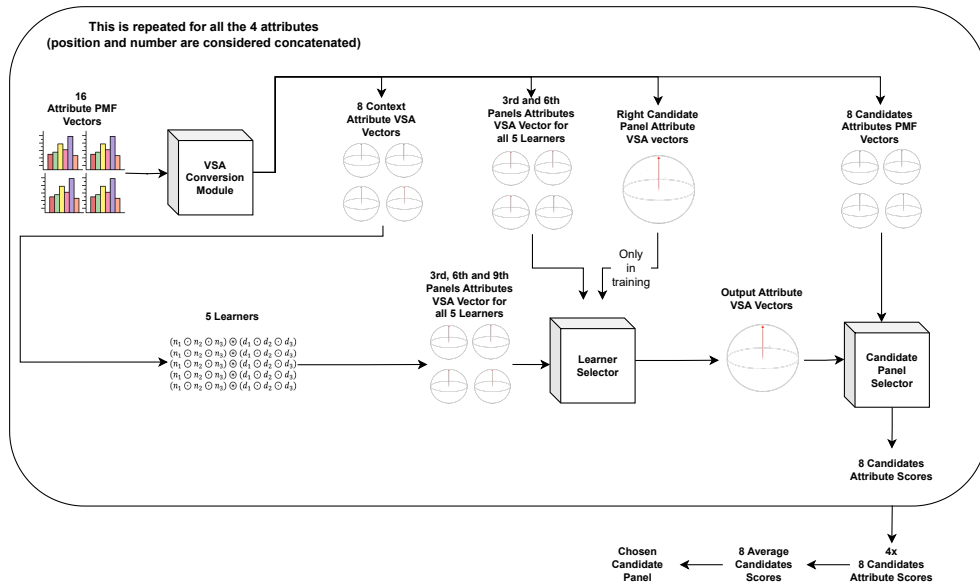
$$(\mathbf{a}_1 \odot \mathbf{a}_2 \odot \mathbf{a}_3) \otimes (\mathbf{a}_4 \odot \mathbf{a}_5 \odot \mathbf{a}_6)$$

In the numeric attribute space, this translates to:

$$(a_1 + a_2 + a_3) - (a_4 + a_5 + a_6)$$

This implies that the model is limited to representing sums and differences of attributes, precluding the possibility of modeling linear combinations. This limitation becomes evident in complex constellations like the 2x2 Grid or the 3x3 Grid, where the Position attribute employs logical rules on its bit-code vector.

### 2.6.6 Integration of Learnable Formula within the AxR-Learners Framework



**Figure 2.18:** Schematic representation of the AxR-Learners Framework incorporating the Learnable Formula. Within this architecture, each attribute's Learnable Formula model accepts all context panels' attribute VSA vectors as input and directly produces a VSA vector as output.

The Learnable Formula was specifically architected to seamlessly integrate within the AxR-Learners Framework. This integration is crucial given the formula's inherent capability to model individual rules for distinct attributes—a

## 2.6. Integration of Learnable Formula into the AxR-Learners Framework

---

feature uniquely facilitated by this framework. The framework's design, especially when incorporating the Learnable Formula, bears resemblance to the VSA variant of the AxR-Learners Framework that employs a Multi-Layer Perceptron (MLP). However, there are notable distinctions in the manner VSA vectors are input and processed.

Key differences include:

- **Input Handling:** In the VSA variant of the AxR-Learners Framework with MLP, VSA vectors from the context panels' attributes are concatenated before being fed into the MLP. In contrast, when using the Learnable Formula, these VSA vectors are input without concatenation. This design choice ensures that the Learnable Formula models can construct distributions over the individual VSA vectors, preserving their distinctiveness.
- **Output Processing:** The Learnable Formula models inherently produce VSA vectors as output. Consequently, there's no requirement for a separate VSA Conversion Module at the tail end of the five Learnable Formula models, simplifying the architecture.
- **Handling of Position and Number Attributes:** The treatment of the Position and Number attributes remains consistent with the VSA variant of the AxR-Learners Framework, barring the concatenation step. A detailed depiction of this process can be referenced in Figure 2.6.

In essence, the integration of the Learnable Formula within the AxR-Learners Framework offers a streamlined approach to modeling individual attribute rules, capitalizing on the strengths of both components.

### 2.6.7 Rule Sharing via the Learnable Formula: Unveiling the R-Learners Framework

The Learnable Formula model presents an avenue for the development of an enhanced framework, building upon the foundational AxR-Learners Framework. Within the AxR-Learners Framework, each attribute-rule combination is represented using a Learnable Formula model. However, an interesting observation from the RAVEN dataset reveals that the attributes of Type, Size, Color, and Number consistently share the same set of five rules. This observation suggests a potential optimization: instead of modeling each attribute-rule pair individually, one could focus on modeling these five shared rules directly. This insight led to the conceptualization of the R-Learners Framework. Here, five distinct Learnable Formula models are instantiated, and these models are subsequently shared across the aforementioned attributes. This innovative approach, termed 'rule sharing', streamlines the modeling process by reducing redundancy and promoting efficiency.

## 2.7 Comprehensive Overview of Frameworks and Variants

In wrapping up the Methods section, we present a consolidated view of all the frameworks and their respective variants explored in this study, as depicted in Table 2.4. A deeper dive into the results of these frameworks will be undertaken in the subsequent Results section. It’s worth noting that the accuracy metrics presented are exclusively derived from the I-RAVEN dataset, which has emerged as the benchmark for evaluations pertaining to the RAVEN task. In Table 2.4, the abbreviation “CS” denotes Context Superposition, representing the framework variants that have implemented this mechanism. The epoch count provided signifies the number of epochs required for a specific variant of the framework to stabilize, i.e., when no significant fluctuations in the validation loss value are observed.

| Framework    | Variant | Learner Type      | # Params            | I-RAVEN Acc. | Epochs    | Features/ Limitations  |
|--------------|---------|-------------------|---------------------|--------------|-----------|--|
| 1-Learner    | VSA     | MLP               | $\sim 7 \cdot 10^7$ | 55.11        | $\sim 10$ | VSA transformation can be perceived as a static linear layer applied to PMF vectors                              |
| 1-Learner    | VSA-CS  | MLP               | $\sim 2 \cdot 10^7$ | 54.69        | $\sim 10$ | CS optimizes model input size without compromising performance   |
| A-Learners   | PMF     | MLP               | $\sim 8 \cdot 10^7$ | 88.27        | $\sim 10$ | Decomposes into simpler classification tasks rather than a singular complex regression                           |
| A-Learners   | VSA     | MLP               | $\sim 3 \cdot 10^8$ | 87.9         | $\sim 10$ | Inherits limitations observed in the 1-Learner Framework   |
| A-Learners   | VSA-CS  | MLP               | $\sim 8 \cdot 10^7$ | 87.05        | $\sim 10$ | Retains advantages characteristic to the 1-Learner Framework   |
| AxR-Learners | PMF     | MLP               | $\sim 10^9$         | 80           | $\sim 10$ | Segregates detection and execution, albeit at the cost of increased parameters                                   |
| AxR-Learners | VSA     | MLP               | $\sim 4 \cdot 10^9$ | 78.17        | $\sim 10$ | Shares limitations with the 1-Learner Framework  |
| AxR-Learners | VSA-CS  | MLP               | $\sim 10^9$         | 78.38        | $\sim 10$ | Embraces the strengths of the 1-Learner Framework  |
| AxR-Learners | VSA     | Learnable Formula | $\sim 2 \cdot 10^4$ | 76.99        | $\sim 1$  | Provides model explainability and efficiency, albeit with constrained expressiveness                             |
| R-Learners   | VSA     | Learnable Formula | $\sim 5 \cdot 10^3$ | 78.32        | $\sim 1$  | Allows OOD generalization, further decreases the number of parameters and exploits rule sharing in RAVEN dataset |

**Table 2.4:** A synopsis of the characteristics and performance metrics of the explored frameworks and their variants.

---

## Experimental Results and Analysis

---

In this chapter, we delve into the results obtained from the various frameworks and their associated variants that were explored in this study. Our aim is to provide a comprehensive understanding of each experiment, elucidating the rationale behind every experimental direction. For the experiments involving the Multi-Layer Perceptron (MLP), we employed an MLP architecture comprising three hidden layers, each with 1024 neurons. Layer Normalization was incorporated between each layer pair. The learning rate was set at 0.0001, with a batch size of 32, and the training was conducted over 50 epochs. From these epochs, the model that exhibited the optimal validation loss was selected for evaluation on the test dataset. All the frameworks and their respective variants were rigorously tested across both the RAVEN and I-RAVEN datasets. This was done to ensure that the models were not merely capitalizing on potential biases present in the RAVEN dataset’s candidates. On the Vector Symbolic Architecture (VSA) front, we utilized 1024-dimensional VSA vectors partitioned into 4 blocks. A discrete randomly generated VSA codebook was employed for all methodologies, with the exception of the Learnable Formula approach, where a continuous random generation technique was adopted using fractional power encoding.

### 3.1 1-Learner Framework

#### 3.1.1 VSA Variant

This variant serves as the foundational step in our investigative journey, representing our preliminary approach to the problem at hand. Here, we sought to understand the rudimentary utilization of VSA representations. The VSA conversion can be conceptualized as a static linear projection transitioning from the PMF space. As evidenced in Table 3.1, the performance metrics are not particularly impressive. This is anticipated, given that this variant is our initial foray into the problem domain.

## 3.2. In-depth Analysis of the A-Learners Framework

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 48.59   | 50.3          | 37.3     | 36.8     | 62.1          | 53.85      | 48.55   | 51.25      |
| I-RAVEN | 55.11   | 61.75         | 37.6     | 27.95    | 75.85         | 47.85      | 66.7    | 68.1       |

**Table 3.1:** Performance metrics, in terms of test accuracies, for the VSA Variant of the 1-Learner Framework across different constellations in both RAVEN and I-RAVEN datasets.

### 3.1.2 VSA with Context Superposition Variant

In this subsection, our focus shifts to understanding the impact of Context Superposition on the model’s performance. As can be discerned from Table 2.2, the incorporation of Context Superposition does not detrimentally affect the model’s performance. This observation underscores the potential of leveraging the VSA vector’s Context Superposition mechanism without compromising on the contextual information’s integrity.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 47.99   | 47.1          | 38.5     | 38.2     | 60.45         | 54.25      | 48.2    | 49.25      |
| I-RAVEN | 54.69   | 58.45         | 39.15    | 30.3     | 74.2          | 48.2       | 65.85   | 66.7       |

**Table 3.2:** Performance metrics for the VSA with Context Superposition Variant of the 1-Learner Framework across different constellations in both RAVEN and I-RAVEN datasets.

## 3.2 In-depth Analysis of the A-Learners Framework

The A-Learners Framework is a cornerstone of our research, serving as a pivotal component in our exploration of machine learning models. This section is dedicated to a meticulous examination of its performance, intricacies, and the nuances across its various variants.

### 3.2.1 PMF Variant: A Deep Dive

The Probabilistic Model Formulation (PMF) variant emerges as the crown jewel in our suite of models, boasting the highest test accuracy. Given its stellar performance, we felt compelled to dedicate substantial resources and time to rigorously investigate this specific combination of framework and variant.

At its core, the architecture of this framework is ingeniously designed to harness the power of the PMF directly, enabling the formulation of four distinct sub-classification tasks. A comparative analysis with the results presented in Table 2.3 reveals a stark improvement in performance vis-à-vis the 1-Learner Framework. A cursory glance might lead one to attribute this performance leap to an increase in the number of parameters, especially since we deploy a dedicated model for each attribute. However, a detailed

### 3.2. In-depth Analysis of the A-Learners Framework

examination of Table 2.4 dispels this notion, revealing a mere incremental increase in the number of parameters from approximately  $7 \times 10^7$  to  $8 \times 10^7$ . This observation underscores the fact that the performance enhancement is not merely a byproduct of increased parameters but is indicative of the robustness and efficiency of the model itself.

#### PMF Variant Augmented with Cosine Loss

In our relentless pursuit of optimization, we ventured to integrate the Cosine Loss function into the PMF variant. This experiment aimed to discern the potential benefits of this loss function in enhancing the model’s efficacy. The results, meticulously tabulated in Table 3.3, offer a granular view of the model’s performance across diverse constellations in both the RAVEN and I-RAVEN datasets.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 90.35   | 98.15         | 90.15    | 73.1     | 98.65         | 73.35      | 99.5    | 99.55      |
| I-RAVEN | 88.27   | 99.2          | 89.4     | 64.5     | 99.75         | 65.45      | 99.85   | 99.8       |

**Table 3.3:** Test accuracies obtained in the PMF Variant of the A-Learners Framework using the Cosine Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets.

#### 3.2.2 Emulating Perception Uncertainty in the PMF Variant

A salient feature of our study was the deployment of a ‘pristine’ perception system, which consistently yielded ‘impeccable’ PMF vectors. However, real-world scenarios are riddled with inherent uncertainties in perception. To bridge this gap between the ideal and the real, we embarked on a journey to emulate perception uncertainty.

Our strategy was to convolve the PMF vectors with a Gaussian kernel of size 5. By modulating the standard deviations, we could simulate varying degrees of uncertainty. To ensure the integrity of our simulation and to preclude the learner from negating the linear transformation induced by the Gaussian smoothing, we devised an innovative mechanism. For each batch, prior to the Gaussian kernel’s generation, we constructed a Gaussian distribution, centered around the chosen sigma, with a standard deviation of 0.1. The ensuing standard deviation, sampled from this distribution, was employed to craft the Gaussian kernel for the smoothing process. This meticulous approach ensured that each batch experienced a unique standard deviation, thereby eliminating any possibility for the learner to counteract the linear transformation.

The results, as meticulously cataloged in Tables 3.4, 3.5, and 3.6, are a testament to the model’s resilience. It’s evident that the model is adept at

### 3.2. In-depth Analysis of the A-Learners Framework

handling a spectrum of uncertainties, even when confronted with escalating standard deviations in the Gaussian smoothing.

#### Simulation with Low Standard Deviation

In our first experiment, we subjected the model to a low standard deviation Gaussian smoothing of 0.1 sigma. The results, elucidated in Table 3.4, provide insights into the model’s robustness under minimal uncertainty.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 90.93   | 99.35         | 87.35    | 78.1     | 99.05         | 73.3       | 99.65   | 99.7       |
| I-RAVEN | 88.31   | 99.55         | 84.7     | 71.5     | 99.7          | 63.1       | 99.85   | 99.8       |

**Table 3.4:** Test accuracies obtained in the PMF Variant of the A-Learners Framework using the Cosine Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets and simulating perception uncertainty with a Gaussian smoothing with 0.1 sigma.

#### Simulation with Medium Standard Deviation

Pushing the boundaries further, we evaluated the model’s mettle under a medium standard deviation Gaussian smoothing of 1 sigma. The findings, presented in Table 3.5, shed light on the model’s adaptability and performance resilience under moderate uncertainty conditions.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 90.76   | 99.4          | 87.75    | 77.1     | 98.95         | 72.8       | 99.75   | 99.55      |
| I-RAVEN | 88.32   | 99.55         | 84.7     | 71.5     | 99.75         | 63.1       | 99.85   | 99.8       |

**Table 3.5:** Test accuracies obtained in the PMF Variant of the A-Learners Framework using the Cosine Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets and simulating perception uncertainty with a Gaussian smoothing with 1 sigma.

#### Simulation with High Standard Deviation

In our most challenging test, we assessed the model’s fortitude under a high standard deviation Gaussian smoothing of 5 sigma. The results, encapsulated in Table 3.6, underscore the model’s prowess in maintaining commendable performance even under severe simulated perception uncertainty conditions.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 90.88   | 99.4          | 87.75    | 78.1     | 98.95         | 72.8       | 99.65   | 99.55      |
| I-RAVEN | 88.65   | 99.5          | 86.45    | 72.55    | 99.75         | 62.55      | 99.9    | 99.85      |

**Table 3.6:** Test accuracies obtained in the PMF Variant of the A-Learners Framework using the Cosine Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets and simulating perception uncertainty with a Gaussian smoothing with 5 sigma.

### 3.2.3 Exploration of Loss Functions within the PMF Variant

The unique architecture of the A-Learners Framework, which facilitates the execution of four distinct sub-classification tasks as opposed to a singular regression task, offers a fertile ground for experimentation. Recognizing this potential, we embarked on a journey to explore various loss functions that are traditionally associated with classification tasks. Specifically, our attention was drawn to two prominent loss functions: the Cross Entropy Loss and the Kullback-Leibler Divergence Loss.

Our rationale for this exploration was twofold. Firstly, we aimed to ascertain if leveraging classification-centric loss functions could further optimize the performance of our model. Secondly, we were curious to understand the nuanced differences in model behavior and performance when subjected to these loss functions.

Upon close examination of the results presented in Table 3.7, it becomes evident that the Cross Entropy Loss neither enhances nor diminishes the model’s performance in comparison to the Cosine Loss. This observation is intriguing as it suggests that the model’s architecture and the nature of the data might play a more dominant role in determining performance than the choice of loss function, at least in this specific context.

On the other hand, the Kullback-Leibler Divergence Loss paints a slightly different picture. As delineated in Table 3.8, there is a marginal decline in test accuracies when this loss function is employed. This could be attributed to the inherent characteristics of the Kullback-Leibler Divergence Loss and its interaction with the model’s architecture and data.

To ensure a level playing field and facilitate a fair comparison with other Vector Symbolic Architecture (VSA) Variants, we introduced an initial linear layer in this variant. This layer serves as a bridge, transforming the concatenated PMF vectors into 1024-dimensional vectors, thereby standardizing the input dimensions across variants.

#### PMF Variant Enhanced with Cross Entropy Loss

The Cross Entropy Loss, a stalwart in the realm of classification tasks, was our first choice for experimentation. The results, meticulously tabulated below, offer insights into its efficacy within the context of the PMF Variant.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 90.1    | 98.2          | 88.35    | 73       | 99            | 72.95      | 99.7    | 99.5       |
| I-RAVEN | 89.49   | 99.25         | 92.15    | 67.6     | 99.4          | 68.4       | 99.8    | 99.8       |

**Table 3.7:** Test accuracies obtained in the PMF Variant of the A-Learners Framework using the Cross Entropy Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets.

### PMF Variant Employing Kullback–Leibler Divergence Loss

Our next foray was into the domain of the Kullback–Leibler Divergence Loss. This loss function, renowned for its ability to measure the difference between two probability distributions, presented an intriguing prospect. The results, presented below, shed light on its interaction with the PMF Variant.

|                | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|----------------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| <b>RAVEN</b>   | 88.88   | 98.55         | 86.85    | 66.7     | 99.05         | 71.8       | 99.6    | 99.6       |
| <b>I-RAVEN</b> | 87.75   | 99.35         | 83.5     | 63.9     | 99.5          | 68.25      | 99.85   | 99.9       |

**Table 3.8:** Test accuracies obtained in the PMF Variant of the A-Learners Framework using the Kullback–Leibler Divergence Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets.

#### 3.2.4 Exploring the VSA Variant

In our ongoing exploration of the A-Learners Framework, we turned our attention to the Vector Symbolic Architecture (VSA) Variant. The primary objective of this investigation was to discern how the employment of a fixed linear projection within the VSA conversion stacks up against the dynamic, learnable linear projection utilized in the PMF Variant. This comparison is crucial as it provides insights into the adaptability and efficiency of fixed versus learnable projections within the framework.

Upon analyzing the results presented in Table 3.9, a clear observation emerges. The VSA, despite its theoretical advantages, does not seem to offer any significant benefits over a generic learnable linear projection in this particular context. This finding prompts further inquiries into the underlying reasons and potential optimizations that might bridge this performance gap.

|                | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|----------------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| <b>RAVEN</b>   | 89.6    | 96.95         | 89.35    | 73.5     | 98.05         | 72.2       | 98.65   | 98.5       |
| <b>I-RAVEN</b> | 87.9    | 98.15         | 88.85    | 65.05    | 98.9          | 66         | 99.45   | 99.45      |

**Table 3.9:** Test accuracies obtained in the VSA Variant of the A-Learners Framework across the 7 different constellations in both RAVEN and I-RAVEN datasets.

#### 3.2.5 Delving into the VSA with Context Superposition Variant

Building on our previous investigations, we further delved into the intricacies of the A-Learners Framework by examining the VSA with Context Superposition Variant. Drawing inspiration from the 1-Learner Framework, our goal was to ascertain if the Context Superposition mechanism, which had shown promise in previous studies, could be seamlessly integrated and prove beneficial within this framework as well.

### 3.3. Detailed Analysis of the AxR-Learners Framework with MLP

The results, meticulously tabulated in Table 3.10, are illuminating. Mirroring the findings from the 1-Learner Framework, the VSA with Context Superposition Variant does not exhibit any pronounced dips in test accuracies. This consistency in performance is indicative of the robustness of the Context Superposition mechanism. It suggests that the mechanism adeptly preserves vital information, even when the context is introduced through this unique method.

|                | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|----------------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| <b>RAVEN</b>   | 89.01   | 96.8          | 88.4     | 70.75    | 97.65         | 72.4       | 98.5    | 98.55      |
| <b>I-RAVEN</b> | 87.05   | 97.6          | 87.05    | 61.75    | 98.7          | 65.55      | 99.35   | 99.35      |

**Table 3.10:** Test accuracies obtained in the VSA with Context Superposition Variant of the A-Learners Framework across the 7 different constellations in both RAVEN and I-RAVEN datasets.

## 3.3 Detailed Analysis of the AxR-Learners Framework with MLP

### 3.3.1 Introduction to the PMF Variant

The AxR-Learners Framework represents an innovative approach in machine learning, aiming to bifurcate the system into two distinct subsystems. The first is a detection system, which is tasked with determining the most suitable learner for a given input. The second is the execution system, which is responsible for implementing the rule chosen by the detection system. This division is hypothesized to enhance the adaptability and efficiency of the overall framework.

Our initial exploration is centered around the PMF Variant. As depicted in Table 3.11, while the test accuracy results are commendable, they do not surpass the benchmarks set by the A-Learners Framework. However, they do show a significant improvement when compared to the 1-Learner Framework. A granular examination of the results across various constellations reveals a pattern that closely mirrors the A-Learners Framework. However, a notable exception is observed in the "In Out 2x2" constellation, which displays a pronounced drop in test accuracy.

|                | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|----------------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| <b>RAVEN</b>   | 80.04   | 96.45         | 83.9     | 67.45    | 94.55         | 25.05      | 97.25   | 95.6       |
| <b>I-RAVEN</b> | 80      | 97.05         | 81.85    | 57.1     | 96.8          | 31.5       | 97.8    | 97.9       |

**Table 3.11:** Test accuracies obtained in the PMF Variant of the AxR-Learners Framework using the Cosine Loss across the 7 different constellations in both RAVEN and I-RAVEN datasets.

### 3.3.2 Rationale and Exploration of the PMF Variant with Simplified MLPs

In the PMF variant, the MLPs utilized as AxR-Learners were characterized by their complexity, consisting of three hidden layers with 1024 neurons in each layer. This intricate design prompted us to investigate the potential advantages of employing more streamlined MLPs. By reducing the architecture to just a single hidden layer, our objective was to determine if the larger MLPs might have been overfitting the data. This was especially pertinent given that our anticipations were closely aligned with the outcomes observed in the A-Learners Framework.

The insights derived from Table 3.12 shed light on this query. A slight decline in test accuracies is observed when using the more simplified MLPs. This nuanced change suggests that the larger, more complex MLPs in the previous variant were not merely overfitting. Instead, their complexity might have been a necessary design choice, tailored to cater to the intricacies and demands of the task they were designed for.

|                | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|----------------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| <b>RAVEN</b>   | 78.73   | 90.45         | 83.3     | 66       | 94.55         | 25.75      | 95.4    | 95.65      |
| <b>I-RAVEN</b> | 79.25   | 94.75         | 79       | 56.9     | 97.2          | 32.35      | 96.35   | 98.2       |

**Table 3.12:** Test accuracies obtained in the PMF Variant of the AxR-Learners Framework across the 7 different constellations in both RAVEN and I-RAVEN datasets using smaller MLPs with only 1 hidden layer.

### 3.3.3 Detailed Analysis of the VSA Variant in the AxR-Learners Framework

In the AxR-Learners Framework, we further delved into the implications of utilizing Vector Symbolic Architectures (VSA) representations. Specifically, we were interested in understanding the effects of adopting a fixed input linear projection, represented by VSA, on the overall performance of our system. Contrary to the outcomes observed in the other two frameworks, the results here indicate a slight decline in test accuracies when VSA vectors are employed. A plausible explanation for this could be the inherent nature of VSA. Given that VSA relies on a discrete random generated codebook, it might pose challenges when the system is tasked with arithmetic operations at the rule level.

### 3.4. In-depth Examination of the AxR-Learners Framework with the Learnable Formula Model

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 78.41   | 92.45         | 81.4     | 64.25    | 92.8          | 26.1       | 95.9    | 94.05      |
| I-RAVEN | 78.17   | 94.85         | 76       | 54.9     | 97.3          | 30.95      | 97.25   | 95.95      |

**Table 3.13:** Test accuracies obtained in the VSA Variant of the AxR-Learners Framework across the 7 different constellations in both RAVEN and I-RAVEN datasets.

#### 3.3.4 Exploration of the VSA with Context Superposition Variant in the AxR-Learners Framework

Continuing our exploration, we also probed the effects of integrating the Context Superposition mechanism within the AxR-Learners Framework. This mechanism has shown promise in the other two frameworks, and our objective was to ascertain its efficacy here. As illustrated in Table 3.14, the results are consistent with our previous observations. There isn't a significant drop in test accuracies, reinforcing the notion that the Context Superposition mechanism does not lead to a substantial loss of contextual information.

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| RAVEN   | 77.68   | 89.2          | 82.45    | 64.6     | 90.8          | 26.4       | 95.05   | 95.25      |
| I-RAVEN | 78.38   | 92.15         | 79.55    | 55.3     | 95.75         | 31.6       | 96.85   | 97.45      |

**Table 3.14:** Test accuracies obtained in the VSA with Context Superposition Variant of the AxR-Learners Framework across the 7 different constellations in both RAVEN and I-RAVEN datasets.

### 3.4 In-depth Examination of the AxR-Learners Framework with the Learnable Formula Model

In our exploration of various frameworks, the AxR-Learners Framework with the Learnable Formula model stands out as a unique and innovative approach. This framework introduces a novel model, the Learnable Formula, which is designed to adapt and evolve based on the data it encounters. One of the distinguishing features of this framework is the adoption of a continuous random generated Vector Symbolic Architectures (VSA) codebook. This codebook, generated using fractional power encoding, contrasts with the discrete VSA codebooks used in other frameworks with Multi-Layer Perceptrons (MLPs). The continuous nature of this encoding facilitates the encoding of attributes in a more fluid manner, making the learning of arithmetical RAVEN rules more intuitive and efficient.

Another noteworthy aspect of the Learnable Formula model is its rapid learning capability. Compared to other models explored in this study, the Learnable Formula model exhibits a significantly faster learning rate, achieving convergence in approximately 1 epoch as opposed to the typical 10

### 3.5. In-depth Analysis of Rule Sharing Across Attributes in the R-Learners Framework

epochs observed with other models. This accelerated learning allowed us to employ a larger learning rate; while previous experiments utilized a learning rate of 0.0001, in this framework, we adopted a learning rate of 0.1.

The results, as presented in Table 3.15, indicate that the Learnable Formula model is competitive when juxtaposed with the use of MLPs within the same framework. However, as anticipated, there is a noticeable decline in performance in the grid constellations. This can be attributed to the inherent limitations of the Learnable Formula model, as detailed in the Methods section. Specifically, the model struggles with logical rules applied to the Position attribute and cannot discern the mutual exclusivity between rules applied to the Position and Number attributes.

|                | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|----------------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| <b>RAVEN</b>   | 76.64   | 96.15         | 73.25    | 68.3     | 88.4          | 26.7       | 92.55   | 91.1       |
| <b>I-RAVEN</b> | 76.99   | 95.75         | 73.75    | 50.95    | 96.45         | 33.4       | 92.95   | 95.65      |

**Table 3.15:** Test accuracies obtained in the AxR-Learners Framework with the Learnable Formula model across the 7 different constellations in both RAVEN and I-RAVEN datasets.

### 3.5 In-depth Analysis of Rule Sharing Across Attributes in the R-Learners Framework

The R-Learners Framework, building upon the foundation of the Learnable Formula models, offers a novel approach to sharing rules across distinct attributes. This methodology, while promising in certain scenarios, exhibits varied performance across different constellations. This section provides a detailed, technical exploration of the framework’s performance, highlighting its strengths and pinpointing areas of potential improvement. Drawing from the empirical data presented in Table 3.16, it becomes evident that utilizing a uniform set of Learnable Formula models across different attributes is particularly effective in simpler constellations. In scenarios where the Position and Number attributes are not involved, the R-Learners Framework’s implementation of the Learnable Formula consistently outperforms its counterpart in the AxR-Learners Framework. However, the efficacy of this approach diminishes in more intricate constellations, such as the 2x2 Grid or the 3x3 Grid. A plausible explanation for this observed discrepancy lies in the inherent differences in rule sets across attributes. Specifically, while the Type, Size, and Color attributes share a common set of rules, the Position attribute diverges in its rule set. When the same learned rules are applied to the Position attribute, the model encounters inconsistencies, leading to suboptimal performance. This underscores the importance of tailoring rule sets to specific attributes, especially in complex constellations, to ensure robust and accurate modeling.

### 3.6. Out-Of-Distribution (OOD) Generalization Experiments

|         | Average | Center Single | 2x2 Grid | 3x3 Grid | In Out Single | In Out 2x2 | Up Down | Left Right |
|---------|---------|---------------|----------|----------|---------------|------------|---------|------------|
| I-RAVEN | 78.32   | 98.85         | 69.95    | 52       | 95.95         | 32.3       | 99.35   | 99.85      |

**Table 3.16:** Test accuracies obtained in the R-Learners Framework with the Learnable Formula model across the 7 different constellations in both RAVEN and I-RAVEN datasets.

## 3.6 Out-Of-Distribution (OOD) Generalization Experiments

In the realm of machine learning, especially in the context of visual reasoning tasks, the ability of a model to generalize to unseen or novel scenarios is of paramount importance. This generalization capability is often tested by evaluating the model’s performance on out-of-distribution (OOD) data, which refers to data that is not represented in the training set.

In our study, we specifically assessed our model’s proficiency in handling unseen target attribute-rule pairs. For instance, if the model was trained on all attribute-rule combinations except for a specific target pair, such as the constant rule applied to the type attribute, how well would it perform when tested exclusively on this omitted pair? This experimental setup provides a rigorous test of the model’s OOD generalization capabilities, particularly for attribute-rule pairs.

To facilitate this, we constructed new training and validation datasets by filtering out examples containing the target attribute-rule pair from the existing RAVEN and I-RAVEN datasets. Consequently, the test set was exclusively populated with examples of the target attribute-rule pair. It’s worth noting that the size of these datasets was reduced, contingent on the specific attribute-rule pair being targeted.

Our exploration encompassed three distinct cases:

1. The A-Learners Framework employing Multi-Layer Perceptrons (MLPs).
2. The AxR-Learners Framework utilizing Learnable Formulas.
3. A variant of the AxR-Learners Framework where Learnable Formulas are shared across different attributes.

The results, as presented in Tables 3.17, 3.18, and 3.19, reveal some intriguing insights. The A-Learners Framework with MLPs exhibited suboptimal performance, which can be attributed to the inherent limitations of MLPs in handling OOD data. On the other hand, the AxR-Learners Framework with Learnable Formulas demonstrated improved performance, albeit not optimal. This can be rationalized by understanding that each attribute-rule pair in this framework is modeled by a distinct Learnable Formula. If a specific pair is not observed during training, its corresponding Learnable Formula remains

### 3.6. Out-Of-Distribution (OOD) Generalization Experiments

untrained. However, the partial success can be ascribed to the model’s ability to leverage other learned arithmetic formulas for a given attribute, which might occasionally align with the new rule encountered.

The third case, where Learnable Formulas are shared across attributes, unveils the versatility of the Learnable Formula approach. Given that the arithmetic rules in the RAVEN dataset are consistent across the Type, Size, and Color attributes (as observed in the Center Single constellation), a shared set of Learnable Formulas can be employed across these attributes. This strategy not only mitigates the need for a unique learner for each attribute-rule pair but also bolsters OOD generalization. Moreover, it offers the added benefit of reducing the overall number of parameters in the framework.

|  | Constant | Progress. | Dist.3 |
|--|----------|-----------|--------|
| A-Learners MLP   | 14.8     | 14.85     | 30.2   |
| AxR-Learners<br>Learnable Formula                      | 97.05    | 38.25     | 41.15  |
| AxR-Learners<br>Learnable Formula<br>with shared rules | 93.55    | 99        | 100    |

**Table 3.17:** Out-Of-Distribution generalization on the unseen rule-attribute pairs of the I-RAVEN dataset in the Center Single constellation for the Type attribute. We report accuracy (%) on test set that contains exclusively examples with the target attribute-value pairs on which it has not been trained on.

|  | Constant | Progress. | Dist.3 | Arithmetic |
|--|----------|-----------|--------|------------|
| A-Learners MLP   | 27.75    | 74.95     | 46.95  | 46.55      |
| AxR-Learners<br>Learnable Formula                      | 99.4     | 73        | 45.15  | 55.1       |
| AxR-Learners<br>Learnable Formula<br>with shared rules | 100      | 96.55     | 99.4   | 99.8       |

**Table 3.18:** Out-Of-Distribution generalization on the unseen rule-attribute pairs of the I-RAVEN dataset in the Center Single constellation for the Size attribute. We report accuracy (%) on test set that contains exclusively examples with the target attribute-value pairs on which it has not been trained on.

### 3.6. Out-Of-Distribution (OOD) Generalization Experiments

---

|  | Constant | Progress. | Dist.3 | Arithmetic |
|--|----------|-----------|--------|------------|
| A-Learners MLP   | 56.25    | 60.5      | 44.35  | 48.85      |
| AxR-Learners<br>Learnable Formula                      | 93.6     | 72.75     | 42.55  | 52.4       |
| AxR-Learners<br>Learnable Formula<br>with shared rules | 98.4     | 100       | 98.8   | 100        |

**Table 3.19:** Out-Of-Distribution generalization on the unseen rule-attribute pairs of the I-RAVEN dataset in the Center Single constellation for the Color attribute. We report accuracy (%) on test set that contains exclusively examples with the target attribute-value pairs on which it has not been trained on.

# Conclusions

---

In the course of this master thesis, we embarked on an in-depth exploration of various frameworks and models to tackle the challenges posed by the RAVEN dataset. Our findings offer several insights into the capabilities and limitations of these approaches.

Firstly, our experiments with Vector Symbolic Architectures (VSA) in neural contexts revealed that it essentially functions as a non-learnable linear projection. There doesn't appear to be a significant advantage in employing VSA over a learnable linear transformation applied directly to the PMF attribute vectors. However, a notable benefit of VSA representations is the feasibility of applying the role-filling key-value composition on the panels, which we have termed 'context superposition' in this work. This mechanism provides a consistent input dimensionality to the models, irrespective of the number of panels. The application of this mechanism doesn't seem to compromise performance, suggesting that there might be minimal information loss when it is employed.

Our investigations into the behavior of different frameworks led us to conclude that relying on a singular MLP model for the entire reasoning system is suboptimal. Transitioning to a model-per-attribute approach not only simplifies the task by converting a complex regression problem into four more manageable classification tasks at the PMF level but also yields a significant boost in performance. We also explored the potential of different classification losses to enhance performance in the PMF context, the results did not indicate any substantial improvements.

The AxR-Learners Framework emerged as a pivotal component of our research. Although initial attempts to model individual rules for each attribute using MLPs did not outperform the A-Learners Framework, this exploration paved the way for the introduction of the Learnable Formula model. While it may not be the top-performing model among those we investigated, the

---

Learnable Formula model boasts several advantages, including explainability, rapid training (achieving convergence in just one epoch), and a reduced parameter count. A particularly intriguing feature of this model is its ability to share learners across different attributes, capitalizing on the shared rule structure among the Type, Size, and Color attributes in the RAVEN dataset. This 'rule sharing' characteristic empowers the Learnable Formula model to excel in out-of-distribution scenarios, making it a unique and valuable asset in our toolkit.

In summary, this master thesis has shed light on the intricacies of abstract reasoning in the context of the RAVEN dataset and has provided a roadmap for future research endeavors in this domain.

---

## Bibliography

---

- [1] H. Gardner, “Frames of mind: The theory of multiple intelligences”, *Basic books*, 1983.
- [2] G. S. Halford, W. H. Wilson, and S. Phillips, “Rethinking intelligence: The role of implicit processes in higher order cognition”, *Implicit cognition*, pp. 137–158, 2005.
- [3] J. Raven, J. Raven, and J. Court, “Raven’s progressive matrices: Change and stability over culture and time”, *Cognitive psychology*, vol. 41, no. 1, pp. 1–48, 2000.
- [4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks”, pp. 1097–1105, 2012.
- [5] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks”, pp. 91–99, 2015.
- [6] H. Nam, M. Baek, and B. Han, “Learning multi-domain convolutional neural networks for visual tracking”, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4293–4302, 2016.
- [7] T.-H. Le, V. Tran, S. Venkatesh, and D. Phung, “Neural-symbolic vqa: Disentangling reasoning from vision and language understanding”, *Advances in Neural Information Processing Systems*, vol. 33, 2020.
- [8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need”, *Advances in neural information processing systems*, vol. 30, 2017.
- [9] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, “Neuro-symbolic vqa: Disentangling reasoning from vision and language understanding”, *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [10] P. Kanerva, “Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors”, *Cognitive Computation*, vol. 1, no. 2, pp. 139–159, 2009.

- 
- [11] R. W. Gayler, "Distributed representations of structure: A theory of analogical access and mapping", *Psychological Review*, vol. 110, no. 3, p. 617, 2003.
- [12] L. Hersche *et al.*, "Neural vector symbolic architectures for abstract reasoning", *arXiv preprint arXiv:2203.04571*, 2022.
- [13] G. E. Hinton, J. L. McClelland, and D. E. Rumelhart, "Distributed representations", *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 1, no. 3, pp. 77–109, 1986.
- [14] C. Zhang, F. Gao, B. Jia, Y. Zhu, and S.-C. Zhu, "Raven: A dataset for relational and analogical visual reasoning", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5317–5327, 2019.
- [15] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "Vqa: Visual question answering", *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2425–2433, 2015.
- [16] M. Malinowski and M. Fritz, "A multi-world approach to question answering about real-world scenes based on uncertain input", pp. 1682–1690, 2014.
- [17] A. Newell and H. A. Simon, "Computer structures: Readings and examples", 1976.
- [18] T. A. Plate, "Holographic reduced representations", *IEEE Transactions on Neural Networks*, vol. 6, no. 3, pp. 623–641, 1995.
- [19] P. Kanerva, "Sparse distributed memory", 1988.
- [20] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, "A large-scale model of the functioning brain", *Science*, vol. 338, no. 6111, pp. 1202–1205, 2012.
- [21] T. A. Plate, "Distributed representations and nested compositional structure", *Machine learning*, vol. 28, no. 2-3, pp. 73–101, 1997.
- [22] —, "Holographic reduced representations: Convolution algebra for compositional distributed representations", *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 1, pp. 30–35, 1991.
- [23] C. Eliasmith, "A flexible and efficient model of neurobiological computation", *Neural Networks*, vol. 21, no. 10, pp. 1260–1270, 2008.
- [24] G. Marcus, "Deep learning: A critical appraisal", *arXiv preprint arXiv:1801.00631*, 2018.
- [25] L. Bottou, "From machine learning to machine reasoning", *Machine learning*, vol. 94, no. 2, pp. 133–149, 2014.

- [26] J. C. Raven, "Progressive matrices: A perceptual test of intelligence", 1938.
- [27] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people", *Behavioral and brain sciences*, vol. 40, 2017.
- [28] A. d. Garcez, L. C. Lamb, and D. M. Gabbay, "Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning", *Journal of Applied Logic*, vol. 37, pp. 152–170, 2019.
- [29] D. Poole, "Probabilistic horn abduction and bayesian networks", *Artificial Intelligence*, vol. 64, no. 1, pp. 81–129, 1993.